Macintosh!
COMPLETE

by Doug Clapp
Macintosh! COMPLETE

by Doug Clapp

SOFTALK BOOKS
This book, the first, is for my parents
Acknowledgements xi

Introduction xiii

Chapter 1 What Is a Macintosh? 1
What Macintosh Isn't . . . The Question Answered . . .
Performance . . . Build-a-Mac

Chapter 2 A Short History of Apple and Mac's Creation 6
Apple's Beginnings . . . Explaining Success . . .
Price and Performance . . . From Whence Mac Sprang . . .
The Demise of Modes . . . Mac's Mom . . . The Wheel Turns . . .
The People

Section I: Operation

Chapter 3 Simple Concepts 18
The Desktop . . . Walnut or Cherry? . . . Windows . . .
In Four Words
Chapter 4 Expressing Your Wishes
The Mouse . . . Pointer Shapes . . . Mouse Manuevers . . .
Special Keys . . . Backspacing . . . Other Keyboard Features . . .
The Numeric Keypad . . . Guessing Ahead

Chapter 5 The Finder: Macintosh's Operating System
The Finder Unveiled . . . Fundamental Thoughts . . .
Desk Accessories . . . The Control Panel . . .
Wherever, Whenever . . . Still to Come

Chapter 6 Windows
Other Paneless Encounters

Chapter 7 Dialogs and Alerts: More Windows
Just the Same, Only . . . When They Come, How They Look . . .
Modal and Modeless Dialogs . . . Alert! Boxes . . . Controls

Chapter 8 Text Editing
Real World Writing

Chapter 9 Cut, Copy, and Paste
The Glamour Trio . . . Pasting Madness . . .
Large Selections and Extending Selections . . .
Shift-clicking Discontiguous Objects . . .
Clipboarding the Desk Accessories . . . A Monstrous Digression . . .
Characters, Numbers, and Symbols . . . Nothing but Bytes . . .
Programs and Data . . . Data Structures . . . ROM . . . RAM . . .
The Heap . . . The Consistency of Macintosh

Chapter 10 Beginning Diskology
Volumes and Filenames . . . Physical Volumes, Logical Volumes

Section II: Software

Chapter 11 Other Tools for Macintosh
Programming . . . Educational Opportunities . . .
Does Anyone Play Games? . . . The Magic Program
Chapter 12  Macintosh MacWrite  122
The First Word Processor ... Features ... MacWrite Text Features ... MacWrite Printing Features ... Enough, Enough! ... Know Your Needs ... Bring on the Waxers! ... The Point ... The Marriage of Mac ... MacWrite's Place

Chapter 13  A Few Lines on MacPaint  132
What Is It? ... What's It Good For? ... How's It Work? ... What Can It Do? ... What'll It Do Next?

Chapter 14  Microsoft Wares  142
Multiple Microsoft Programs ... A Closer Look ... Functions ... Joy or Dismay? ... Chart and Multiplan

Chapter 15  What To Use a Macintosh For  150

Section III: Programming

Chapter 16  Learning To Program  156
Problem Solving ... Algorithms ... Interpreters and Compilers ... Syntax, Problem Definition, and the Joy of Hacking ... Learning the Art

Chapter 17  Programming Languages  163
Confused? ... Users Groups

Chapter 18  MacBASIC: A Preview  169
Concurrency ... Optional Line Numbers ... Labeled GOTOs and GOSUBs ... Flow of Control Structures ... Data Types ... Graphics ... Inside the User Interface ... In Conclusion

Chapter 19  Macintosh Pascal  180
Pascal Made Easy

Section IV: Hardware

Chapter 20  Taking Care of Mac  186
Routine Care ... What We Have To Fear ... Screen Care ... Dust and Fingerprints ... Dust and Disks ... More About Dust ... Ventilation ... Sensible Precautions ... Insuring Mac at Home ... Insuring Mac at Work ... A Final Thought
Chapter 21  A Look at Mac's Back  195
The Back Panel . . . Serial Considerations . . . Sound

Chapter 22  Inside Macintosh  200
A Peek Inside . . . Mr. Smith Goes to Town . . .
The Motherboard . . . Our PALs . . . The Analog Board . . .
Building Macs . . . Details Forthcoming

Chapter 23  Inside Macintosh ROMs  209
How It Was Done . . . Accessing Routines . . .
Breaking Down ROM . . . The Operating System . . .
QuickDraw . . . The Toolbox . . . Digging Into ROM

Chapter 24  The Motorola MC68000 Microprocessor  215
Architecture and Design . . . Registers . . . Speed

Section V: Epilogue

Chapter 25  Hints, Tips, and Advice  226
Attitude . . . Workstyles . . . Traveling With Mac . . .
Extracurricular Activities . . . Change of Setting . . .
Telecommunications . . . Other Hobbies; In the Future . . .
When Mac is Old and Gray

The People Who Made Macintosh  236

Glossary  239

Further Reading  271

Further Software and Hardware  287

Appendices

A: Type Fonts  293
B: Hardware Specifications  303
C: Standard ASCII Control Codes  305
D: Standard Key Character Formats  307
E: MacBASIC Keywords, Data Types, Operators, and Precedence  311
F: 68000 Instruction Set  315

Index  317
The author wishes to thank the following people for their contributions, support, aid, and advice:

Steve Jobs, whose existence made Apple possible, Macintosh possible, and, certainly least of all, made this book possible. Steve's innovative presence permeates Apple, the Macintosh group, and Mac itself. I owe you one, Steve. A big one.

Mike Murray of Apple Computer for withstanding a barrage of badgering and remaining helpful and courteous to the end. Thanks, Mike.

The Macintosh programming group: Jerome Coonen, Bill Atkinson, Steve Capps, Donn Denman, Andy Hertzfeld, Bruce Horn, Susan Kare, Larry and Patti Kenyon, and Rony Sebok. They were of more help to me than I can possibly express: they allowed me a multitude of stupid questions, explained patiently, granted lengthy interviews, allowed me to showcase their work in this book (thank you, Susan), and even took me on picnics. As programmers, and as people, they are unmatched. Thank you.

Many other people at Apple's Macintosh division, whose contributions are varied and too numerous to single out: Chris Espinosa, Dan Kottke, Mike Boich, Mary Ellen MacCammon, Bob Belleville, Dave Egner, Bud Colligan, Dennis Grimm, Guy Kawasaki, Carol Kaehler, Sandy Tompkins, Burrell Smith, Caroline Rose, and Joe Shelton. You all helped, and you all know how. Thank you.

Microsoft's Mark Ursino for his help in fielding frequent questions, and Microsoft Corporation, for permission to use a paragraph, and a list of func-
tions from their Multiplan manual. Good companies are made up of good people; Microsoft, like Apple, is testimony to that truth.

Gerald P. Rafferty, mastermind of Softalk's book division; the kind of person that authors wish existed. This one does. Thank you, Gerald.

Paul Mithra, my editor: great editor, great guy. Anything you like about this book is Paul's doing; anything you don't like is probably my fault. Thanks, Paul.

Margot Tommervik, one of the best-loved people in all of computerdom. Now I know why.

Al Tommervik, husband of Margot, publisher of four magazines, one book division, and even a notorious cassette of computer music. Other corporations have management divisions; Softalk has Al Tommervik. His intuitions and business judgements are unmatched, he works like a dog, and he's a terrific host [when forcibly torn away from work]. Al flew me out, put me up in style, yelled and screamed with me til 5 in the morning, then tossed his American Express card across the table—thereby insuring that I would, after all, make it to Florida for Christmas.

I've never had so much fun. Almost makes you want to write another book.

Mike Griffin, friend and programmer extraordinaire, for his wonderful word processor Palantir. I used Palantir to write this book, and it never failed me. Thanks, Mike.

Claudia, my wife, who handled every non-book aspect of both our lives—and there were many—while this book was being created. As always, Claudia was magnificent. Thank you, dear.
Welcome. Come on in, have a seat, put your feet up, get comfortable; we’ve much to discuss and explore.

As a new or prospective Macintosh owner, you’re probably interested in what Macintosh is, what it can do, and, most important, what it can do for you.

It can do a lot—more than you might think, more than you might imagine. Some people believe that Macintosh represents a new, advanced stage in computer evolution. I’m one of those people. When you finish this book, you, too, will probably be one of those people.

I first saw the Macintosh in March of 1983. The Macintosh project was in full swing then; new features and greater capabilities were being added almost daily. There was excitement, and quiet pride, in the air. Activity was non-stop, 24 hours a day. There seemed no more wonderful task than creating Macintosh. Visitors left envious of the talent, comradesie, sense of discovery and—heck, fun—shared by the members of the project.

And there was Macintosh. The first time I saw Macintosh, I laughed with delight; maybe you did, too.

There is more to Macintosh than this book, or any single book, can cover. What we hope to do here is begin at the beginning, with two guys named Jobs and Wozniak, and briefly sketch the develop-
ments that led to Macintosh. We'll then take a tour of Macintosh: the concepts that underlie the machine, the objects that appear on the screen, the commands that do your bidding, and the tools that Apple provides for some creating of your own.

We'll also glance at some aspects of Macintosh that aren't required knowledge: disk drives and files, the MC68000 microprocessor, hardware add-ons (and speculations), and the fascinating and mysterious realm of programming. These chapters are thoroughly optional; read them now, save them for later, or skip them altogether. You may find, however, that it's difficult to own a Macintosh without wanting to know more than Apple, rightly, believes is necessary.

But technicalities are few. More than anything else, the Macintosh is a machine that doesn't require mastery of technicalities. Mac was designed, after all, so you could do things without needing to know exactly how they work.

The Macintosh opens a world of possibilities. Some of the possibilities are computer-related, but many—probably most—are not. Further Reading, starting on page 271, suggests books and periodicals that explore some of those worlds and point, in turn, to others.

Wondrous as it is, the true wonder of Macintosh isn't Macintosh at all. Instead, it's the wonders that you create, with the aid of this marvelous machine.

Do good and be wise. Surprise us.

We can hardly wait.

Doug Clapp
Cupertino, California
1983
1. What Is a Macintosh?
What is Macintosh?
A good question. And a good place to begin.

**What Macintosh Isn't**

First, here's what The Apple Macintosh computer isn't. It isn't a big stodgy computer. That's obvious. It also isn't a little stodgy computer. It isn't a true business computer, although many businesses will buy the Macintosh and be well satisfied. And it isn't a game machine disguised as a home computer (although Mac's capable of marvelous game-playing).

In many ways, Macintosh isn't what you've come to expect from a computer. Mac isn't hard to use, doesn't require a Computer Science degree to operate, and won't force you to do things its way. You don't even need to be a good typist.

Instead, Mac is simply Mac. A small, friendly machine that's easy to use and fun to get to know. (Note: This is called anthropomorphism: giving an inanimate object human qualities. Certainly, machines can't be friendly any more than wine can be bold or frisky. Apple, Inc. frowns on anthropomorphism, and rightly so — too much of it makes one queasy. Still, it is difficult not to do, when speaking of of Macintosh. The Macintosh will inevitably be known as "Mac," despite Apple's [and anyone else's] best efforts to the contrary. This book bows to that reality. Herein, Mac and Macintosh are used interchangeably.)

One important Mac Rule needs to be mentioned. A rule not mentioned in Apple's otherwise excellent manuals: don't let anyone else in your family use Mac for any length of time. Breaking this rule has dire consequences; it usually involves buying one or more additional Macs.

Then everybody will have a Mac.
You've been warned.

**The Question Answered**

In one sense, Mac is the first "hoopless" computer. Unlike other computers, it has no hoops. Therefore, you don't need to jump through hoops to use the computer.

This hoopless design didn't come easy. Many top designers and programmers worked long and hard to design the Macintosh. Much
of that work consisted of identifying and eliminating hoops. Some missing hoops are:

- The need to learn a foreign language (computerese).
- The need to understand how a computer works to work with the computer.
- The need to be a good typist.
- The need to learn a complex Disk Operating System.
- The need to learn (and memorize) a lengthy list of arcane commands.
- The need to learn (and memorize) a new list of commands for each software program purchased.

A perfect example, employed by some computers, is this command:

COPY[/A][/B][d:][path]filename[.ext][/A][/B]
[+ [d:][path]filename[.ext][/A][/B]...]
[d:][path]filename[.ext][/A][/B][/V]

Is that entered correctly? Let's hope the typesetters who put this book together got it right. And, hopefully, you entered it correctly, if you ever needed to use it.

The Macintosh designers took a different approach. They envisaged a computer that was powerful enough for sophisticated applications, yet easy to use. They knew what they wanted, then worked with a joyful vengeance to bring it to life.

Don't be fooled by Mac. Macintosh is not a toy. It is a rugged, full-scale computer. It can run rings around many computers used by business and scientific organizations. Rings without hoops, you might say.

Macintosh can do all the things you'd expect a computer to do. It also does things you might not have believed any machine could do.

One of Macintosh's many marvels is the ability to turn work into play. Nothing is hidden and no passwords are needed. Information is visible and easily ordered, re-ordered, calculated, shuffled, moved, or stored. Time-consuming tasks are accomplished quickly. You'll find you're actually enjoying what you previously called work or study. And because it's easy, you'll find the quality of whatever you're doing will probably improve.

That's a big claim. Another word for it is performance.
Performance

In general, computer performance is measured in terms of flexibility, memory, and speed. The Macintosh has each in abundance.

Flexibility comes from mechanical design and from the programs [or software] that control the machine. Macintosh application tools and internal programs allow you to begin computing right away.

Memory can be divided into three areas. ROM memory contains internal instructions that run Mac. RAM memory, or user memory, is a storage area for application tools or documents that you create. Disk memory, or disk capacity, refers to available space on reusable disks for tools, or documents (or pictures, or songs, or . . .).

The final measure of performance is speed. Speed is useful and often necessary. If your primary use for a computer is what's called number crunching, you'll find that Mac can add, subtract, multiply, divide, shift, or rotate faster than any other computer in its class.

Speed shows up in other areas, as well. A slow Macintosh would make solving problems (or playing games) drudgery. A slow Macintosh wouldn't be much fun. Macintosh is not slow.

Mac's speed, for the most part, is the result of Mac's brain: the Motorola MC68000 microprocessor. Micro because it's small, processor because it processes instructions one after the other. Each instruction tells the microprocessor to perform one specific operation; ADD two numbers, for example.

Because all microprocessors perform only one instruction at a time, speed [and the power of those instructions] is important. The MC68000 combines an extensive instruction set with a fast chip design; literally thousands of instructions are processed each second. The result is a computer that does what you want, now.

---

Build-a-Mac

Without the 68000 and a wealth of other tiny chips, Mac would have a vastly different appearance.

Remember tubes? Televisions and stereos, in the old days, were filled with vacuum tubes. They burned out a lot, too, it seems.

Imagine 68,000 of those tubes. You're about to use those tubes to make a computer. First, imagine a circuit board big enough to hold 68,000 tubes. That's how much room would be taken by the 68000 microprocessor, not too many years ago.

But since we're imagining an entire computer, not just one chip, you'd better throw in a few more tubes. Say, maybe 4,000. All those
tubes will certainly take some power to run, so you'd better add a power supply. Make it the size of a refrigerator.

We're almost done. Now we need memory. Add about 224,000 more tubes. Better make that power supply a bit bigger.

Now we need tape storage, so we can save our work. Add a tape storage unit. That's about the size of another refrigerator. Better bump up the power supply again.

What did we forget? Oh yes, the display. Throw in (carefully) a television. Then, for input, throw in a typewriter keyboard.

Congratulations. You've just configured the mechanical parts of a Macintosh computer. Stand back and take a look.

Now we need to write a few volumes of software to control the machine. For the Macintosh, better allow approximately 200 person-years of effort in software development. That figure assumes, of course, that your programmers are as good as those that programmed Macintosh. Quite an assumption.

All that's left now is deciding where to put your newly designed computer. And determining how to pay your electricity bill.

Fortunately, Apple designed the Macintosh somewhat differently than we did.

The point of this foolishness is not to be fooled. Beneath the fetching exterior, behind the often whimsical display, Macintosh is a powerful computer. It's not a toy, unless you consider Lear Jets, speed boats, fine sports cars, and space shuttles to be toys.

In that case, it is a toy.
2. A Short History of Apple and Mac's Creation
To understand Macintosh, we must begin at the beginning. In the beginning, (this beginning, anyway), there were computers. Big computers. Big, expensive computers.

As technology advanced, computer parts become smaller, as we just illustrated. Oddly enough, though, the big computer manufacturers didn’t think small computers were a big idea. Computers, after all, were mammoth machines behind glass windows that could only by cajoled by the data processing priesthood.

The large corporations thought small computers were a silly idea, if they thought about small computers at all. So they didn’t make any small computers.

Hobbyists, however, thought differently. In the basements of America, they were busy soldering circuit boards, inserting chips, stringing wires, and trying to make their contraptions “listen” to keyboards and “talk” to television screens.

In those days, hobby computers had rows of toggle switches on the front panel. The switches were used to program the computer by entering a precise series of ones and zeros. Only a true hobbyist could withstand such a grueling task.

---

**Apple’s Beginnings**

About this time, the mists of history recede a bit. Enter Steve Jobs and Steve Wozniak, one hobbyist/visionary and one hobbyist/engineer. Jobs, at the time, was one of the first ten programmers hired by Atari to program video arcade machines. Wozniak was an technician employed by Hewlett-Packard, a large computer manufacturer. Wozniak helped design calculators for HP, but found calculator design boring, at least in comparison to designing computers. He asked for a transfer to HP’s Research and Development division, but was turned down; after all, he was only a degreeless technician, not a Computer Engineer.

Scorned, Wozniak designed a computer anyway, putting in four months of almost non-stop after-hours work. The result was a motherboard: the complete circuitry of a computer, less display, drives, and keyboard.

Wozniak brought the computer to his supervisors. Did Hewlett-Packard want to sell it? No. They did, however, grant Wozniak a legal release for his design.

Wozniak proudly showed his creation to fellow members of the now legendary Stanford Homebrew Computer Club, many of whose
members in the early seventies went on to create Silicon Valley's high-flying, hi-tech companies. Another of the club's members was Steve Jobs, who convinced Wozniak to form a business and market a computer based on Wozniak's design.

Wozniak and Jobs next met with Paul Terrell, who had started a chain of hobbyist computer stores. Terrell agreed to buy 50 of the circuit boards for $549 each, provided they could deliver the boards in a month. The retail price of these Apple I's was to be about $650.

Making computer boards meant spending money for parts. Jobs happened to own a VW microbus (fittingly). Wozniak owned a scientific desktop calculator (appropriately). The bus and the calculator were sold and pawned (respectively).

Jobs and Wozniak next paid a visit to a large computer parts distributor. They presented their list of required parts and were told that the terms were "net 30 days."

Jobs and Wozniak didn't know what "net 30 days" meant, but they did know it meant they didn't have to pay immediately. So they took the parts.

"30 days net" means, of course, that the entire balance is due in 30 days. It was a hefty balance.

Few computers were ever assembled as quickly as those that next flowed from Job's garage. By month's end, 20 boards were completed and delivered. And they worked.

Jobs, from the beginning, saw beyond the small-scale hobbyist operation. With help from Wozniak's father, himself an engineer, he persuaded Wozniak to integrate the computer into a case, complete with keyboard (a unique idea, in those days), for sale as a consumer item (another unique thought).

Sales boomed, talented people and investment capital both arrived at the proper times, Wozniak designed other marvels, Jobs made canny business decisions that belied his youth, and Apple Computer was born and prospered.

A happy ending.

A happy middle, actually. Because Jobs and Wozniak weren't the only would-be small computer manufacturers of those early days of microcomputing (in the late 1970s). Others tried, but most failed. Some, but not many, of the others are still around today.

**Explaining Success**

Apple's success can be explained in many ways, timing not the least of them. But the crucial ingredient was probably this: as Jobs envi-
sioned from the start, Apples have always been created for use by ordinary people.

Not dull, simple-minded, or illiterate people, but reasonably intelligent adults [and children] who aren't necessarily fascinated by Computer Science. People who want computers for what they can do, not merely for what they are.

People who aren't thrilled by toggle switches.

That's a lot of people. By the time you read these words, over one million people will own Apples. Apple Computer Inc. is now a member of the Fortune 500, the only company to make the 500-largest list within 5 years of incorporation. Apple's gross revenues are now about 1 billion (with a b) dollars a year. And growing. Fast.

Good products alone don't insure success, however. Apple Computer also made an early commitment to support both Apple dealers and Apple customers. The result is a national network of knowledgeable and reputable dealers that not only sell, but also service Apples, should the need arise. And lots of Apple customers who feel well satisfied with "their" computer company.

Apple also made a strong commitment to Independent Software Vendors; non-Apple companies who create and market software programs for Apple computers. That close cooperation has led to a staggering number of programs for Apples. A well-known advertisement (maybe you've seen it) uses two magazine pages to ask: "Will somebody please tell me what an Apple can do?" then goes on to fill both pages with most (but not all) of the programs available for the Apple. The print is very small. In fact, there are over 15,000 software programs available for Apple computers.

If anything, Apple was even more supportive of outside software developers during creation of the Macintosh. If 15,000 programs are now available for the Apple II (and IIe and III), think what will soon be available for the Macintosh.

Certainly, there are other computers, some of them fine machines. Some are more expensive, some less expensive. By now, many of the major computer manufacturers are marketing computers called Personal, Advanced Personal, Personal Business, Professional, or simply Home Computer.

It's easy to forget that Apple Computer, Inc. now is one of the major computer manufacturers, with approximately 4,500 employees world-wide. Apple computers are manufactured in California, Texas, Singapore, and Cork, Ireland. Apple has corporate offices in Tokyo, London, Paris, Milan, Munich, Sidney, Toronto, and other spots around the globe, including a few places in Cupertino, California. By the end of the decade, international sales of Apples may account for 50% of corporate revenues.
Somehow, success was achieved without creating an unresponsive, elitist, corporate bureaucracy—one of the reasons Apple consistently attracts some of the best corporate and engineering talent available.

Which means that other corporations will be playing catch-up for some time to come.

Which almost brings us to the Apple Macintosh Computer. But first let’s talk about money.

**Price and Performance**

That mammoth fictional computer we designed in Chapter 1 could actually have been marketed 30 years ago as a home computer. But it wasn’t. It would have cost hundreds of thousand of dollars, and nobody would have bought one, even if they had a place to put it.
One of the Macintosh's most amazing features is its price. Amazing because that price reflects:

- The high cost of new and powerful chips, like the MC68000.
- The considerable cost of parts such as advanced, high-capacity disk drives, displays, and power supplies.
- Research and development—the cost of many "person-years" of work creating the Macintosh hardware, software programs, and operating systems. (If it's easy to use, you can bet it took a long time to develop and program.)
- All other costs.

About 10 years ago, a computer with some of Mac's power and features actually existed. It was the Xerox Alto computer. It cost roughly $32,000. A few years later, Xerox announced another computer with some Mac-like features: the Xerox Star. It cost $16,000.

As you might expect, neither machine caught the public fancy. It took talent and hard work to create the Macintosh. It also took talent and hard work to make it affordable.

From Whence Mac Sprang

The Xerox Alto and Xerox Star were Mac's ancestors, but the story begins somewhat earlier. A good place to begin is with Alan Kay, a man who was, and still is, a brilliant computer scientist.

In the early seventies, Kay founded the Learning Research Group at the Xerox Palo Alto Research Center. The LRG at Xerox PARC, as it was called, conducted research in a number of areas aimed at making computers more powerful and easier to use. Much of the group's work was fueled by Alan Kay's vision of personal computing: the Dynabook.

The Dynabook was a simple but stunning idea: a computer with the power of a mainframe computer contained in a portable unit the size of a notebook. The Dynabook would have a flat screen, both visual and audio communications capabilities, and be able to tap into larger computing and information networks.

In one form or another, scientists have been trying to realize the Dynabook ever since.
The Demise of Modes

One piece of the Dynabook puzzle was the Preemption Dilemma. This dilemma is familiar to everyone who works with computers, although most people don’t know that the condition has an academic-sounding title. It means simply that computers "trap" you into doing things "their" way; and heaven help you if you’re not sure what that way is.

The traps are called modes. With most computers, you’re always in one mode or another; some things can be done in one place, other things can’t. To get from, say, Edit mode to System mode, you need to know the right commands. If you forget what those commands are, or use them incorrectly, you won’t get where you want to go. Instead, you may get somewhere you greatly don’t want to go.

Much of the misery inflicted on computer users over the years comes from these omnipresent modes. The dilemma of the Preemption Dilemma is that choices you might desire are denied, or Preempted by the computer. “Do it my way, or else!” seems to be the message.

About this time, most people have a few messages of their own for the computer.

So the LRG of the PARC set about the task of eliminating modes. It wasn’t easy. When you eliminate modes, you have to change quite a few other things as well.

The standard User Interface was the next target for demolition.
The User is you. The Interface is everything between you, and whatever you want the computer to do. Generally, it means how information is displayed on the screen, and how information is entered from the keyboard or other devices.

If modes are bad, most user interfaces are worse. It’s still possible, this very day, to visit your local computer store, plunk down four or five thousand dollars and take home a computer that, when turned on, greets you with the marvelously expressive symbols:

A>

on an otherwise blank screen.

Not, you’ll agree, a swell user interface.

The solution to all this was the concept of Windowing; a solution that made what was in the computer visible to users outside the computer.

The hammer-stroke of inspiration was that windows shouldn’t just appear on the screen; they should overlap, like sheets of paper. If you can shuffle papers, you should be able to shuffle windows; and
each window could contain information entirely different from information in other windows. Each window could hold a different document, and a different tool to work with each document. If you don't like where you are, leave.

No traps.

That's a pretty good basic definition of Macintosh.

Implementing the ideas was difficult. It's easier to merely want something than actually to do it. Often, some of the best ideas are impractical—the technology isn't perfected, or affordable, or even yet invented.

A> isn't friendly, but it is a cheap and simple way to design a user interface.

Mac's Mom

Xerox never produced a machine that both typified these ideas and was even moderately affordable. But Apple did.

Lisa was first out of the blocks. After fifty million dollars of research and development costs, and months of speculation, Apple announced the Lisa in January, 1983.

Lisa is another of Mac's ancestors; in this case, Mac's Dad (well, maybe "Mom" would be the proper metaphor). Lisa incorporates the hardware and software advances just described; windows, modeless operation, and more. Optionally, it features a number of integrated programs ideal for business use and other demanding applications.
Lisa costs around $5,000. Is it worth it? Is a Porsche worth the money? How about a high-quality copier machine? Lisa does offer a tremendous array of features for the money, but $5,000 is still $5,000, right?

Macintosh is not merely Lisa writ small, however. In many ways, Macintosh is Lisa writ affordable.

The Wheel Turns

Now history repeats itself. Xerox is a big computer manufacturer. As we’ve seen, they had (and have) some top computer people on the payroll.

But Apple introduced the Macintosh, not Xerox, or IBM, or any of the other major computer companies. And once again, the visionary Steve Jobs was the instigator.

Making Macintosh affordable required many elements and precise timing. A fast and powerful processor, such as the 68000, was needed. A new generation of high-capacity memory chips were needed to save space and reduce power consumption. A small high-capacity disk drive was needed for storage. Other sophisticated chips, parts, and gizmos were needed, as was a small, lightweight, dependable power supply.

If fancy parts were all it took, we’d be inundated with Mac-like machines. To make Macintosh small, fast, and affordable also took a special hardware design. You may never understand the electronic details of Macintosh (and you’ll never need to understand them), but be assured that Macintosh embodies a magical hardware design.

Even that wasn’t enough. Without internal programming, the Mac would be only an attractive exterior on a pile of potential.

In many respects, software systems are machines as real as any machine we can see and touch. They’re designed, built, and tested with the same extraordinary care and patience given to machines made of metal and silicon.

Here’s where Apple one-upped even Xerox: software. The software within Lisa and Macintosh is based on something called QuickDraw, something that no one else has.

The People

Creating the two machines that together comprise Mac required skills possessed by only a handful of people.
Assigning credit is difficult. Many people worked long hours to create Macintosh. Their names appear in this book on page 237.

Where did they come from? They came from other corporations (including Xerox), from Apple, from Major Universities, and small universities, and small business, and other typical and not-so-typical places.

If you ever chance to meet them, they’ll say that they’re proud of the Macintosh, and proud to work for a company that never holds them back, or expects less than the best they can produce.

They’ll also tell you that what they accomplished “Wasn’t all that difficult.”

Don’t believe them.
3.
Simple Concepts
The Desktop

So far, we have the simple and powerful concepts of windows and modeless operation. Simple concepts, but difficult to implement. Difficult for a number of reasons: what about programs that need to be loaded from disk into the computer? How's that done? What about programs that are necessary for use with other, user-created, files: word processing programs, for example, that manipulate text? And what about likely, or potential, errors on the part of users? How are they handled? And what about selections? What can be done where, and how, and what can and can't be allowed to happen?

And on and on and on.

It seemed that a few more conceptual models were needed. The first was the Desk.

Walnut or Cherry?

Where are papers (read windows) shuffled? Why, on a desktop, of course! A fundamental concept. Everything is done on a conceptual desktop. The display is, and will always be, no matter what is being done, the visual equivalent of a desktop. Anything that can be done on a desktop (within the bounds of good taste, of course) can conceivably be done on the Macintosh desktop. Including drawing and painting, which can get you in trouble if done on traditional desktops.
Everything seen on the Macintosh display is on top of the desktop; nothing can ever be under or below the desktop; the desktop is always on the bottom. Often it will be completely covered—but, like a traditional desktop, if you dig down far enough, there it is.

And nothing ever rolls off the desk and into some hard-to-reach place.

The transition to Macintosh will be eased if you consciously think of the display as a desktop—an object, just like an old-fashioned desktop. The desktop is where all work is done, and appears automatically whenever Macintosh is switched-on.

In more general form, this **objective** thinking deserves an illustration all to itself. Here it comes:

**Everything**

is displayed

by Macintosh

when possible

as an **object**.

What's everything? Everything: the contents of a disk, files, programs, all documents (regardless of type), error messages, buttons, dials, clocks, everything and anything that shows up on the Macintosh desktop is presented as an object.

Some of the objects look much like their everyday counterparts (such as the Calculator Desk Accessory), and other objects may look like nothing you've seen before. All objects, however, are shown in a manner that at least *hints* at their intended purpose.

---

**Windows**

In the last chapter we glanced at (through?) windows, another primary concept. In the Macintosh, almost everything on the desktop, in some manner or another, is in a window. The only real exceptions are icons, which we'll investigate in Chapter 5.

Although you may already know about *Alert Boxes* and *Dialog Boxes* and *Menus*, it may not have occurred to you that these, too,
are windows. They are. They may not have some, or all, of the controls associated with many windows, but windows they are.

If objects are a way to represent things (such as clocks, calculators, and the like), then windows are a way to open up the objects and look inside, to information displayed.

Why windows? Because windows offer the easiest, and most elegant way yet discovered, to present information for manipulation. And that's the purpose of Macintosh: to allow you to manipulate information quickly and conveniently.

The information most often seen in windows comes in the form expressed by the next concept.

**Documents**

The term *document* has a needlessly stuffy connotation; actually, a document is any collection of information—and not necessarily in the form of numbers or words; pictures, doodles, and songs are all perfectly good documents.

A letter is a document. So is a schedule, a list, a sketch, a payroll record, a BASIC program, a financial statement, or your Last Will and Testament. Each, hopefully, will be used in a different manner, and for different purposes, but they are all documents, and will all appear within windows.

There are three types of Macintosh documents:

**Graphic documents**: Anything created with MacPaint, or similar programs, is a graphic document.

**Text documents**: Anything created with MacWrite, or other word processors, is a text document. Text documents include anything from notes to novels. You can put information from other documents into text documents (we'll get into that later), but anything originally created as a text document will remain, essentially, a text document.

**Cell documents**: The last type of document are those that, generally, deal with numerical information. A cell document is the most structured of the three types. The word *cell* refers to areas where information, usually numbers, are entered, manipulated, or otherwise massaged to perfection.

The electronic spreadsheet, or worksheet, is the classic cell document. Information is entered on a grid and the numbers shown are the results of formulas. Changing the formula changes some, or all, of the numbers on a grid. With Macintosh, the grid may be visible, invisible, or, (typically) whichever you prefer.
The three types of documents can be defined in another, more abstract way, which also makes a good deal of sense.

Graphic documents are free-form documents. Totally unstructured, as using MacPaint demonstrates. Free-form documents are the least choosy of all document types. Blank areas of the display, regions of idealized spray paint, or precise circles and squares, it makes no difference to free-form documents. Free-form documents have few rules to remember.

Text documents can be thought of as a strings of characters, like beads on a string. Luckily, the string can be broken, which happens whenever anything is inserted into text documents. Although text may appear as sentences and paragraphs when we work with groups of words, the string-like nature of text is seen whenever text is scrolled. In reality, we're either pointing, and moving, forward through the text, or pointing, and moving, backward through the text. Along the string, if you will.

Not only text characters are beaded on the string. Whenever a tab is set into text, for example, a code that says "here's a tab marker" is beaded onto the string, whether the actual tab marker is visible on-screen, or not.

The string concept also helps when we consider cutting or deleting text. The act of selecting text tells MacWrite the beginning and end locations to snip off the text. After all, you can't cut without snipping, right?
Other documents share a string-like nature, even those that aren’t, primarily, concerned with word processing. Many documents, those used in filing programs, for example, are often best thought of as strings of words.

The string concept is a powerful way to represent information within a computer. Entire languages (LISP among them) have been built around this basic idea. If you learn Macintosh BASIC, you’ll soon find that strings of characters can be added, searched, sorted and shuffled in a number of clever and useful ways.

Cell documents, again, are the most structured of the three types of documents. Within the computer, the grid-like nature of cell documents is quite rigid, even if we, on the outside, never see the actual grid. Documents composed of rows and columns are typically cell documents. In these, information can be entered within the cells, and cells may be changed and calculated in a number of ways, but the cells themselves, visible or invisible, are there for the duration, which, if you think about it, simplifies things considerably.

Enough about structure. How about mix-and-match? A few cells, a little text, maybe a spot of graphics?

Sure, within reason. But you’ll have to wait until Chapter 9 to learn the precise cans and cannots (and there are a few—the ideas given above may give you a better feel for the cannots when they appear, and they will).

Well then, you may be thinking, if a text document is something created by MacWrite, then what is MacWrite?

### Tools

This is the last heavy concept we need, for the moment anyway.

Simply put, programs, or tools, are what create documents. Some
of the less advanced computers on the market have something similar, called application software. They are often clumsy to use and typically hard to understand and command.

(Note to everyone: from now on, the term tool is a synonym for application program. The word program will still be thrown in, once in a while, so as not to totally baffle experienced computerists. But on the Macintosh, the word tool makes a great deal more sense than does program. It really does. Trust me.)

The Macintosh tool concept simplifies things. The primary simplification is this:

Documents know what tools they need.

An important point, worth repeating:

Documents know what tools they need.

Tools can arrive in two ways. Double-clicking an icon that represents an application program opens a window governed by the application program. Both tools and documents are shown on the Macintosh display as icons: visual representations that suggest the type of tool or document lurking within. To create a new text document, double-click the MacWrite icon. Up pops an empty MacWrite window, ready for action, menus blazing in the sun.

The second way is the cleverest. Merely open any icon that represents a document. Any document: graphic, text, or cell; you needn’t worry about which is which. If the icon clicked is Letter to Mayor, up pops the proper Letter to Mayor window and MacWrite tags along—automatically!

That’s one of the most striking innovations of Macintosh. You never need to first load an software application program, then load a file that you wish to work with. With Macintosh, tools always come along with documents that require their services. In practice, the program used to create the original document is the program that tags along.

When you buy software for Macintosh, you may purchase programs, documents, or both. Good software will behave just as Apple software behaves: if you start from scratch, you first select the icon that represents the program you wish to work with. If you work on a document, you select the document and it gets the proper program.

**Principal Tools**

Now that the waters are clear, let’s muddy them.

Say that you’ve just completed writing your autobiography. The
document titled *Stirring Autobiography* was created with MacWrite. Thus, MacWrite is the tool associated with the document.

Now, however, you have some other nifty tools you'd like to use on the autobiographical document. Possibly an electronic spelling checker or grammar checker. Can it be done? Will MacWrite disallow it? Is it going to be confusing?

No problem. MacWrite, in this instance is called the *Principal tool*. But documents can be understood by more than one tool. The spelling and grammar checkers are *Secondary tools*. They won't tag along unless asked, but can be easily used.

Documents have only one principal tool, but may have any number of secondary tools that allow further document manipulations. Some potential secondary tools for text documents might include:

- Spelling checkers
- Grammar checkers
- Indexers
- Word counters
- Thesaurus
- Word frequency of use counters
- Reading grade-level analysis
- Customized form letter generators
- Foreign language translators

We probably won't see that last secondary tool for a while, though. Tools come in all sizes and shapes. For the most part, anything that creates a window, and displays a Menu Bar is a tool. That includes tools for writing, filing, drawing, analysis, charting, graphing, programming, communications, budgeting, and game-playing. All tools.

You may have noticed that the objects have been getting less concrete all the time. A tool is a fairly abstract concept. There is one more concept that's even more abstract: files.

**Files**

Knowledge about files isn't necessary to use Macintosh, but it may clarify the above comments.

Files are where tools and documents come from. They typically reside on disk; globs of magnetic information without a care in the world. There are three types of disk files (remember, you don't really need this information):

*Document files*, created by users, like yourself, and either graphic, textual, or cellular in nature.
Tool files, created by programmers, stored in a fashion that only computers and programmers can appreciate, and...

Resource files, also created by programmers and, in some ways, similar to tool files. Resource files are of no concern to users, but provide programmers with methods or data to work on tools or certain characteristics of the Macintosh, or to add nifty features to other tools. A resource file containing a special type font, for instance, could be used by a tool file. The resource file would tag along with the tool file much like tool files tag along with documents. In this case, however, we wouldn't be aware the resource file even exists! And why should we? The result, we can hope, is more and better tools for Macintosh.

Regardless of their type, all files have certain attributes: names, descriptions of size and content, and the date of creation and last modification.

In Four Words

The important concepts are few, but far reaching: Desktop, Windows, Documents, and Tools. A one-sentence summation is in order. Here goes:

Documents are displayed in Windows, manipulated by Tools, and found on Desktops.

What could be simpler than that?
4. Expressing Your Wishes
The Mouse

The mouse is one of the most novel aspects of Macintosh. Maybe you’ve never used a computer mouse. Maybe you’re not sure that using a mouse will be easy, or that mice were even a good idea in the first place.

Read on. It was a good idea. It’s easy.

The Macintosh mouse was chosen as a pointing device after Apple investigated every other conceivable way to point: cursor keys, trackballs, light pens, graphic tablets, joysticks, you name it. All had drawbacks.

The mouse has many advantages: it’s light, easy to use, inexpensive, resolves to an accuracy of 200 screen points per square inch, and retains its position when not in use. Best of all, it somehow feels right after only a few minutes of use.

Bill Atkinson, one of the Macintosh designers, believes that one benefit of using a mouse is a “sense of quiet interaction” with the machine. He’s right; using a mouse enables you to do things without first having to determine how to do them. The mind thinks, the hand moves. Quietly. No knitted brows, no anxiety about “what do I have to do next?”
Initially, some people are afraid to first use the mouse. No kidding. The fear probably comes from a belief that moving the mouse will "cause something to happen."

That makes sense, but it's not true. It is true that rolling the mouse will cause the pointer on the Macintosh screen to move, in perfect sympathy with the movements of the mouse. But rolling the mouse—and we're not yet talking about clicking or pressing that button—will never cause anything to happen (except, of course, moving the pointer).

A software digression: the Macintosh user interface is designed around the concept of events. An event is defined as:

- a press of a keyboard key,
- a click of the mouse button,
- a double-click of the mouse button,
- a shift-click, or
- pressing and holding the mouse button down, usually while moving the mouse.

Merely moving the pointer leaves Macintosh totally undisturbed. Nor is there any need to move the mouse cautiously. It's impossible to:

- roll the pointer off the desktop, or
- to get ahead of the on-screen pointer, thereby confusing or destroying the machine.

Once the pointer is rolled to the edge of the screen, it stops. It can still be moved up or down (or left and right if you're trying to move it off the top or bottom of the screen), but it can't be entirely removed from the screen.

**Pointer Shapes**

Like all Macintosh objects, the pointer conveys whatever helpful information it can. It does this by changing appearance.

Typically, the pointer is shaped like an arrow, and points north-northwest (sorry, we're not taking questions at the moment). In cer-
tain documents, the pointer takes on an appearance that, hopefully, conveys useful information:

- In cell or numerical documents, the pointer will often become a hollow cross.
- In some graphic documents, the pointer will be a non-hollow cross, much like a cross-hair in a rifle scope.
- In text documents, the pointer appears as a blinking vertical line, called an Insertion Bar. The Insertion Bar marks the position where text will next be entered.

In most remaining cases, the pointer will be an arrow-shaped pointer. Notice the words "often", and "in some graphic documents...".
The lack of specificity is purposeful. Pointer (or cursor) shapes are left up to the best thinking of the software designer. There's no law that says:

"Pointers must be Hollow Crosses when in Cell documents."

It would be nice if they were hollow crosses, but it's too soon to tell if programmers will adhere rigorously to the implicit suggestions Apple Inc. has made about pointer shapes. [In truth, much time was spent at Apple pondering matters such as pointer shapes. The original shape for "Wait a second, operation in progress" (as in: loading a program from disk) was an hourglass shape. After much deliberation, the hourglass became a wristwatch; the old-fashioned wristwatch with hands. This was thought to be more psychologically pleasing, and to make the passage of time seem somewhat shorter.] The diversity of pointer shapes is a wonderful feature of Macintosh. As we stated earlier, information is presented in the form of objects; pointer shapes are just one more object capable of conveying information.

In a graphic application, the pointer may become a paint sprayer, a pencil, an eraser, or a paint brush. In a table of numbers, the pointer may become a fist with one out-stretched finger, or a left-pointing arrow. In a game, the pointer may become a running man, a dancing girl, an invader from space, or a hapless earthling.

It's all up to the imagination of the software designer.

---

### Mouse Maneuvers

Mice are becoming a popular input device. It's possible to buy one-button, two-button, or even three-button mice. Apple took the one-button approach for the simplest of reasons: it's simpler.

Mouse commands come from these basic actions:

- **Clicking**—quick tap and release of the mouse button,
- **Double-clicking**—fast double-click of the mouse button,
- **Shift-clicking**—holding down the keyboard shift key while clicking, and
- **Dragging**—pressing and holding the mouse button down while moving the mouse.

Clicking, the most fundamental operation, always selects; often, what you've selected will be instantly highlighted in inverse to confirm the selection.
Clicking chooses something to work with, usually an icon, but often a single item, number, or position within text. Point at it, then click.

Double-clicking is a shortcut, a fast way to both select and perform an action. The response to Double-clicking is always a superset, an enhancement, or an advanced feature of single-clicking the same object. Some single and Double-clicking examples are:

- Single-clicking selects an icon; Double-clicking both selects and opens a window into the document.
- In MacPaint, single-clicking over the Eraser selects the eraser as the current tool. Double-clicking erases the entire painting screen (in effect, both choosing the eraser then using it to its fullest).
- Also in MacPaint, clicking over a pattern chooses the pattern as the current "paint." Double-clicking over a pattern gives you an enlargement of the pattern, and allows you to redesign the pattern to your own liking. In this case, Double-clicking is a way into an advanced feature, not merely a enhanced version of the same command.

Double-clicking is never absolutely necessary. Any action initiated by Double-clicking should also be found on a menu. To take the first example, after single-clicking the icon, the File menu may be opened and the command Open selected.

Most Macintosh users Double-click with abandon, and seldom open icons via the File menu. It's, well...silly to go to all that much extra work.

Remember that Double-clicking is clicking twice in rapid succession—not two, rather spaced-out clicks. To be precise, the second click must come no later than 700 milliseconds after the first unless altered by the user, which we'll explain in the next chapter. Don't get nervous; it just means that a Double-click is clickclick, rather than click..........click.

Shift-clicking is another short-cut. When the second click is accompanied by pressing either Shift key, the item previously selected stays selected when you move to, and Shift-click, something else. A handy way to select three different icons in an icon-filled window.

Shift-clicking is most often used in text, where the first click marks the beginning of a lengthy selection [three or more windowfuls of text, say], and the second click, accompanied by pressing the Shift key, marks the end of the selection. The text between the two clicked positions is immediately selected and inverted.
Pressing, for the moment, is confined to scrolling a large document through a window: the pointer is positioned over a scroll arrow and the mouse button is held down. This initiates a smooth scroll that stops when the button is released. The arrow may also be clicked to scroll one line at a time.

**Point-Press-Drag-Release**

Dragging deserves a section all to itself. This is it.

Dragging is an inclusive term for a series of actions: pointing, depressing the mouse button, moving the mouse, then releasing the mouse button.

If it were as hard to do as explain, Apple would have never allowed it.

A user's first acquaintance with dragging comes with the discovery that those little icons can be moved around, provided that you keep the button down while moving the mouse. Point-press-drag-release.

The next discovery has to do with the Menu Bars at the top of the screen: point at a File menu, press the button, and the menu pulls down. Continue holding the button while dragging down through the menu. Release the button when the item you want is inverted. Point-press-drag-release.

The next higher conceptual level of dragging is this: dragging is selecting, much like select-by-clicking, but allows us to select a number of items, or a large area, with one motion.

Examples, examples:

![Screen 4.1 Icons within Selection Rectangle](image)

- Pick a spot on the desktop, at a time when a few icons are lurking nearby. Drag across the icons. Notice the shimmering rectangle? That's the *Selection Rectangle*. Keep dragging until two or more icons are fully snared by the rectangle, then release
the button. The icons instantly invert, an admission of selection. You're now free to plunder the Menu Bar, doing your will on all the icons at once. Maybe you'd like to turn them all into windows, by selecting Open from the File menu. Or Get Info on the whole bunch? Go for it.

Screen 4.2 Selected Text in MacWrite

- In MacWrite, dragging again selects; anything from a single character to an entire document. The text inverts as the pointer is dragged over it; when the button is released, the menus can be accessed to change the selected text (for example, cut, copy, or change font), or the Backspace key can be hit, which sends the text to alphabet heaven, a place not known for clipboards. (Note: "Undo Typing" may get your text back if deleted in this manner, but don't count on it.)

- In MacPaint, dragging is how everything is painted: pick a tool, pick a pattern, then drag away.

Screen 4.3 Cells Selected By Dragging In Microsoft Multiplan

- In spreadsheets, such as Microsoft's Multiplan, dragging across the spreadsheet selects one or more cells for manipulation; pos-
sibly moving a row or column, or changing the format of displayed numbers.

Another good use for dragging is scanning the menus. Place the pointer over the Menu Bar, press the button, then move across the menu titles. Each menu will obediently pull down for your perusal—a good way to check out the commands of a new program. When no one's looking, drag across the menu bar really, really fast. See how quickly those menus can pull down. It's okay, you can't hurt Macintosh by moving the pointer, or dragging too fast over the screen.

This sort of fooling around is also testimony to the amazing graphic speed of the Macintosh software. [The hardware's not too shabby either: the 68000 processor can do over one million operations a second. How many can you do?]

---

The Keyboard

It's small, it's light, it doesn't have a lot of keys: the Macintosh keyboard.

If you're acquainted with computer keyboards, you know that they come in all shapes, sizes, and with any number of extra function keys, or computerese keys, such as Ctrl, or Ins, or Esc.

The addition of extra keys on the Macintosh keyboard was deliberately held to a minimum. In practice, the keyboard is used only for entering text or numbers; it's possible to do an enormous amount of work with Macintosh and seldom use the keyboard.
The keyboard comes in two versions: American and European. We'll confine our remarks to the American version, after first noting that the European version

- conforms to the ISO standard, and
- has 50 keys (one more than the American version).

For the most part, the American keyboard layout is Standard American typewriter: a Tab key, two large Shift keys, and a Return key (similar to a typewriter carriage return), all in their proper positions. If you're a reasonably handy typist, you'll find no trouble adjusting to the Macintosh keyboard.

**Character Keys**

The alphabetic, numeric, and punctuation/symbol keys behave, again, as on typewriters: holding down either shift key gives you either a capitalized letter, or the upper punctuation mark/symbol on non-alphabetic keys.

So much for typewriters.

The Caps Lock key is a toggle: press it down and it stays down. To release Caps Lock, press again. When the Caps Lock key is depressed, letters appear [not surprisingly] capitalized [exactly as if the Shift key were held down].

Don't fall asleep yet. Caps Lock has no effect on any of the non-alphabetic keys. In other words, pressing Caps Lock down, then hitting the 8 key will display 8—not the symbol above 8 [the well-known: *], as you might expect.

The reason is simple: people often want to type in all caps, but seldom want to type in all symbols: ! @ # $ % ^ & * ( ).

So far, we have two interpretations for each standard key: lower and upper. Holding down either Option key produces a third symbol for each of the keys.

And a strange collection it is. The Apple marketing people, we can assume, met long into the night pondering the proper arcane symbols to give consumers as Options.

The results can be seen in Appendix D: Standard Key Character Formats (snappy title for an appendix) and include, roughly speaking, letters from far-off lands, Greek symbols, mathematical symbols, the ever-popular Copyright and Trademark symbols, and a few just flat-out odd-looking squiggles.

Marketing, we can assume, hopes that everyone is now satisfied. Moving along, we come to the Command key. The Command key
If we regard the survival of even the smallest tiny bit of something or another, isn't it at all conceivable that something else is possible somewhat the same? Or, isn't that right? Maybe it isn't, but then, who can say for sure? I certainly can't. Can you? I didn't think so.

Moving along, we must also consider the possibility

Screen 4.4 Pull-down Menu Showing Command Key Options

is found to the immediate right of the leftmost Option key. It's the one marked with the strange symbol: ⌘

What is this symbol? No one knows. Rumor has it that, long ago, when Macintosh was young and unreleased, the key read Command. Other rumors say that the key was marked with an Apple symbol. Both rumors may be correct.

Today, it's merely a strange symbol. Obviously, this is the universal symbol for Command.

What's it for? Giving commands: the Command key replaces use of the mouse for many frequently given commands.

In many pull-down menus, the menu item will be followed by the Command symbol and a single letter. Holding down the Command key while pressing the letter will duplicate the act of selecting the item by mouse. Command-symbol C will accomplish the same action as the mouse selecting Cut from a MacWrite menu, for example.

This is a have your cake and eat it too maneuver on Apple's part; fast typists, provided they remember the commands, aren't penalized by needing to remove their fingers from the keyboard. Use of the Command key also insures that everything may be done from the keyboard, if programmers are on their toes.

It will be interesting to see how much use is made of the Command key. Most users will begin selecting menu items by mouse, and are likely to remain fixed in their ways. Using the Command
key, although faster, requires a good memory (because the pull-down menu isn't visible).

For repetitive use, it's probably worth the minimal effort to deliberately use the Command key. So what if it's not fun: it's fast.

**Special Keys: Tab, Enter, and Return**

Tab is similar to a typewriter Tab key: it moves the insertion point to the next tab stop or field. Also, some programs may let you Tab vertically. In most cases, Tab will be used in word-processing applications, and you'll get the opportunity to first set your Tab stops before tabbing.

The Return and Enter keys are confusingly similar. Both keys have their genesis in the days of clunky teletype printers; computer screen displays. In those days, Enter meant line feed: move down one line and continue printing. Return meant hard carriage return: end this line, move down one line, and start printing at the leftmost position on the new line.

Both Line Feed and Return had, and have, respective hexadecimal codes (Return is represented by $0D, one of the first hex codes learned by budding assembly language programmers).

Which key is which, and what does what, and what do you want? These key questions are still causing confusion, especially in the world of modems and bulletin boards, where no one, it seems, is sending, or not sending, the proper carriage return and/or line feed codes across the telephone lines.

Many computer keyboards have now abandoned the Enter/Line Feed key and use Return to signal both Carriage Return and Line Feed. Apple also makes Return serve as both Carriage Return and Line Feed. So what does Enter represent? Enter represents End of Text, another control-code throwback to teletype days. It's difficult to say precisely what each key does, because their actions will be determined by the software program running at any given moment, but in general:

- Return both confirms and enters what you're just typed. File-name? Type: Some Silly File, then hit Return. Return means: Do It.

- Enter also means Do It.

Both keys offer another mouseless short-cut: pressing either duplicates clicking "Okay" in a Dialog Box.
In Real Life, it's likely that Return will suffice for all necessary actions, and Enter will be seldom, if ever, used. But as you buy new software for Macintosh, you might want to scan the manual's index for the word “Enter” or “Enter key,” just in case some courageous programmer has decided actually to use the Enter key.

Stranger things have happened.

**Backspacing**

Technically called a *destructive backspace*, hitting the Backspace key moves the insertion point left one position and erases any character in that position.

The Backspace key becomes truly useful when used with selecting text. After first selecting (usually by dragging with the mouse), hitting Backspace deletes *the entire selection*. A very useful action; the select/Backspace combination results in fast editing-by-deletion, which is what editing is mostly all about anyway.

One stumbling block to novice computerists is the idea of white space on the display. To Macintosh, white space in text is just as real as letters or numbers. The white space can also be deleted just as easily as characters. A good exercise involves tapping out a short note on, say, MacWrite, then centering the text by clicking the appropriate box. Next, drag through the non-text areas and Backspace to delete.

Your words will obediently shuffle around and give you new insights into “white space,” “blank characters,” “carriage returns,” and “what's really going on.”

**Other Keyboard Features**

Almost all the keyboard keys have an auto-repeat function: press down for a moment, and the key will automatically repeat. The following keys won't repeat: Caps Lock, Shift, Enter, Option, and Command.

Macintosh also features *typeahead*. Fast typists will never lose keystrokes, even if they type faster than the screen can display their efforts. The keystrokes are saved by Macintosh, not lost. It's also impossible to lose mouse clicks, Double-clicks, or drags; they, too, are saved if entered faster than Macintosh can process them. The software device here is called an *Event queue*, which queues up waiting keystrokes or mouse commands, then processes them in the order they were entered.
The keyboard also features Two-key rollover: the Macintosh hardware can detect any two-key combinations held down simultaneously—a necessary feature for Command key sequences. But beyond two keys, Macintosh cannot go, which means that Macintosh users will never, never, need to memorize three-key combinations to perform commands.

Yes, the Macintosh keyboard is softer than it appears. The letters and symbols associated with each key aren't "hard-wired." Instead, the keyboard is said to be "software-mapped." In practical terms, this means that non-standard keyboard layouts (such as a Dvorak keyboard) are easily configured, provided you have (or can write) the proper software utility program to accomplish the feat.

The Numeric Keypad

The Macintosh Numeric Keypad is an option; if you seldom work with numbers, you'll never miss it. If you do work extensively with numbers, you won't be able to live without it.

The keypad doesn't require an additional Macintosh socket. Instead, the keypad connects between the Macintosh and the keyboard, thus saving an additional socket (and a few pennies of retail cost for those who abhor numbers).

In general, the pad contains all the numbers you typically use, and the operators add, subtract, multiply, and divide. There's also a comma.

The keypad has the keys Clear and Enter as well. Clear is usually
used to clear single entries. At the risk of getting into the Return/Enter discussion again, let's just say that Enter enters. Don't be misled into thinking that the keys marked with arrows are cursor keys. They're not. Almost, but not quite.

Apple calls them field motion keys. Their use is strictly confined to spreadsheets, and other cell-based documents, where they provide a means to navigate between cells.

Of course...like the Macintosh keyboard, the keypad is software-mapped, so could, in theory, be used for just about anything. And the key tops can all be unplugged and replaced, so...how about a pad of function keys, each with an associated string of commands? How about assigning a single word to each key, for use with databases that allow sentence-like queries? How about a dedicated game-pad? How about a household control pad, for turning on lights, or starting the coffee?

It gets better.

---

**Guessing Ahead**

Keyboards and numeric keypads aren't the only devices that might plug into that little socket beneath the monitor.

For the moment, though, don't plug anything else into that socket. Other plugs (those on modern telephones, for instance) will fit, but won't work, and may cause damage.

In the future, we'll surely see more than keyboards and keypads used with Macintosh. The reason is the electronic interface that mediates between Macintosh and the Macintosh keyboard. The interface (ready?) is a general-purpose clocked bi-directional serial port.

Good.

In other words, any serial input device, if properly designed, can work with Macintosh—provided that it includes software to clue the computer into its new configuration.

The first possibility that leaps to mind is a musical keyboard. Macintosh already has the innate capability to produce four-voice sounds simultaneously. A piano/organ-like keyboard is a natural, and will probably arrive soon.

Beyond that, the future is cloudier. Graphic tablets (are they really necessary?). More control panels?

Who knows? Like other aspects of Macintosh, the keyboard, keypad, and related electronics have not only wide capabilities, but also intriguing possibilities.
5. The Finder: Macintosh's Operating System
Of all the many unpleasant qualities of computers, one of the worst, maybe the worst, is the Operating System.

In other computers, the operating system is the abhorrent, but necessary collection of command lines and modes that we mentioned earlier. A lousy, rotten, perplexing, and despicable way to give orders to a computer. Worse, a confusing jumble of commands that you, the user, must memorize and then type perfectly correctly.

Unfortunately, operating systems are necessary. Necessary because computers work with disks, and disks are made up of individual files of information. Some of the files are programs, others are data that work with programs, and other files are utilities that do helpful and needed things to other files.

There are many different operating systems. Many computers offer a choice between two or more versions. Some have more features than others, some are faster or slower than others, and some are larger or smaller than others. In general, all operating systems provide for these functions:

- Displaying the contents of the disk(s),
- Copying files from one disk to another disk,
- Renaming and erasing files,
- Loading files from the disk into the main memory of the computer. The file may be a language, or a program. There are many ways to do this: it does get confusing. And,
- Formatting disks for use by the operating system (generally, all operating systems have a unique way of preparing disks for use).

In addition, many operating systems offer other conveniences, if you ever learn to use them properly: comparing files, combining files, printing files, checking the amount of space remaining on disks, and even rudimentary methods of entering text directly from the keyboard into files.

Advanced operating systems even offer sub-directories, a way to manage many files. For example, all your word processing files might exist in a sub-directory, and all your business correspondence in a sub-directory of your word processing sub-directory.

A powerful idea, but complicated in practice.

Alas, as long as computers have files and disks, they will also have operating systems.
The Finder Unveiled

The Finder is the Macintosh operating system. It does most of the things that other operating systems do, but in a way that makes the term Operating System seem needlessly complex.

The Finder might also be called the Desktop Manager, or the Organizer; both candidates, by the way, when Apple was deciding what to call its operating system.

The term that won was the Finder; a title given by Apple programmers to the fruits of their long labors in the binary world.

It's a good title. The Finder, after all, let's you find things. It allows all the traditional operating system functions, and a few functions that no other operating system can offer (except for Lisa, of course).

Other operating systems work with commands typed from the keyboard. Macintosh, as always, works with objects, windows, and pull-down menus.

The most common objects are Icons: pictorial representations of tools, documents, actions, files, or entire disks. The Empty Folder icon is a good example of a possibility. It just hangs around within a window, hoping that someone will use it for something.

That's not a very specific explanation of what icons are.
Your first task as a Macintosh owner is to find out (via the Finder) what the various icons represent. This information is gained in a variety of ways:

Icons usually look like what they are. The Trash is used to delete other icons, and the information they contain. Text document icons look like sheets of paper filled with writing; graphic documents look like, well...graphic documents. Empty folders look like empty folders. And so on.

Icons have a title that describes their contents. If you come up with a better title, you can usually change it.

Using the File menu. First, click once on an icon. Next, place the pointer over the word File on the menu bar and press the mouse button. Down pulls a menu. Keep the button pressed and drag down to Get Info. Release. Up pops a window that explains the icon. After reading information about the file, click in the white Close Box in the upper-left corner of the window.

Or, Double-click (two fast clicks, remember?) on an icon. You’ll be rewarded with either a window that is empty or that holds still more icons, or you’ll find that you’ve just started a Macintosh application program. If you find yourself in an application, don’t worry; just drag down the File menu to Quit, and quit. Back to the Finder. No harm done.

Screen 5.1 Get Info Windows Selected From the File Menu
Fundamental Thoughts

The basic thought behind the Finder, and behind the entire Macintosh design, is this:

First you select something
then
you manipulate it.

Selecting is choosing. It's how you get the Finder's attention. Selecting can be done in a number of ways, some simple, some simpler. The simplest way to select an icon is to move the pointer over the icon and click once. The icon will immediately invert: white will become black, and black will become white. Point and click. The icon inversion is a basic Macintosh trait: when you do some-

Screen 5.2 An Inverted Icon After Selection
thing, Macintosh does something. You can immediately see the result of your action, even if the result is merely an okay, now what?. The constant feedback from Macintosh proves that you're in command. You always know what's happening on the Macintosh display.

Now Double-click to see what's inside an icon. Up pops a window, like that shown in Screen 5.3, full of other icons.

Are these other icons files? Sure.

Filing Thoughts

Rather not look at icons? Rather see something that looks like a more traditional directory of files? Okay, just open an icon that contains other icons. Now, go to the View menu and try out the various choices. Drag down and release the button when your choice is inverted. Give each choice a whirl. In most cases, you'll be rewarded with a line of file information for each icon: name, type, size, and date.

When your curiosity is satisfied, choose by Icon and change everything back to icons.
Screen 5.4  Files Arranged By Name

Screen 5.5  Open Trash Window
Next, open the Trash by Double-clicking over the Trash icon. Up pops another window: empty.

Nothing in the trash.

Is the Trash a file, also?

Sort of. Actually, the Trash is a good example of why thinking about files isn't necessary with Macintosh. In other operating systems, files are deleted by giving the proper command. In Macintosh, files (icons) are deleted by placing them in the trash then emptying the trash.

Trashing icons reveals one of the Finder's most engaging qualities: the ability of icons to suck up other icons. Well, maybe "absorb" is a better term.

Screen 5.6 Icon Moved Into Trash

For practice, place the pointer over the Empty Folder and press the mouse button. Now, without releasing the button, drag the folder across the display until the folder's outline is directly over the Trash. Notice that the Trash icon inverts when you center the Empty Folder icon over the can.

Now release the button. Voila! No more empty folder.

Where is it? Click twice on the Trash to find out: there's your folder, nestled in the Trash window.
Aahhhhhh!

If you've used other computers, about this time bells and whistles and rockets will begin going off in your head. Try and keep the noise down. If you haven't used other computers, all of this legerdemain may be a bit disconcerting.

Don't be nervous. Be bold. The Macintosh programmer's intention was to give you the maximum function and flexibility in working with everything within Macintosh.

That flexibility is best seen in the Finder. With the exceptions of Desk Accessories, Dialog Boxes, and Alert Boxes (which we'll get to in Chapter 7), these things will be found on the Finder desktop:

- The desktop itself,
- The Menu Bar and associated menus,
- Icons on the desktop,
- Windows that display what's inside icons, and,
- More icons within those windows.

Now the fun begins. Drag the Empty Folder out of the Trash window and back into the other window. Next, drag the Empty Folder
off the window and onto the desktop. Now close the window by clicking the Close Box (the white square in the upper-left corner of the window).

There's the Empty Folder, all alone, no longer just one more icon in some window.

Now experiment, a suggestion that will be made, again and again, throughout this book. Drag icons in and out of windows. Close the windows and open them up again. Try out all the menu options. Get a feel for the possibilities of the marvelously easy manipulation of files. Close windows by using the Close Box or by choosing Close from the File menu. Both methods do the same thing: close a window.

Notice that opened icons leave an outline behind, to mark where they came from.

Now, try out the Put Back selection, also on the File menu. Watch how it puts icons back where they came from.

**So What?**

So what! So what!

Look at it this way:...no, better yet, let's do one more exercise.

Click once on the Empty Folder icon that now rests alone on the desktop. It inverts. Now, go to the File menu and choose *Duplicate.*
Presto: another folder is created. The new folder is titled: copy of Empty Folder. Now drag it over into a vacant corner of the desktop. Now open up Empty Folder, by Double-clicking on it.

Finally, drag copy of Empty Folder into the Empty Folder.

You've just created a sub-directory. Not a very important sub-directory, since it contains nothing, but a subdirectory nonetheless.

The possibilities inherent in all this clicking and dragging are far-reaching. Most important is this: you're in control: your files, programs, documents, and anything else on your Macintosh disks are at your command. Anything can be moved from one window to another. Anything can be opened and examined in a variety of ways.

Organizing information has never been easier. And organizing and manipulating information is what Macintosh is all about.

For instance, you might have an icon named Personnel. You open it up and see a number of other icons: Full-Time, Part-Time, and Temporary. Each one a separate icon. And each of those icons might contain still more icons. The Full-time icon might contain other icons: Hourly, Salaried, The Boss, or whatever.

Or, you might have an icon named Appointments. You open it up and see other icons: Today, This Week, Holidays, Family, and Golf. Each one a separate icon. The Golf icon might contain the icons: With the Boss, Tournaments, and Golf League.

And each of those icons...

Screen 5.8  Four Layers of Subdirectory Windows
Directories, sub-directories, sub-sub-directories, and sub-sub-sub-sub-sub-directories. All as easy to manage as clicking a button on a mouse.

That's just a taste of the Finder. There's more. Not only can file icons be arranged, re-arranged, and managed, but information within the files can be easily transferred from one application to another.

The subject deserves a chapter all to itself, and gets it, later, in Chapter 9.

For now, let's look at some of the more fascinating Macintosh objects: the Desk Accessories.

**Desk Accessories**

All computers require programs. To add two numbers, it's necessary to either buy a program that can add numbers, or buy a computer language and learn to write a program that can add two numbers.

Painful. If you choose the simplest route of buying a program that adds, it's still necessary to properly load and run the program, and give the commands that will add the numbers together.

In real life, people use calculators to add numbers.

So Macintosh includes a Calculator. It's a Desk Accessory, and looks and behaves like more concrete desk accessories found on more typical desks.

Selecting an accessory is easy. Just pull down the Apple menu, the black Apple that always appears at the extreme left of Menu Bar, drag down to whatever looks good, then release the mouse button.

The current choices include:

- Calculator
- Clock
Note

The Note Pad holds up to eight pages of notes. Pages can be flipped forward by clicking the upturned corner, or backward by clicking the page visible underneath. Only one type font, style, and size allowed, but it's enough; a truly useful accessory.

• Note Pad

Key Caps

The keyboard in 'Caps Lock'

• Key Caps

Control Panel

• Control Panel

Puzzle

• Puzzle

Scrapbook

This space for real...

• Scrapbook
The Scrapbook, Key Caps, and Control Panel aren't found on most desktops, but provide some interesting opportunities.

For starters, the Calculator appears reassuringly familiar. It works like an ordinary, four-function calculator. Use the mouse to punch in a few numbers. Try adding, subtracting, multiplying, and dividing. To clear an entry, click over the C (for Clear). The results of your clicks will be totally self-explanatory, unless you've never used a hand-held calculator, which isn't likely.

The Calculator can be moved by dragging with the pointer positioned on the black Calculator Title Bar. The Calculator can be dismissed by clicking in the Close Box in the upper-left corner.

You never liked numbers anyway? Drag down to Alarm Clock and release. (Don't worry about putting the Calculator away by clicking the Close Box. It can hang around until you're done for the day, or be dismissed whenever you wish. It can be completely covered by other windows or objects or fully visible. Makes no difference. You decide.)

The Clock is repositioned just like the Calculator: drag on the top, black border. The Close Box is located, as always, in the upper-left corner.

The Clock displays the current time, down to the second. Thanks to an internal battery accessible through the Macintosh back panel, the Clock continues to keep perfect time, even if the computer is off or unplugged. The Clock's time, date, and alarm can be set by clicking the arrow at the right of the display, thereby revealing the lower section. The time can also be set from the control panel.

Next is the Note Pad. Although a Desk Accessory, discussion of the Note Pad belongs with editing, the Clipboard, and the Gallery. So that's where discussion of the Note Pad is: Chapter 9.

The next accessory, Key Caps, is a good demonstration of Macintosh graphics, and the flexibility of the Macintosh keyboard. Select Key Caps from the menu, then press the keyboard Shift key. The keyboard representation of the Shift key immediately changes to black and also displays new key cap legends: shifted characters. Next press the Option key for a look at other strange symbols accessible from the keyboard. Some are useful in math, some for foreign languages, and some for business.

Because the Macintosh keyboard is completely software re-definable, future programs may offer totally new keyboard designations and, possibly, even new key caps to plug onto your keyboard (not at all difficult to do).

Practice typing and watch the letters appear on the keyboard display. Or, "type" by clicking the keyboard key caps with the mouse—that works, too. For a preview of Mac's editing capabilities, drag over the typed words, then hit the Backspace key. Aha!
Beyond demonstrating Macintosh's keyboard flexibility, the Key Caps accessory is a great way to impress your friends. But then, all the accessories are pretty impressive.

Next is the Scrapbook. The Scrapbook has much in common with the as-yet-unintroduced-Clipboard. (More to come in Chapter 8.) In general, the Scrapbook transfers pictures and text from MacPaint (or other graphic tools) into other applications. In a sense, it's another Clipboard, with some unique features. Take a peek then put it away.

**The Control Panel**

Of one the most visually attractive accessories is the Control Panel: a way to customize Macintosh to your preferences. (In fact, a similar accessory with Lisa is called Preferences.)

The Control Panel lets you select or adjust:

- The volume of the Macintosh internal speaker,
- the appearance of the desktop,
- the number of times that selected menu items twinkle when chosen,
- whether keys repeat when pressed; and, if they do repeat, how quickly the repeating begins, and the speed of repeat,
- the length of time permissible between two clicks of a mouse Double-click,
- the rate at which the mouse travels over the desktop surface,
- the speed of Insertion Bar blinking, and
- the time displayed by the clock.

The slider at the panel's left sets speaker volume. Use the mouse to drag the position indicator. When you release the mouse button, or merely click the slider, the speaker beeps once to indicate the volume setting. It is even possible to completely turn off the volume. (Talking Macs probably won't show up for a few months, but they will certainly show up, because they do exist.)

One of the most fun controls is the Desktop Pattern Editor: two boxes within a white box located at the bottom-middle of the Control Panel. The box at the left is an enlargement of the current desk-
top pattern: a collection of large, black dots. At the right is a repre-
sentation of how those dots will look as a new desktop pattern.

The white bar at the top of the right-hand box is used to page
though a number of pre-set patterns. Click the white bar, and the
new patterns appear the box below. When you see a pattern you like,
move the pointer down into the box and click: the pattern now fills
the entire desktop visible on the Macintosh screen.

But why be satisfied with someone else's choices? Move to the
box on the left and click individual black squares, or drag through
the squares. Either method creates your own pattern, which is imme-
diately shown in the right window. [Notice, too, that if you begin
dragging when on a black square, that square and all subsequent
dragged over squares will be erased. Beginning a drag when not on a
black square draws squares under whatever areas are dragged over.
The principle is called Fat Bits and shows up again, delightfully, in
MacPaint.]

Find a pattern that best expresses your unique personality. When
your personality changes (or you get tired of the current desktop)
change the pattern.

Between the volume slider and the Desktop Pattern Editor is a
box filled with what looks like a mouse in motion. Beneath the
mouse are two small buttons, marked 1 and 0.

The buttons control the way rolling the mouse translates into
pointer motion on the Macintosh screen. When 1 is selected (by
clicking within the button), the pointer flies across the screen with
very little mouse motion. When 0 is selected, it's necessary to roll
the mouse farther to produce an equal amount of pointer movement
on-screen.

Click 1. Now place your pointer at the top-left of the screen and
roll the mouse horizontally. It will take only a few inches of mouse
movement to move the point to the opposite side of the screen. If 0
is clicked, it takes about double the roll to get the same results.

Which to use when? In general, select fast movement; it lets you
work faster, and makes the most of small real desktops. For precision
work within documents, where exact mouse positioning is desired,
use the long roll choice. The precise setting is especially useful in
graphic documents, where work may be done pixel by pixel. Those
black squares, in the Desktop Pattern Editor, by the way, are enlarge-
ments of individual screen pixels.

In the center of the Control Panel are five boxes, marked 0 through
4, as shown on page 54.

The boxes control keyboard repeat. If box 0 is clicked, the keys
will not repeat. Clicking 1, 2, 3, or 4 will cause the repeating effect
to begin at varying lengths of time after the key is first depressed.
Click 4 and the keys repeat almost immediately after the first character is displayed. Click 1 and it takes awhile for the repeating to begin.

Only the fastest typists, with the lightest of touches will dare choose 4; most others will select no repeat (0), or repeat only after a civilized wait of a few milliseconds.

Moving along to the right of the Control Panel, in the corner, is a mouse with a finger upon it, flanked by three more buttons. The buttons determine how long Macintosh will wait between mouse clicks, and still assume that you mean to Double-click. Experiment with each. You'll probably choose the far-right button, which sets the shortest allowable time between clicks. After all, you did want to Double-click, not double click.

In the upper-right corner are buttons that set menu item twinkle. You may not have noticed, but when you select an item from a menu, the item twinkles briefly when you release the mouse button. Here's your chance to have lots of twinkles, some twinkles, or no twinkles at all. 3 is maximum twinkle. 0 is no twinkle at all. You decide.

Who thinks all these things up, anyway?

Control Panel settings need only be made once. When you quit for the day or change to another disk, the Control Panel settings are saved on disk. When you next use that disk, your Control Panel settings will be automatically put into effect. You might wish to identify particular disks with particular desktop patterns; black for, say, disks full of numerical tools, or a plaid for word processing.

**Wherever, Whenever**

The benefits of Desk Accessories are two-fold: they can be used anytime, anywhere, no matter what else you're doing with Macintosh, or what application program you're currently working with. Just pull down the menu, pick an accessory, and use it. The second, and best benefit of Desk Accessories, is that information from some of the accessories can be copied *from* the accessory, then *into* whatever you're currently working with. Need to add some numbers for addition into a letter? Get the Calculator, add, then copy the result into your letter. How? We'll tell you in Chapter 9.

**Still to Come**

Those are the accessories now packaged with every Macintosh. But
more accessories are sure to be offered; if not by Apple, then by other software developers. Applications software that you purchase may add a number of specialized accessories, each as easy to use as those delivered with Macintosh.

It's possible to envisage software that consists of nothing but new and useful accessories, for use with some or all of your other Macintosh tools.

For starters, how about a Rolodex, a scientific calculator, a U.S./metric conversion chart (with lots of dials and buttons), a Dow Jones stock ticker (for use with the Macintosh Modem), an automatic telephone dialer (pushbutton, maybe a Princess model), and a few more Control Panels: one to run the household appliances, one to water the lawn and turn on outside lights, one to feed the dog...
6. Windows
"Look through any window, yeah; What do you see?"

— Look Through Any Window
The Hollies

Time to discuss one of the most important Macintosh objects of all: windows.

In general, information displayed by Macintosh is always shown within a window. It may be a document window with all the border accompaniments, an on-screen message or warning, or a request for additional information.

But it will be in a window.

It's helpful to think of windows as sheets of paper on a real desk-top. Like paper, there may be lots of windows on the desk at any time. Like paper, they may overlap, or be neatly stacked, with the top page hiding many pages underneath.

But the analogy breaks down, for windows can do things paper can't. Macintosh windows:

• can be dragged to the edge of the desk, but never dragged completely off the desk,

• can be resized easily; from small as a matchbox to large enough to fill the entire display (less menu bar), and

• can be dismissed with the click of a mouse button.

Also unlike paper, each window may contain many windowfuls of information.

It's okay to think: windows contain information; but to be even more precise, think of them as windows into information. You're looking through the window and seeing what lies within; and there may be more in there than can be shown all at once, even if the window covers the entire display.

To open a window, either:

• click an icon that represents a document or application program, then select Open from the File menu, or

• Double-click the icon.
Screen 6.0 shows a fully decked-out document window. Not all document windows will have all the border additions shown here, but all documents will always appear in a window—or more than one window—and all document windows will have:

- a title bar across the top, with the title centered and displayed in the Macintosh System font and . . .

It's tempting to go on and list scroll bars, etc., but they might not be a feature of every document window. The same goes for Close Boxes; many document windows can only be closed by selecting Quit from a menu bar.

Before we begin looking at all the visible features found around windows, let's delve a bit into the Macintosh binary world: the mass of routines stored in the Macintosh internal memory (ROM).

One group of routines bears the collective name of Window Manager. The routines provide programmers with means to create, move, re-size, and shuffle windows on the desktop. The presence of the Window Manager routines in ROM makes it easy for programmers to create new applications with all the present Macintosh windowing sophistication; their use will also insure that all Macintosh windows look and behave like all other Macintosh windows. The Window Manager also gives programmers the capability of defining windows with unique characteristics—a powerful option, if used wisely, not merely to confuse users.

The Window Manager breaks (sorry) windows into two classes: document windows and system windows. System windows, in gen-
eral, surround Alert Boxes, Dialog Boxes, and all Desk Accessories.

A necessary windowing concept is the Active Window. Only one window at a time can ever be active, or on. Only active windows can be moved, re-sized, or written-in. If only one window is present on the desktop, it is always active.

If two or more windows are present, the active window is always the frontmost window. Put another way, the frontmost window is always active.

In a pile of overlapping windows, the frontmost window, again, is active. To make any other window active, click in any portion of any visible window (and only a sliver of the window needs to be visible). This will:

- bring the window to the front, and
- make it active.

Two exercises will make the idea of an active window clearer. For the first, open two icons, then size the resulting windows so that both fit, side-by-side, without overlapping. (The fastest way to do this is by opening, say, the Empty Folder and Trash icons shown in the Finder.)

What you now have should look something like what's shown in Screen 6.1.
Screen 6.2 Overlapping Windows

Notice that one of the windows has a filled-in window frame, and that the other window has a slightly de-nuded appearance. The filled-in frame is a sign of activeness.

Now move the pointer back and forth between the windows, giving each a click. Each window will come alive at a click of the mouse button. Don't worry about clicking too fast, it can't be done: Macintosh will merely store any clicks, then process each when it gets around to it (which is usually faster than you can observe).

For exercise two, open up another icon and move the three windows around until they overlap in a casual, yet winsome fashion. Let your good taste be your guide.

Now, once again click in different portions of the three windows; notice how one click brings a window to the front, and makes it active. Notice also that the first click in an inactive window will only bring the window to the front and make it active; you can't hold the button down and use that first click as the start of a drag or close or any other window action. The first click on a back window brings the window to the front and makes it active. Nothing else.

For the truly adventurous, and those not completely bored with merely clicking windows, here's an optional extra credit exercise: use Duplicate on the File menu to make a number of Empty Folder windows. Practice rearranging them, clicking them to the fore, and so on.
Screen 6.3  Plethora of Windows

This should answer your unspoken question about how many windows can be open at once. The answer, of course, is more than you would want open all at once. It does get distracting. Also, when the number of open windows reaches eight, the Finder will ask you to close one or more, as those seeking extra credit will find.

There's more to the question of what's on top. In fact, the hierarchy of on-topness has been carefully specified by Apple. The rules, if you will.

1. The desktop is always at the bottom. Nothing can ever be below the desktop. Nothing. Ever. Really.

2. Next are windows: inactive below, active on top. Many may be inactive, and at various levels of belowness, but only one can be active and on top.

But other things can be above active windows.

3. Pull-down menus are always above all windows.

4. Dialog Boxes are above even pull-down menus. You wouldn't want to miss an important message like: "having a nice day? [Okay] [Cancel]" just because it was buried under a bunch of windows and menus.

5. Alert Boxes are above Dialog Boxes, which gives a good indication of the difference between the two. Alert Boxes demand attention and, usually, action. A message like: "Go to bed or I'm going to
erased your disk. [Okay] [Who cares?] could properly be called an Alert Box.

Anything left? Yes: the pointer. The pointer is always, always, topmost, if not in your mind, at least on your display. It can't be hidden or removed.

Now back to windows.

**The Concepts**

Before discussing what windows consist of, let's look at how the Window Manager sees windows.

To Macintosh software, a window consists of two general areas:

- a *content region*, that the application program uses to display information, or for the user to write in, and
- a *structure region*. The structure region is the entire window and includes the window frame.

Things get interesting when we examine the window frame. Again, the frame is made up of a number of different regions, including:

- a *drag region*,
- a *grow region*,
- and a *go-away region*.

So what's a region? For our purposes, we can simply say that a region is any window area that Macintosh software evaluates in a particular way. In most cases, the regions also have visual representations: the drag region is represented by the Title Bar, the grow region by the Size Box, and the go-away region by the Close Box.

But the visual representation is, at least partly, cosmetic. A go-away region can, if the programmer wishes, include the entire window or box. One click anywhere and the window's gone.

Although most document windows will have the familiar visual Close Box and Size Box, other types of windows will often implement the same basic concepts in different ways. The box marked Okay in a Dialog Box is, after all, just another way of presenting a go-away region in a different form.
Scrolling Apparatus

What left on the window frame? The area along the right side and along the bottom of the window: what Apple calls the Scroll Bars.

Because windows may not be able to show an entire document at once, some method of moving the information through the window is needed. The solution, originated long before Lisa or Macintosh came into being, was elevators.

Elevators are what Apple calls Scroll Bars. Elevators is a more metaphorical, fanciful term; Scroll Bars is a more descriptive term. Take your pick. The author, however, enjoys elevators, coming, as he does, from a barely civilized part of the world that is sorely lacking in elevators.

Elevators have three components: scroll arrows, the gray bar, and
the white rectangle within the bar, which Apple calls the Scroll Box, and the rest of the world calls the thumb. The thumb is a visual indicator of the portion of the document currently displayed in the window. When the thumb is at the top of the bar, you’re looking at the beginning of the document; when it’s at the bottom, you’re looking at the end.

Elevators provide four ways to scroll:

- Clicking the up or down arrow scrolls a line at a time.
- Pressing the up or down arrow scrolls continuously.
- Clicking in the gray box scrolls by page, forward or backward, depending on whether the click is above or below the thumb.
- Dragging the thumb to a new position displays a new portion of the document. Dragging to the middle of the bar results in a display of the approximate middle of the document, an easy way to thumb through a lengthy document.

When using the scroll arrows or clicking in the gray box, the thumb will instantly move to show the new position of the document.

Horizontal elevators, displayed at the bottom of the window, provide the same methods to scroll horizontally through documents that are wider than can be shown in the window. A common use for horizontal scroll bars is electronic spreadsheets, where the information may stretch across many columns. Not all document windows, however, will include horizontal Scroll Bars.
Other Paneless Encounters

There's one more window feature that you may encounter: splittable windows. Windows that offer this feature can be split into two separate panes. Each pane can then be scrolled independently, to see two separate portions of the same document at the same time.

The ability to split windows is often helpful, particularly when viewing long, detailed contracts, large spreadsheets, or when comparing different sections of lengthy text documents.

Windows that allow splitting will show a split bar icon, either at the bottom of the vertical scroll bar, or on the right end of the horizontal scroll bar. The split bar is operated just like the scrolling thumb: drag the split bar to a new position along the scroll bar and release the mouse button. The window will then split at the chosen location, and new scroll bars will be created for each pane.

We can't offer an illustration of a splittable window. At present, none of the Macintosh software uses splittable windows, but the capability is present in every Macintosh.
7. Dialogs and Alerts: More Windows
Just the Same, Only . . .

Only different. Dialog Boxes, Alert Boxes, every other kind of box that shows up on the Macintosh screen: they’re merely different types of the familiar window.

The evolution of these boxes is interesting, and goes back to the concept of Modes. Modes, as you remember, are particular system states that demand certain kinds of responses from users.

Even with Macintosh, there are times when a specific response to a specific question is needed before anything else can happen. Times when you need to be in a form of mode.

But Mac makes it as painless as possible.

As an example, let’s say you’re ready to print a document. You drag down the proper menu to Print and let go. Immediately, up pops a Dialog Box, like that shown in Screen 7.0.

Now, obviously, if you want to print the document, you’ve got to answer some questions. At the very least, you must agree to the Macintosh preset choices about size of paper, continuous or single sheet paper, and printing speed.

Unavoidable questions. In a very real sense, you’ve just entered what’s called [in other computers] a command mode: you give the commands, and the computer carries them out.

Macintosh, being a less militant machine, calls the boxes Dialog

| Quality: | ○ High | ○ Standard | ○ Draft | OK |
| Page Range: | ○ All | ○ From: | To: |
| Copies: | 1 |
| Paper Feed: | ○ Continuous | ○ Cut Sheet | Cancel |

Screen 7.0  Dialog Boxes
Boxes. Rather than giving commands, you have a dialog. (Someday, we may have machines that truly have dialogues with us; for the moment, we must settle for Dialogs: a scripted conversation rather than a real conversation. These future, more educated Macs will probably be named Voltaires, or Bacons, or Carsons.)

When They Come, How They Look

In general, Dialog Boxes appear whenever Macintosh needs information to carry out a command. All Dialog Boxes, by convention, appear slightly under the menu bar (never obscuring the menus), are narrower than the screen, and are centered in the display.

This is to get our attention.

Dialogs may contain a number of things:

- icons,
- controls,
- rectangles for text entry ("What should we call this mess?),
- comments, information, instructions,
- graphics, or
- anything else that programmers can devise.

Dialog Boxes don't always require that you make choices. At their simplest, a Dialog Box may pop up merely to make an appropriate comment: "This certainly is a sparse looking Dialog Box, isn't it? (Yes, now go away) (Just go away)."

Come to think of it, Dialog Boxes (and Alert Boxes) have two other common characteristics:

They show up without being asked for, and

They go away after getting what they require.

Their requirements may be as modest as a click in an Okay or Cancel Box, or as complex as specifying the complete format for a lengthy document; but they will go away after getting whatever they require.

In theory, programmers could create Dialog Boxes that appear for a fixed length of time then vanish of their own accord, but Apple strongly frowns upon objects having a mind of their own. We, the
users, are always supposed to remain in control, and to click the boxes back into the binary bowels of Macintosh. The only possible exceptions may occur during lengthy operations: a box may appear to update the progress of whatever ("3,354 items sorted, 5,598 left to sort"); then disappear. Even here, though, a click will usually be required to do away with the message.

Modal and Modeless Dialogs

Dialog Boxes that require an immediate response are called Modal Dialog Boxes. Failing to address a Modal Dialog Box, and attempting to continue work outside the window will reward you with an annoyed beep. Only a beep: attend to the dialog; try anything else and you'll only be beeped at, again and again.

The second type of Dialog Box is a Modeless Dialog Box. Modeless Dialog Boxes have a slightly different appearance: they may have a title bar or an enhanced border. Modeless dialogs don't require an immediate response; you're free to perform work in other windows while the dialog is displayed.

Screen 7.1  Modal Dialog Box
Screen 7.2  Modeless Dialog Box

Modeless dialogs are most useful in situations where you need the command box to remain in view, often to enter frequent commands. An example might be a Star Trek game, where a Modeless Dialog Box might represent a control panel that would allow fiddling with engine thrust, navigation, and various forms of blasters and space torpedos (space torpedos?). Because commands would be entered often in the control panel, a disappearing/reappearing Dialog Box would be unwelcome.

A more serious example might involve spelling corrections of a lengthy text document. A spelling correction Dialog Box might appear, then list potentially misspelled words, one by one, for your inspection and correction. While the box remained on screen, you could also continue editing text. A perfect use for a modeless Dialog Box.

Alert! Boxes

Alert Boxes are more strident cousins to Dialog Boxes. Alert Boxes mean business; they appear when:

- Something has gone wrong
- Something is *about* to go wrong, or
- Immediate action is required, usually to forestall doom.

Warnings, errors, alerts, or calls-to-arms are the reasons for an Alert Box's being.
Alert Boxes differ in placement and appearance from their milder relatives. An Alert Box will usually have a wider, blacker border than a Dialog Box. Also, they will usually appear farther below the Menu Bar than will a Dialog Box—more toward the center of the screen. Dialog Boxes request your attention; Alert Boxes demand your attention.

Alert Boxes will seldom be filled with controls, dials, check boxes, or radio buttons. Instead, the message will be simple and to the point. One specific message or request. In almost every case, there will be a response that is strongly suggested; this response will take the form of a boldly outlined button.

Here's a typical Alert Box example:

Choose Cancel at your own risk.

Alerts will often be accompanied by one or more beeps from the Macintosh speaker—a good indication of the severity of the Alert. Often, the first time an Alert is presented, a single beep will sound. If the proper steps aren't taken, two beeps will accompany the next Alert Box, and so on.

But don't be alarmed by Alerts. The primary purpose of Alert Boxes is to give you a graceful way out of even the stickiest situation; the proposed choice will almost always snatch victory from disaster.
What are some potentially sticky situations? Let's see ... how about:

- attempting to print when the printer isn't switched on?
- attempting to save a document on a disk that's already full? Or,
- attempting to copy documents to a disk (or other device) that just isn't there?

The most dire alerts of all are those that involve system crashes. A system crash is never your fault; it's the program's fault. Bugs, or errors, within programs often cause crashes. A crash can manifest itself in a number of ways: the program may suddenly hang-up: suddenly everything on the display is totally unresponsive (you might not even be able to move the pointer!). Or, you may see an Alert Box with a dreaded message such as: Stack overflow error, or Address error.

Even programmers make mistakes. All that you need know about subjects such as stack overflows is that there is something called a stack in some computer programs, and if the program is poorly written, the stack can overflow.
Which no one hopes will happen.
If it *does* happen, or something equally disastrous happens (and equally beyond your control to prevent) there's usually only two solutions, neither of them particularly pleasant:

- return to the Finder, or
- reboot the system.

Returning to the Finder is a nice way to say that everything you've done so far is gone.
Rebooting is a nice way to say that the machine will now turn itself off, then on again [re-booting]. This is another way of saying that everything you've done so far will soon be gone.
But this may never happen to you. Remember, this is a mark of shoddy, bug-ridden, repellent, despicable software. With any luck, all the programs you purchase for Macintosh will perform flawlessly.
If not, write the manufacturer of your program, inform them of the bug, tell them when and where it occurs, and tell them if it's possible to deliberately make it occur (this will help them isolate and fix the problem). Next, demand a solution. This may not produce any results, but it should make you feel better.

**Controls: All Types**
Controls are usually found in Dialog Boxes. What are controls? Any object that lets you express your wishes. That wish may be as simple as clicking Okay to dismiss a Dialog Box, or setting a number of check boxes, radio buttons, or dials.
The most common control is a button. Here are two of them:
Cancel Buttons require one click within the chosen button. Buttons are usually "this one or that one"; not both. A clear choice of command.

For situations where more than one selection is necessary, or possible, there are check boxes, another form of button. Here's a typical example:

![Check Boxes](image)

**Screen 7.5 Check Boxes**

Check boxes are also clicked once to mark your choice. More than one choice may be made. The click usually results in an X being placed in the box. Not a check-mark, an X. (X-boxes? Doesn't sound as good.) Radio buttons, in turn, are a variation on check boxes. Their name and behavior comes from station select buttons on car radios; like those buttons, only one radio button can be selected at any given time. Selecting a new button de-selects the previous choice. An example might be the type of paper used to print documents: single sheet, or fanfold. You wouldn't really want to select both, would you? Selected radio buttons are marked with a round, black dot, rather than an X. Here's an example:

![Radio Buttons](image)

**Screen 7.6 Radio Buttons**
Last are dials. Dials are visual representations of continuous magnitude, rather than discrete states like on or off. Instances of magnitude might be the volume of a speaker, level of fuel in a tank, time remaining in an operation, or a visual display of storage available on disk. The dials most often seen are the scroll bars, which have both buttons (the scroll arrows), and a visual dial (the Scroll Box, or thumb).

Expect programmers to have lots of fun designing new dials for Macintosh. Expect to have a lot of fun using them. Clocks with hands, car gauge dials, thermometers, and anything else that tells a tale, or gives helpful information is a good candidate for dialization.

Screen 7.7 Dial
Windows. As usual we build
software to do what it's built
for. But text editors that does it is built
not surprised at all that
a text editor is located in

Text Editors

1. Text Edit

"I'm not sure if this is enough Text Edit
including fonts and stuff.

The programs work "blind"
and allow all the really really
next stuff."

8.

Text Editing
Intro to Editing (Macintosh 101)

A brief and rather sketchy introduction to text editing on the Apple Macintosh computer. Students will read a chapter in a book about the Macintosh, then try out what they've learned, provided they buy a Macintosh. Attendance, questions, papers, final exams, and letters to the author are optional. A knowledge of preparatory material is assumed on the part of all those enrolled. Class will meet M, W, F, 7–9 am.

Do you still want to read this chapter?

TextEdit

So far, we've talked about windows, and noted that the software that manages windows is built-in to Macintosh. We've also talked about mouse actions and the Macintosh keyboard again, the software that makes Mac behave as it does is built into the machine.

You probably won't be surprised to learn that a text editor is also provided in Macintosh's internal software ROMs.

Actually, there are two text editors: one, called TextEdit is built-in; the other, called CoreEdit is an optional chunk of programming code that can be placed on a Macintosh disk. If CoreEdit is present, it adds additional features to TextEdit: the ability to use different type fonts and type sizes, among other things. You'll never see either program, but their existence underlies all the typing, changing, cutting, copying, and pasting that Macintosh allows.

Cursors

Except for Macintosh and Lisa, most computers still have something called a cursor. Cursors are necessary for most computers. They indicate where the next character typed will appear on the screen. Cursors are often short horizontal lines to the immediate right of the last character typed, or white blocks that blink insistently.

In other computers, the screen itself is arranged as a number of lines; usually 24 or 25 horizontal lines. The cursor is always located at some position on one of those lines. Most computers allow you to move the cursor down the screen by hitting Return. It's often impossible, however, to move the cursor up again. Backspace will
usually allow you to move the cursor left along the current line. When you get to the leftmost position, you're usually stuck, unless you want to move the cursor to the right (usually by typing something), or straight down the screen, if that's allowed.

Extremely clumsy.

Macintosh does away with all that. The mouse and pointer work together to make your current location be wherever on the screen you choose. There are no invisible lines that force you into particular locations. And no cursor.

But display screens are still with us, and keyboards are still with us, and typing and changing text is still moderately desirable to most users. Knowing where the next letter you type will appear on the screen is, after all, somewhat helpful.

So, having eliminated the cursor, Apple was forced to reinvent it. The reincarnated cursor is vastly niftier than other cursors. Not that Apple even deigns to call it a cursor; what we have in Macintosh is an

*Insertion Point*

### A Fine Point

The Insertion Point is seen whenever you work with text. But the Insertion Point is *not* a point; it is a thin, vertical, blinking line. Unlike cursors, which are as wide as individual characters, the Insertion Point may be placed *between* characters for precise editing.

The Insertion Point marks the point where the next character typed will appear. The text is always inserted; there is no typeover mode in Macintosh; any text that appears to the right of the Insertion Point will be pushed over as the new characters are entered.

Now's a good time to begin typing. If MacWrite isn't handy, drag down the File menu to Get Info and begin typing in the comment box. Notice that when you reach the end of a line, the next word is automatically placed at the beginning of the next line. That's called *word wrap*: it wraps the text to the next line without your assistance. Notice also that the Backspace key moves the Insertion Point to the left, erasing any character in that position, and pulls any text to the right of the Insertion Point along.

The Insertion Point has only one function: to mark where the next character will appear in the text.
In text documents, the Insertion Point and the mouse pointer work together. In text, the mouse pointer changes from the familiar arrow to become an I-bar: a vertical line capped, top and bottom, with somewhat horizontal bars.

The I-bar is a change of pointer appearance, but not a change of pointer action. The mouse controls the text pointer as always, swooping and gliding over text documents as easily as over the desktop.

The Insertion Point is a selection, like all other Macintosh selections. A thin, blinking selection, but a selection nonetheless. Think of it as a very, very narrow selection.

The pointer, like all other pointers, determines what is to be selected. Move the pointer anywhere in the text and click: that's the new location of the Insertion Point.

Notion #1, then is:
Single clicks in text select [position] the Insertion Point.

Practice a while on a glob of text. (If Moby Dick wasn't included with your Macintosh, use the comments in the Get Info box for practice.) Click. A new Insertion Point. Click somewhere else. There it is again. Try to click precisely between the last character in a sentence and the period. Click. There it is, nestled before the period, calmly blinking.

Super Super Ciehoffienstoer should be available from Apple
More Clicks

If single clicks select, then double-clicks probably also do something, correct? Double-clicking is usually an enhancement of single-clicking, or an added feature, correct? Some blown-up way of selecting, right?

Without a doubt. Single-clicking in text selects an Insertion Point within the text. Double-clicking in text selects an entire word.

Notion #2:
Double-clicking selects individual words.
Again, try it. Double-click madly away in text.
Notice that the word selected is immediately inverted.
Notice that the next click outside the selection un-selects the word and selects a new Insertion Point.
Notice that selecting a new word un-selects the previous selection.
Notice that clicking in the white space beyond the last character chooses the space after the last character as the new Insertion Point.

Dragging the Text

After single-clicks and double-clicks, dragging is inevitable.
In text, dragging is used to select portions of the text. The portion

```
program, click 'Go', and the program runs right now and amazingly fast. Super. Super. Clehofferstoer is an ideal Clehofferstoer for beginners, but also offers features that Clehofferstoer veterans will appreciate, including a number of pre-written procedures and functions to access windows, menus, buttons, and other nerdish delights.
```

Screen 8.1 Text Selected by Dragging
selected may be as small as a single character, or as large as the entire document.

Click once to set a new Insertion Point, then hold the mouse button down and drag across a line. The text that is dragged through is selected and inverted.

Click in the middle of the line to place a new Insertion Point. Now drag forward from the Insertion Point. Now move the pointer somewhere else and click once to de-select. Now again place the pointer at the Insertion Point, only this time drag backwards along the line.

Works both ways.

Now get creative. Move the pointer to the Insertion Point and drag up or down through the text. Aha! Big selections.

Drag in circles around the Insertion Point and watch what becomes selected.

Do this for six months, or until you get the hang of it.

The 3rd notion:

Dragging selects a portion of text, from a single character to all characters. (Dragging aside: white space can also be selected, as you've found if you correctly followed the instructions.)

Then What?

Isn't learning great?

Now that you've managed to select text in a variety of ways, it's only fair to mention a few things that can be done next. There is more to editing than merely seeing text white-on-black.

Here are your options:

*Typing*. Actually, pressing any number or character key on the keyboard. The first keystroke will delete the text selected and enter the character you typed. This is a shortcut; rather than select, then delete, then type, all that's necessary is selecting and typing: the selected text leaves, the new text begins. And in most cases, the old text is gone for good. If Undo Typing is found on a nearby menu, it *might* get the deleted text back, but don't count on it. One extra click between deletion and Undo Typing will make the deleted characters un-undoable.

*Backspacing*. One of the handiest ways to delete text. First select, then hit Backspace. Wham: it's gone. Text deleted this way can usually be restored [if done right away] by use of Undo.

*Cutting*. A menu option, Cut removes selected text and places it
program, click 'Go', and the program runs: right now and amazingly fast. But also offers features that Cleeoffenstoer veterans will appreciate, including a number of pre-written procedures and functions to access windows, menus, buttons, and other nerdish delights.

Super Super Cleeoffenstoer also boldfaces Cleeoffenstoer

Screen 8.2 Text After Deletion by Backspacing

on the Clipboard. The text stays on the Clipboard until new text is Cut or Copied. The new text then replaces the old text. At present, the Clipboard can only hold one selection at a time, regardless of how large the selected text is: one character or one page, it doesn't matter; when the new text shows up on the Clipboard, the old text is gone for good.

Copying. Another menu option, Copying places text on the Clipboard but does not delete it from the text. With that exception, copying works just like cutting.

Of the four options, the less glamorous Typing and Backspacing will probably be used most often.

Real World Writing

Preparing documents is normally a two-stage process: first entering the text, then editing the text. In the first stage, it's important to get those words on paper—oops, on the screen. Don't worry about syntax or style; type away, using Backspace for its conventional use, and hitting Return to mark the ends of paragraphs.
The mouse isn’t needed to enter text. Type away and let the I-bar pointer rest. The alternative is attempting to both write and edit at the same time; a confusing process that may inhibit you from creating anything longer than a memo. Force yourself to stick to the keyboard, even if you see a phrase appear that is truly wretched.

If you’re a lousy typist (and most people are), invest in a typing tutor program, then invest a few hours in learning to hit the right keys with the right fingers. Your time will be well-rewarded, and the money well-spent.

Once the text is entered, it’s time to grab the mouse. If you’re like most writers, you’ll find that most of your editing consists of deleting and re-spelling. Here’s where typing and Backspacing prove their worth. To delete words or phrases, drag over the text, then hit Backspace.


For misspellings, drag then type in the correct spelling. The first letter typed will erase the misspelled word. Or, if you merely added an additional letter to the word, drag through the offender then Backspace.

With practice, the editing process becomes amazingly fast. Until editing with a mouse, it’s possible to think that mice are only clever toys for computers. After using a mouse to speed editing, it’s agony to edit using only a keyboard.
9. Cut, Copy, and Paste
Here we are: Cut, Copy, and Paste. The good stuff. The capabilities that set Macintosh apart from other computers.

Other computers offer software that is integrated; programs where the different sections of the program communicate with other sections of the programs, in one fashion or another.

With Macintosh, every program communicates with any other program. Period. And everything communicates in exactly the same way. An awesome thought.

If software developers follow the rules, every piece of software you ever buy will easily be able to pass information to any other piece of software you ever buy.

Another touchstone to the inevitable success of Macintosh.

How will it work? Simply. All you need to know is Cutting, Copying, and Pasting. That's all there is to it.

The Glamour Trio: Cut, Copy, and Paste

The stars arrive: Cut, Copy, and Paste. Appearing on your local File menu, ready to perform stellar acts within your documents.

You probably won't use them very often, however.

In traditional word processors, the key words are block move; a block of text is first selected [often a tricky process], then moved to a new position.

Block moves and copying are what Cut, Copy, and Paste are all about. Cut deletes selected information and places it on the Clipboard. Copy places the selected information on the Clipboard without removing it from the document. Then comes Paste. Paste transfers information from the clipboard into a document.

At the risk of total redundancy:

Cut deletes information from the document and places it onto the Clipboard.

Copy copies information from the document onto the Clipboard.

Paste copies information from the Clipboard into the document.

Information Pasted into a document is always inserted at the Insertion Point. It doesn't matter where the Insertion Point is located. If the Insertion Point is within the middle of a word, and you're pasting in a sentence, the sentence will be inserted in the middle of the word—the remaining letters in the word will be pushed over to make room for the insertion, regardless of how long the inserted material is. [By the way, the noun to describe the contents of the Clipboard is Clippings. Clippings are what's found on Clipboards.]
Remember, you can only Paste *from* the Clipboard into a document—it won't work the other way around (no Pasting *onto* the Clipboard, in other words).

**Pasting Madness**

Here's a helpful exercise to become familiar with Copying and Pasting: fire up MacWrite, then type in a sentence. Any sentence, doesn't matter. When you finish, hit Return to mark the end of the paragraph.

Now drag through the sentence to select. Next, pull down the Edit menu to Copy and release the mouse button. Then click outside the selection to de-select.

Nothing happened, right? Wrong: take a look at the Clipboard, either by clicking on a sliver of the Clipboard window, if it's visible, or by choosing Show Clipboard from the menu. There's your sentence, transformed into a Clipping.

Now hide the Clipboard, either by choosing Hide Clipboard, or by clicking anywhere in the document window, which puts the Clipboard window to the rear. (Don't use menus unless you need to—it's quicker to click.)

Now the fun begins. Open the Edit menu again and drag down to Paste. Release.

You now have two sentences in the document window.

Now drag through both sentences and Copy. De-select the text by clicking outside the selection. Now Paste again. Now again drag through *all* the text and Copy, then Paste. Do it again: drag through all the text, Copy, de-select, then Paste again. Now choose Paste again. Choose Paste again. Do it again. Do it again. One more time. Now select and Copy all the text. Now Paste, Paste, Paste, Paste, Paste, Paste, Paste. (As you select larger areas of text, you'll find that the best motion to select is a diagonal slash from the upper-left to the lower-right, through the text. Practice the diagonal slash until you're comfortable with it.)

By now, you should be seeing a large amount of text. Not very interesting text, but text nonetheless. Next, continue to Paste, but this time use the mouse to select clever Insertion Points. Paste between sentences. Paste between words. Paste between letters. Really make a mess of it.

If you get too bored, try out Undo Paste, which will Undo your last Paste. If you haven't gotten completely carried away and exceeded MacWrite's capacity for text (about 10 single-spaced pages), you're now ready for even more fooling around. You've just created a mass of words with very little effort. Your reckless Copying and Pasting
MacBASIC will be well worth the wait.
should have convinced you that it's impossible to hurt anything by recklessly Copying (or Cutting) and Pasting. Don't be timid about using Cut, Copy, or Paste. At the very worst, anything Pasted in strange places can be easily deleted.

**Large Selections and Extending Selections**

You may wish to select information that encompasses more than a single screenful. No problem; it can be done in two ways:

- Drag through the text until you reach the bottom of the screen. When the pointer reaches the bottom, the text will scroll through the window automatically. If you keep your finger on the button, the text will continue to be selected and be displayed in inverse. When you've selected enough, let go.
- To select an entire document, click an insertion point before the first character of the document, drag through the text to the bottom of the screen, and hold on until the last character passes before your eyes. Let go; it's all selected.

The second method, given below, is a shortcut, and, like all shortcuts, is the easier method.

Place an Insertion Point immediately before the text you wish to select. Next scroll through the text (using the scroll arrows or the thumb) until you reach the last position of your anticipated selection. Place the pointer at that position, hold down the keyboard Shift key and click: everything from the first insertion point to the Shift-click will be instantly selected.

Shift-clicking, of course, will also work with backward scrolling through text. Try it.

**Shift-clicking Discontiguousous Objects**

To be strictly formal, when Shift-clicking in text we are using the Shift-click to select contiguous areas of a document; areas that are all connected; all one chunk, if you will.

Shift-clicking is also used to select a number of separate objects. Also known as selecting discontiguous areas. Icons, being discontiguous fellows, will serve to illustrate. Open a window full of icons and give it a try. First, click individual icons. As you click the second
and subsequent icons, the icon previously selected will be de-selected. It's easy to see that you can only select one icon at a time by clicking.

Shift-clicking works like this: first, click an icon; next, while holding down the Shift key, click a second icon. Now both icons are selected! Keep Shift-clicking away, holding down the Shift key during each icon click. All the previously selected icons will remain obediently selected.

Shift-clicking is an easy way to manipulate a number of tools or documents, all at once. For practice, Shift-click a bunch of icons, then choose Get Info from the File menu. Whiz—whiz—whiz—whiz, up come the windows. After perusal, choose Close All from the File menu to whiz the info windows back into their respective icons.

Shift-clicking is a nice convenience. As you move into new applications programs, check out the opportunities to save time and effort by Shift-clicking.

One final thought: at present, it's not possible to select a second area of a document without de-selecting the first selection. It's probably just as well: the potential for confusion with more than one selection in force are immense. Still, Macintosh will support multiple selections (discontiguous selections) in documents, if program-
mers can devise schemes where multiple selections are simple and helpful. So far, programmers haven't done this, but they may in the future.

**Clipboarding the Desk Accessories**

If you've been paying attention, you now know enough about Cutting, Copying, and Pasting to last a lifetime. Although we haven't even mentioned the Note Pad, the Scrapbook, the Calculator, or the Clock, you've probably already guessed how to take information off those Desk Accessories and into other documents: simply Cut or Copy when the Desk Accessory is the front window, then Paste into the document.

That's all there is to it. Really. As an exercise, choose Clock from the Apple menu next time you're in MacWrite. When the Clock appears, select Copy from the File menu. (Notice that you didn't have to select the information displayed on the Clock. It was enough that the Clock was the frontmost window.)

Again, nothing seems to happen. But now select Show Clipboard and there's the date and time, tucked into the Clipboard. To move the information into your text, just select an Insertion Point, then choose Paste.

Moving information for the Note Pad works just the same. The Note Pad is made up of eight Maclike pages. Clicking the upturned page corner flips to the next page. Select an Insertion Point and type away (you'll be limited to one font, style, and size, however). When you're ready to transfer the information, select by dragging, then choose Cut or Copy. After you've opened the application program you wish to work with, select an Insertion Point in the document and Paste the Note Pad text into the document.

Using the other accessories is exactly the same, with one exception: the Scrapbook. The Scrapbook is used to hold graphic selections: anything created with MacPaint or similar graphic applications programs. If effect, the Scrapbook is like a graphic Clipboard. When you use MacPaint and Cut or Copy, the information goes onto the Scrapbook, not the Clipboard.

The Scrapbook information can be Pasted into any document, but Scrapbook information can only *originally come* from a graphic document.

In practice, this limitation is an asset. In text or cell documents, it's possible to use two Clipboards: the actual Clipboard *and* the Scrapbook.
Okay: that’s how the Clipboard works. But how does the Clipboard really work? You know, down there deep in the guts of Macintosh. What's going on down there?

Do you really want to know? If so, be prepared for a discussion of bits and bytes and ROM and RAM. Hopefully, an interesting discussion, but that’s only one opinion.

Read if you will, or flip to the next chapter if you’d rather.

Still here? Okay, time for memory madness.

Inside Macintosh is a long line of memory locations. Each location consists of a group of 8 bits. Each bit can have one, and only one, state: on or off, 1 or 0.

Each group of 8 bits makes up one byte, like this:

01001100

Each byte has a specific location. The bytes are arranged in sequence, starting from 0, and proceed upward to the last, or top, byte in memory.
Imagine a ladder of bytes, stretching off into the distance:

01001100
11110010
00101100
10111110
11001100
11101100
00000000
00000000
00101100

(You may prefer to imagine the bytes marching toward, and finally over, the horizon.)

Each byte has an address. Houses have addresses, bytes have addresses. Information is no good if you can't find it.

<table>
<thead>
<tr>
<th>Address</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>01001100</td>
</tr>
<tr>
<td>0002</td>
<td>11110010</td>
</tr>
<tr>
<td>0003</td>
<td>00101100</td>
</tr>
<tr>
<td>0004</td>
<td>10111110</td>
</tr>
<tr>
<td>0005</td>
<td>11001100</td>
</tr>
<tr>
<td>0006</td>
<td>11101100</td>
</tr>
<tr>
<td>0007</td>
<td>00000000</td>
</tr>
<tr>
<td>0008</td>
<td>00000000</td>
</tr>
<tr>
<td>0009</td>
<td>11001100</td>
</tr>
<tr>
<td>000A</td>
<td>11101100</td>
</tr>
<tr>
<td>000B</td>
<td>00000000</td>
</tr>
<tr>
<td>000C</td>
<td>00000000</td>
</tr>
<tr>
<td>000D</td>
<td>00101100</td>
</tr>
</tbody>
</table>

Notice that after address number 9 (or 0009) comes address 000A. Welcome to hexadecimal arithmetic! At this point, all you need know about hex math is that it exists, and that it's a convenient way to represent bytes, which come in 8-bit chunks.

**Characters, Numbers, and Symbols**

Certain bytes mean certain things. Since a byte consists of 8 separate bits, there are 128 separate combinations of ones and zeros possible (a little math proves this correct).
With 128 combinations to work with, it’s easy to allow for representations for the lower-case alphabet, upper-case alphabet, numbers 0 through 9, and still have a bunch of byte combinations left over.

That’s what happened. The most common representation is called ASCII—pronounced as-key. The combinations left over are used for a number of helpful functions like Backspace, Start of text, End of text, Carriage return, File separator, and so on. There’s even a character to represent Null or nothing; good for killing time during, say, data transmissions over phone lines.

Appendix A shows a number of ways that type fonts can be represented by combinations of bits. Notice also that a few combinations in the tables are unused. Want to create an extension to a font? Maybe create some new symbols? That’s what the undefined combinations are there for.

Nothing But Bytes

Of course, if bytes can represent letters, numbers, and carriage returns, they can also represent other things. And they do: every piece of software now available for Macintosh, or that will ever be written in the future, are combinations of binary bytes.

Then there are programs within Mac that control what’s going on—screen display, mouse functions, reading disks, or whatever. Those programs are stored in a type of memory called ROM, for Read Only Memory. Read Only because the information stored there can’t be written over. It’s there forever, or until someone replaces the chips that hold the programs. (It’s good it can’t be written over, or we’d probably find some klutzy way to change or erase it, thereby totally crippling Mac!)

Finally, there’s data used by programs; data for programs you write, and internal program data.

### Programs and Data

Programs and data. You will find nothing else in any computer’s memory, with the exception of areas of memory that are blank and available for storage. Storage of either programs or data.

If we look at the various byte patterns in memory, it’s difficult to determine which series of bytes are programs and which were data. Fortunately, the microprocessor takes care of those details (usually—
it's possible to flub a program and have the microprocessor start munching its way through data, believing all the while that it's consuming a tasty program. Nasty results come from these situations.

**Data Structures**

Even persons with minimal computer literacy know that many different programming languages exist. What fewer people know is that a wealth of data structures also exists, and comprise a fascinating field of study.

Again, everything in memory is a series of bytes. How the com-
puter selects and interprets the bytes decides whether the bytes are programs or data.

All the above is introduction. The purpose is to prepare you for a brief overview of what happens in the Macintosh memory when tools and documents do their tricks.

**ROM**

Let's begin at the bottom: location 0. Beginning here are the programs that reside in ROM memory; unchangeable patterns electronically blasted onto memory chips. In all, over 500 separate programs, each with a specific function. Some manage windows, others pull-down menus, still others mediate between the computer and disk memory.

**RAM**

Beginning above ROM, and extending a short way beyond location 128,000, is RAM memory. RAM is volatile, changeable memory, and houses programs and data that shuffle in and out and around RAM with alarming speed. RAM contains both applications programs (tools) and the data they work with (documents and other data).

RAM use becomes interesting when we consider a few possibilities:

If tools take up too much space in RAM, there's no room left for documents. And,

Where does the information on the Clipboard go?

What if the Clipboard data is extremely large? What gets bumped off the boat?

Problems, problems.

There are three popular ways around the first problem. The first way is for programmers to make sure their programs fit; that there is room enough in RAM for both their program, documents, and for any data (text, charts, and so on) that the document contains.

Then there are other solutions. Programs in Macintosh need not reside entirely in RAM memory. Two popular schemes used by large programs are:

Virtual memory. Here, the computer is tricked into believing
that disk memory is actually RAM memory. In theory, this means that programs can fill most of RAM and also an entire disk (although we still need some room for data). The drawback is that disk memory takes longer to get than RAM memory, which slows up program execution somewhat.

Overlays. A large program can be written as modules that shuttle in and out of RAM memory. This gets tricky to program, but saves RAM space. It, too, can slow up program execution.

So much for large programs. Now, what about the Clipboard?

The Heap

Macintosh takes care of Clipboard worries with a handy data structure called a Heap. In reality (reality?) the heap is the Clipboard. The heap isn't complicated; it's merely a particular area of memory. The heap begins both small and empty. As data is entered onto the Clipboard, it grows.

The heap can be as small as one byte or as large as the amount of free space left on your disk. How large the heap becomes depends on the tool that creates it, what you Copy or Cut to the Clipboard, and the available memory left on disk.

The heap grows one thousand bytes (or 1K) at a time, as requested by the governing tool—somewhat like the The Blob, if you're young enough to remember that screen classic.

The heap is an fairly stupid area of memory. As far as programs are concerned, it's just a chunk of data. Any data can go on a heap. But when new data comes along, it erases the old data. That's why the Clipboard only contains the last information placed there—the previous contents are written-over by the new data.

Tools can't do anything with data on a heap. All tools can do is slam the contents of the heap into a document. Which document doesn't matter; after all, the heap is merely a series of bytes as far as the tool is concerned. To you and I, those bytes may represent words, numbers, or graphics; to the tools, again, the heap is just a pile of bytes.

The heap concept explains why data can be transferred between documents created by different tools. It also explains why the heap contents often can't be manipulated once they show up in a document governed by a tool different than the one that created the heap contents.
Graphic tools can work with heap contents created by other graphic tools (once they are first taken off the heap and placed in a document, of course). Text tools can work with heaps created by other text tools. Cell tools can work with heaps created by other cells tools.

Whether Clipboard (also called the scrap) contents can be manipulated by other tools, then, depends on the tool that created the Clipboard contents. Know your tools and know your heaps.

"Hey Buddy! Where do you want this heap?"

"Just set it down over there, please."

"You got it."

PLUNK.

That's the heap for you.

---

**The Consistency of Macintosh**

We've covered a lot in the last two chapters. Out of necessity, MacWrite has served as a practice ground. But these editing notions aren't confined to MacWrite; they're built into each and every Macintosh, regardless of the presence, or absence, of MacWrite or other tools.

The same editing conventions apply whenever you type characters into Macintosh. The only exception is the Clipboard and the trio of Cut, Copy, and Paste, but, in almost all cases, they too will be available.

The Insertion Point, the mouse pointer, Backspace, and the other edit mini-tools are available whenever you type on Macintosh, whether you're renaming an icon, typing a filename into a Dialog Box, or adding a comment to a Get Info box.

The best way to learn and use all the features of Macintosh is to give them a try. Don't be shy; you can't hurt anything by mousing or keyboarding around. Learn the easy methods, practice the shortcuts, and select and change with abandon. Learn and use every Macintosh feature to its fullest. That's what they're there for.

And try not to grimace when you walk by typewriters.
10. Beginning Diskology
Anatomy of a Disk

Disks (sometimes called diskettes) are made up of three parts: a circular disk, a fabric liner, and a jacket that covers and protects the disk.

Most small computers use disks that are 5 1/4 inches square. You may have heard the term 5 1/4 inch floppy; the word floppy refers to the somewhat flexible nature of the disk. 5 1/4 disks store varying amounts of information, depending on the type of disk (single or double sided, single or double density) and type of computer operating system.

With Macintosh, Apple took a more advanced approach. The Macintosh contains a 3 1/2 inch disk drive; a newer, more compact model of disk drive that offers significant advantages. Sony supplies Apple with the disk drives, which have already proved to be rugged performers in computers manufactured by Hewlett-Packard and others. The disk is encased in a rigid, plastic case. When not in the internal drive, the disk is well-protected; a metal, spring-loaded shutter covers the sensitive, inner disk where information is stored.

The disks fit in a shirt-pocket, withstand extraordinary punishment (such as being dropped or thrown on the floor), and will probably arrive intact even after suffering indignities at the hand of the U.S. Postal Service.

Have pity on all those other people, who must contend with delicate floppy disks.

Within the plastic protection is a circular disk. The disk's ability to store information results from a layer of magnetic coating applied to the disk. The coating contains particles similar to that on cassette tapes. The magnetic coating is extremely thin; 100 microinches does the trick.

Remember those high school experiments with iron filings and a magnet? The principle is the same with disks. When the original coating is applied, the particles of iron oxide are pointing in all directions. Information is stored on the disk by magnetizing portions of the disk.

At this point, all disks, large or small, plastic-encased or floppy, are the same: blank. Before use, they must be customized for a particular computer and disk drive. This process is called formatting or initialization; Apple uses the latter term.

Initializing gives information someplace to go. The scheme used to place information on disks is built on tracks and sectors. Tracks are concentric circles that cover the disk. Inside tracks (near the disk's center) have a small diameter, and those on the outside of the
disk have a larger diameter; much like individual tracks on a record album. Sectors are divisions within the tracks, like slices from a pie. Sectors are the smallest divisions found on disks; the Macintosh disk sectors can each hold 512 bytes (characters) of information.

All disks need to contain certain information; a directory of what is on the disk, for example. This, along with the track and sector markings, reduces the usable space for programs and data (but let's call them application programs and documents, as Apple prefers).

When blank, a Macintosh disk has room for about 437.5 thousand characters of information (or 437.5K bytes, since each byte translates to one character). After formatting, the capacity is reduced to about 410,000 characters—still a considerable amount of storage, and greater than that found on most other computer disks.

The next time you open an icon, you may be loading a document that resides on track 6, sectors 3 through 8. Don't concern yourself about it.

The important fact is this: initializing creates the tracks and sectors. If a disk isn't first initialized, it can't be used.

Initializing only needs to be done once. You can delete all the information on a disk (not by accident, hopefully!), then continue using the disk for other information, without needing to re-initialize. You can erase the information, but you can't erase the track and sector divisions that initializing originally laid down.

Of course, you may re-initialize a disk if you wish, which does erase absolutely everything, including track and sector markings, then lays down a new set of tracks and sectors. This can be done as often as you'd like, although you probably have better things to do.
Why won't a disk from one computer work on another? One of the reasons, other than size, is formatting. There are many ways to format a disk. Because formatting is software controlled, how a disk is formatted is determined by the computer manufacturer. Different manufacturers opt for differing numbers of tracks per disk, or larger or smaller sectors, or put needed information (such as directories) in different places on the disk.

Any disk that contains information has already been initialized. This includes disks of programs that you purchase, and any other disk that isn't totally blank.

Remember, don't initialize a disk that contains information. Initializing erases everything. The only time you need to initialize a disk is:

Before you first use the disk

or

When you wish to erase all information on a disk.

**Disk Insertion**

There are eight different ways to insert a disk into a drive. Only one is correct.

To avoid confusion, place the label side of the disk up, put your thumb on the label, and insert the disk. Notice that the side with the oval shutter goes in first, and that the shutter is on top.
Write-protecting

To write-protect a disk, slide the small red tab toward the edge of the disk. This will prevent the Macintosh from writing information onto the disk.

What disks should be write-protected? As a rule, any program you purchase should be write-protected, and any document that you don't want changed or erased (changed is merely another word for erased, isn't it?). When in doubt, protect. You can always easily change your mind later.

Okay. The disk has been made, covered with fabric, placed in a jacket, inserted in the drive, and formatted (oops, initialized). Now what?

A Peek Inside

When information is read (loaded), or written (saved) to disk, a number of things happen. First, the drive motor comes on and begins spinning the disk. When the disk is up to speed, the computer software tells a stepper motor what track to find. The stepper motor steps out to the appropriate spot, then lowers the read/write head to the surface of the disk, over the oval cutout (when the shutter is opened, the disk is inserted in the drive). The read/write head does its business, then lifts off the disk. All done.

The Apple/Sony disk has another advantage during this process: speed. Here's the secret: unlike other computers, Macintosh uses a variable-speed disk controller. Depending on where the read/write head is located over the disk, the disk motor spins at the appropriate speed: between 400 and 600 revolutions a minute. Varying the rotation speed allows maximum information to be packed onto the disk, and actually increases the chances that what you read will be exactly what you earlier wrote.

Ordinary disk drives, by comparison, spin at fixed speeds of, usually, 300 rpm and hold from 80 to 320K of information.

Will Macs Eat Disks?

If you've used other computers, you can better appreciate the simplicity of the Macintosh disk drive: no little doors to fumble with, and no chance of taking a disk out at the wrong time.
But you may have one worry: what if I can't get the disk out? What about the worst case imaginable: the machine has frozen, or locked-up with the disk firmly within its grasp? How do I get the disk out?

Easy. In the absolute worst case, it's always possible to eject the disk in this manner: turn off the Macintosh, hold the mouse button down, and, while continuing to hold the mouse button down, turn the machine on again. This always ejects the disk. After using various pre-release software packages [that crashed in various and enticing ways] the author can testify that, yes, you can always get the disk out.

**Naming Files**

Another Macintoshian convenience is the freedom given in naming documents or other files that are created, then saved on disk. The rules are few. Document names must be 255 characters or smaller in length, and the colon [:) has a special meaning.

First, the length of document names. To give an idea of just how long a 255 character name is, here's a hypothetical filename:

This filename is going to be, probably, very long; possibly even exactly—if it is done correctly, that is—the magic maximum number, which is 255 characters, but I don't think I can make... it... all... the... way!

Darn. That was only 223 characters long—we still had 32 characters to go! As you can see, it’s unlikely that you'll ever need more than 255 characters to clearly name a document, or any other file.

One caveat: just because the Macintosh software, *in theory*, lets you use 255 characters to name files, individual programs still have the option of limiting document names to a more reasonable length. Here's a good rule of thumb: when the Dialog Box appears to ask for a filename, try to keep the name within the limit given by the size of the box. You may be able to type off the end of the box (even if your keystrokes aren't visible) and still see the complete filename under your icon, after you've closed the window, but it's not guaranteed.

Try it for yourself, using different application programs.

**Volumes and Filenames**

The colon is the only special character in a document or file name. Singling out the colon for special treatment adds a great deal of flex-
ibility to systems that have more than one disk drive. When used within a filename, the colon tells Macintosh that the characters to the left of the colon represent a volume, and the characters to the right of the colon represent a filename.

Volume usually refers to particular disk drive; Macintosh disk drives are numbered, cleverly, 1 and 2. The drive within the Macintosh system unit is always referred to as drive 1.

If you have a document titled shopping_list on the disk in drive 1, the volume would be 1, and the filename would be shopping_list. To get the document from disk, the correct phraseology would be:

1 : shopping_list

Where 1 is the name of the volume (drive 1), and shopping_list is the filename.

volume : filename

That's all there is to it.

If you have only one drive, it's never necessary to give a volume name. Even with two drives, a volume identifier is only needed when referring to a drive other than drive 1 (the system unit drive).
Physical Volumes, Logical Volumes

This may seem much ado about not much. The volume: filename approach becomes truly handy, however, when hard disks enter the scene. The capacities of hard disk drives vary from 5 to 30 or more megabytes. By comparison, each Macintosh disk contains about 410 thousand bytes of information. A megabyte is one million bytes of information. A lot of storage.

A hard disk can appear to Macintosh as just another volume; disk 1 could be volume 1, disk 2 could be volume 2, and the hard disk could be volume 3. In this arrangement, the physical volumes are the same as the logical volumes: three separate drives, three separate volumes.

But you might like it otherwise. With millions of bytes thirsting for knowledge, you might find it convenient to fool Macintosh into believing that the hard disk is really four (or more) separate drives.

Why? Well, it allows easy grouping of similar files within similar volumes; it also allows you to copy files between volumes easily—even if the logical volumes are all contained within one drive.

It makes no difference to Macintosh; logical volumes seem just as physical to Macintosh as physical volumes seem to you.

This is advanced filing; remember that the volume name is never needed with only one drive: Macintosh assumes you mean the drive within the system unit unless told otherwise.

Enough about filenames?

Enough.
Section I: Operation
Section II: Software
Section III: Programming
Section IV: Hardware
Section V: Epilogue
11. Other Tools for Macintosh
What can you do with a Mac? What do you want?

Hundreds of tools will soon be available for Mac. Everything from Anthropology (learning), to Zoo Management (how-to).

But what will you do with your Macintosh?

You'll use Macintosh in these areas, if history is any guide (and it isn't, always):

- Word processing
- Data management
- Business
- Communications
- Programming
- Education
- Games

Here's part of what to expect, in each category.

**Word Processing**

Macintosh is a phenomenal writer's tool. It leaves other computers in the dust when it comes to processing words.

With Macintosh, as you've probably noticed, what you see on the screen is exactly what you get on paper. That includes a variety of type fonts, styles, and sizes. All with no tricky, hard to remember commands.

After learning how to process words with MacWrite, or another Macintosh word processor, you'll cringe at the sight of an ordinary typewriter. Or an ordinary computer word processor.

Word processing is one of the most liberating uses for computers. If you don't like what you've just written, change it. If a block of text looks better somewhere else, move it. If your spelling is lousy, run your text through a program that checks (and corrects!) your spelling. If your grammar is poor, run your text through a program that checks your grammar, then gives you preferred alternatives (yes, these tools will soon available for Macintosh).

The ramifications of Macintosh word processing are almost endless. The art of writing, known by all but mastered by few, is suddenly accessible, even easy. Your relatives may be inundated by letters that, before, you used to put off.
If computers do nothing else, they may help spur the practice of letter writing.

But don't stop there. In years past, the pen was mightier than the sword—if you happened to own a printing press. And typesetting equipment, and camera equipment, and paste-up equipment, and . . .

No more. With Macintosh, you have the equivalent, on your desk, of a newspaper production room, fully staffed and at your bidding.

Don't let your imagination stop at letters and reports. With Macintosh, you can easily produce newsletters, posters, flyers, brochures, magazines, even entire books! You'll still need a print shop for the dirty work (unless you only require a few copies), but Macintosh can produce camera-ready copy that your printer will relish.

Further Reading, at the end of this book, offers pointers toward books that can get you started with sophisticated publishing applications.

Oddly enough, there's some debate over whether word processing is good or bad. Some say that every fool in the country [present company excepted, of course] will churn out reams of garbage. Others contend that good writers will become better, and poor writers may become at least readable.

Whichever, it's safe to say that you'll write with Mac, if you don't already.

---

**Data Management**

Data management covers a range of applications from the trivial to the complex; from a few recipes, to thousands of personnel records; from Christmas card lists, to business inventories that require cross-indexing, sorting, merging, totals, sub-totals, and sub-sub-totals.

Computers are well-suited to store, display, sort, find, list, and print records, whatever the contents of those records may be. And everybody has records; insurance records, tax records, household budget records, and records of record collections.

Expect to see many data management, or database, tools for Macintosh. Expect to see prices that range from contribution suggested to many hundreds of dollars. Expect a difficult decision over which program to buy.

When choosing a database program, or any program, ask yourself and your dealer these questions:

- How much information will I need to store?
• How extensively will I need to manipulate the information? Will I require totals, sub-totals, etc.?

• How closely does the program adhere to Macintosh User Interface conventions? All Apple-supplied Macintosh programs work the same way; mouse clicks mean one thing, double-clicks another. Pull-down menus, scroll bars, Close Boxes, Dialog Boxes—everything is consistent from application to application.

Apple has already written large portions of the tools that outside programmers, in the past, had to write themselves: graphics packages, text editing routines, menu routines, and much more.

When Apple made the Macintosh secrets available to outside software developers, they made one request: please do things our way. We developed the tools, you use them, but use them in a manner that presents the end user (us) with clear, consistent, and easy ways to manipulate information.

If a software package abuses this golden (and free) opportunity, and confronts you with a confusing or old-fashioned way of doing
things, don't put up with it! Cast your economic vote and don't buy the software. At the very least, write the manufacturer a nasty letter. Among other things, Macintosh excels at producing letters. Use a 24-point bold font—that ought to convey your feelings rather nicely. Other software determinations include:

* Is the manual readable?
* How much do I want to spend? Am I buying more capabilities than I need?
* Have I read reviews about the program? If not, there may be a reason.

Finally, for all programs (particularly expensive programs), get a demonstration. Don't be content, however, to watch a salesperson put Wondersoft through its paces. Sit down and try to run the program yourself.

These suggestions apply to all software purchases, but especially when the prices are high and the subject is:

### Business

The Macintosh wasn't designed to be a business computer, whatever that is. But Macs will surely proliferate in offices, if only because Macintosh is a fast, flexible computer.

Originally, Apple Inc. didn't believe that businesspeople would be interested in Macintosh. As Macintosh was being designed, Apple showed its latest wonder to groups of people selected at random; Apple calls these Focus Groups. The intent is to gather reactions; what people like and don't like about a particular product.

People liked Mac. People in business, it turned out, loved Mac. They weren't necessarily supposed to, but they did. A bit of serendipity there for Apple.

The consequence of this reaction will be a raft of business-related programs for Macintosh, many of them available when Macintosh is first released. The most popular will likely be spreadsheet programs, database programs, and programs that create graphs or charts from spreadsheets or databases.

VisiCalc (for VISible CALCulator) was the original spreadsheet program. The idea behind VisiCalc is simple: you enter numbers, and the calculations that produce the numbers, into a worksheet displayed on-screen. Once entered, the numbers can be manipulated in an astounding variety of ways.
Say you’re pondering purchase of a duplex. You have expenses, gross rents, taxes, and so on. Each part of the income and expense will vary over time; some will rise and some will fall. If your projection is for 20 years, a great deal of pencil scratching is required to determine if the investment is sound.

And what if the cost of heat rises by 20% a year, instead of the 10% that you’ve already calculated? With a spreadsheet program, you simply change the formula to increase by 20% a year and—Voila: 20 years of numbers change before your eyes.

If you pay your accountant by the hour, let’s hope he and she uses an electronic spreadsheet.

Communications

It’s a big world out there. Computer bulletin boards, computerized shopping, consumer databases, outfits like The Source and Compuserve that offer a variety of services, even dial-up computer ser-
services that aim to match you up with members of the opposite sex.

Hook up a modem to Mac, and your life may never be the same again. Your phone bill may never be the same, either.

A modem is a device that allows you to send, or receive, information through phone lines. The standard speed for modems is 300 baud. Faster, 1200-baud modems are becoming common, but usually cost about double the price of their slower cousins (though lots of people are waiting for prices to fall). Have your local computer dealer demonstrate both the 300 and 1200 baud modems, then decide for yourself if the extra speed is worth the extra cost. A further consideration is the fact that many on-line services charge higher rates per hour for 1200-baud users.

Be prepared to spend $80–$600 for a modem, $0.00 to $300 for a communications program (yes, some are free), and $30–$100 for initial subscriptions to for-profit services such as Compuserve (which, like The Source, also charge a per-hour fee, which varies depending on the time of day and other factors).

Ever seen the boating wall plaques that read: "A boat is a hole in the water into which you pour money"? Whoever wrote those words should be introduced to computers. Not that you need to spend enormous amounts of after-purchase money, it's just that the temptations are nearly irresistible. And they almost all cost money.

Programming

See Chapter 17, or one of the many, many books on computer programming. Some of those books are listed in Further Reading.

Educational Opportunities

Computerized home education is popular, but not as popular as many once thought it would be.

Why? A few reasons. Programming languages, with some exceptions, aren't geared for creating the drill and practice approach often taken with computers. Also, until now, most computers didn't have enough memory or speed to teach complex, non-trivial areas of knowledge. And fewer people may desire home education programs than was once thought.

Still, a variety of educational programs are available. The widest selection are written for the Apple II/Ile. The most popular programs
Educational Opportunities / Does Anyone Play Games!

are those that teach use of the computer, or how to program in BASIC. Programs that teach the Russian language or genetics can be found, but the competition between manufacturers is those areas is, well, cool.

The Macintosh may change all that. Mac has the speed and storage necessary to guide students through the lengthy and winding paths that make up learning. Look for pre-school and elementary programs to appear first; they’re relatively easy to program and have a proven market. They’re also effective teaching aids for small children.

If you’d like to use Mac to make a million, you could do worse than investigating educational software. Remember, you heard it here.

Does Anyone Play Games?

The Macintosh will introduce a new level of computer games. The tools that come with Mac are an ample demonstration of what’s possible. The best is certainly yet to come.
Mac's prodigious game-playing ability stems from the same reasons that make it suited to database management and other serious pursuits: the Macintosh is very fast, and Mac's internal software operating system is extremely powerful and flexible.

Is playing games reason enough to buy a Mac? Why not? Adults buy lots of other silly toys, don't they? We, of course, have better things to do than play games, right? Right?

**The Magic Program**

If you're a typical user, you'll use Mac for writing, filing, business, education, programming, communications, and games.

"Will" is not "should," however. What you *should* use Mac for may be entirely different.

VisiCalc, more than any other program, helped fuel the sales of small computers. Suddenly, small computers weren't merely helpful in business, they were almost mandatory, particularly to people who work with numbers, and especially projections, for a living.

VisiCalc was first made available for the Apple and probably caused the sale of thousands of Apples. Electronic spreadsheets, it can be argued, are the generic necessary program for business.

So what's the necessary program for the home?

Nobody knows. Maybe there isn't one. Maybe there won't be one program, alone, that makes purchasing a home computer a required duty. Many people believe that the combination of uses listed above are reason enough.

But maybe not. There's a good chance that a philosopher's stone of home computer programs will, indeed, be discovered. Lots of firms are frantically searching for that *one program* (whatever it is) that will make their company's fortune, and cause sales of home computers to explode in a manner never before seen.

Some of the doubters say the search is misguided. They say the ultimate home computer isn't a computer at all—it's a Home Robot, a machine that will clean the house, do the dishes, make the bed, and put out the cat.

A great idea, but easier said than done.

No matter what comes along, it seems safe to say that your uses for Macintosh will result from who you are.

After all, computers don't think; computers are simply fancy machines that, like other machines, have absolutely no values. If you decide to use your Macintosh to help you squeeze every last penny from your renters, fine—Mac won't lift an eyebrow. If you're
a banker and think that Mac will keep track of foreclosures, you're right. Some people may need a computer to keep track of how much poison they're spreading on fields. Mac can do it.

Personal computers have been called lots of things. Tools for Modern Times, Brain amplifiers, Electronic levers, and so forth.

Action amplifiers is also a good description. What you're doing may be right, wrong, or inconsequential, but doing it with a computer will probably magnify the results of your actions.

Nobody yet knows the effects of personal computers. But hundreds of thousands of people now have action amplifiers, so it's safe to say that computers will have an effect of some sort.

If Socrates were alive today, he might conclude that "The unexamined use of computers is unforgivable."

If he was, and he did, would he be right?

You decide.
Illustrious Career and Heroic Deeds of Colonel Roosevelt

"The Intellectual Giant"

Containing a full account of his marvelous career, his early life, adventure on a western ranch among the cowboys, famous leader of the Rough Riders: President of our great country; his wise statesmanship, manly courage, patriotism, etc., etc.

Including his famous adventures in the wilds of Africa in search of lions, rhinoceri, elephants and other fercious beasts of the jungle and plain; journeys in unknown lands and marvelous discoveries, together with his triumphal journey and receptions by the crowned heads of Europe

By Jay Henry Mowbray, Ph.D., LL.D.
The well-known historian and traveler

Embellished with a great number of superb phototype engravings

Leslie-Judge Co.
New York
The First Word Processor

MacWrite is the first word processor available for Macintosh and an excellent demonstration of Mac's capabilities.

MacWrite rigorously adheres to all the Macintosh software design guidelines. And it should: MacWrite is an Apple product that was created in step with the creation of other Macintosh software.

If you've used any of the other Mac programs, you'll master MacWrite in minutes. All the editing functions found in Macintosh, and detailed earlier in this book, are found in MacWrite along with a number of more advanced features.

Program operation is quick and fool-proof; as always, printing is quick, easy, and amazingly faithful to what you see on the Macintosh screen.

Still, it's necessary to ask: is MacWrite the right word processor for you?

If your needs are average, or typical, it probably is; MacWrite has many good features, is extremely easy-to-use, and, maybe best of all, follows all the rules for how Macintosh programs should behave.

For most people, MacWrite delivers more than enough features. Some buyers, however, will need more power than MacWrite can deliver.

That's okay: MacWrite wasn't designed to be all things to all people. The goal of the MacWrite designers was to produce a modest program that could be mastered quickly, used dependably, and would incorporate some of the most requested features of other popular word processors, all in a fully Macintosh style.

The major MacWrite drawback (and it might not be a drawback to you) is a limited document length: 10 single-spaced pages, or 20 double-spaced pages. If you don't expect to type anything longer than 10 pages, have at it.

To be fair, most people don't create documents longer than 10 pages. To be fairer still, it's always possible to break longer works into sections, or chapters, and then edit and print each separately; often a good idea, even when not forced upon you. And MacWrite will begin numbering pages starting with whatever number you specify, so sectionalizing is easily done.

Features

Here are, first, some of MacWrite's features, followed by some of the features not included in MacWrite. Keep in mind that featurism isn't
always good; too many features become confusing, and features that are hard to use, hard to remember, or unpredictable in action are usually never used at all; they just hang around, often making the entire program slower, bigger, and sometimes buggier.

With that in mind, we have:

**MacWrite Text Features:**

- Nine type fonts, completely mixable.
- Five type sizes: 9, 12, 14, 18, and 24 point.
- Six style options: Plain, bold, italic, underline, outline, and shadow. The latter five are mixable and may be used in combination; italic and bold, say, or outlined and shadowed and bold and italic.
- An Undo typing feature to retrieve deleted text.
- Four text formats: flush left, flush right, fully justified, and centered.
- Three spacing options: single, double, or triple.
- Easy insertion of additional rulers into text, to control justification, spacing, tabs, or other format characteristics.
- Multi-line headers and footers, which may contain page numbers, date, and current time.
- Decimal tabs, which aligns columns of numbers with respect to decimal points.
- A maximum of 10 user-set tabs per ruler.
- Word wrap, which makes use of the Return key necessary only to mark the end of a paragraph.
- Flexible search and replace with various options.
- Full and easy use of the Macintosh Clipboard.

And then we have:

**MacWrite Printing Features:**

- Choice of four paper sizes, including legal size and 12” European.
NUMBER AND PROPORTION

EUCLID

FIRST PRINCIPLES

DEFINITIONS:

1. A unit is that in virtue of which each of the things that exist is called one.
2. A number is a multitude composed of units.
3. A number is a part of a number, the less of the greater, when it measures the greater.
4. But parts, when it does not measure it.
5. The greater number is a multiple of less when it is measured by the less.
6. An even number is one that is divisible into two equal parts.
7. An odd number is one that is not divisible into two equal parts, or that differs from an even number by a unit.
8. An even-times even number is one that is measured by an even number according to an even number.
9. An even-times odd number is one that is measured by an even number according to an odd number.
10. An odd-times even number is one that is measured by an odd number according to an odd number.

- Tall or wide paper orientation.
- Choice of connected, perforated fanfold or single-sheet paper.
- Three choices of print quality/printing speeds: high resolution (high quality, slow), standard resolution (some of both), or draft (blazing speed, only one type font and size, no italics).
- Optional printing of a range of pages within document; from, say, page 3 to page 7.
- Printing of multiple copies of a single document.
Enough, Enough!

By now, you may be thinking: this program doesn't have enough features? What more could anyone want?

Actually, full-time writers, secretaries, and even many students need features that MacWrite doesn't offer.

These days, the microcomputer word processing market is awash with feature-laden programs. Many of those programs have astronomical sales—either the result of canny marketing, or the result of consumers who do, indeed, want or need a wealth of features. Or both.

Here, in no particular order, are some of the features found in some of the other word processors, and absent in MacWrite:

- Variable number of lines-per-inch.
- Virtual filing, which makes document size dependent on storage available on disk, rather than room available in user memory.
- Split screen capability. Allowed for in the Macintosh User Interface, this feature would permit windows to be split, either vertically or horizontally. The windows would then display different portions of the same document, or, better still, two separate documents, with the ability to cut and paste between each.
- Integral electronic spelling checker. (And how about a thesaurus, a word counter, a punctuation checker, a phrase checker, and a word frequency-of-use counter? They're here, now, for other computers. They will all be available for Macintosh, eventually, but possibly not all these functions will be integrated into word processors—at least not right away.)
- A lexicon of commonly used words or phrases that could be inserted into the text with one or two keystrokes. Invaluable for boilerplate documents.
- Superscripts (words/numbers printed a half-line above text) and subscripts (a half-line below).
- Automatic back-up capability.
- True footnote capability, rather than page number/date footings.
- Printer selection, to allow use of printers other than the ImageWriter. In particular, provisions for letter-quality printers.
- Printer spooling, which allows a document to be edited while another document is being printed.
Enough, Enough!

**Know Your Needs**

Some of those features are extremely desirable to some people. Some would probably never be used at all by most people. The reason to bring up the subject in the first place is this: unlike MacPaint, which will have no competitors, MacWrite is only the first of many word

---

**XXIV**

Mine eye hath play'd the painter, and hath stell'd
Thy beauty's form in table of my heart;
My body is the frame wherein 'tis held,
And perspective it is best painter's art.
For through the painter must you see his skill,
To find where your true image pictur'd lies,
Which in my bosom's shop is hanging still,
That hath his windows glazed with thine eyes.
Now see what good turns eyes for eyes have done:
Mine eyes have drawn thy shape, and thine for me
Are windows to my breast, where-through
Delights to peep, to gaze therein on thee;
Yet eyes this cunning want to grace their art,
They draw but what they see, know not the heart.
processors for Macintosh; some more expensive, some possibly less expensive. And many, certainly, with more features.

If processing words is your primary use for Macintosh, it's worth taking time to carefully determine which features you need. Find out what's available, compare, and buy only after careful consideration. You'll have to live with your decision and we want you as happy as possible. If you're like most people, though, you may find that no word processor perfectly fills your needs. The day when users can easily create completely custom software is still in the future. But with Macintosh—custom publishing has arrived.

**Bring on the Waxers!**

The invention of movable type transformed western civilization. The invention of Macintosh will have less profound effects, despite the claims of Apple Computer, Inc. But one effect of Macintosh may be far-reaching: the ability for individuals to generate camera-ready copy, easily, quickly, professionally, and inexpensively.

This is power to the people in the best sense of the phrase. Until now, freedom of the press was usually confined to those who could afford a printing press. With Macintosh, the print shop comes home, at last.

The traditional cycle of creating newspapers, magazines, or books goes something like this: first, the copy is written; usually on a typewriter, but now frequently on a traditional word processor of some sort.

After editing, sizes and styles of type are chosen, usually by an editor: where are the heads, how large is the body type, what font will be used?

Next comes retyping; the text is entered into to an expensive typesetting machine with all the necessary codes that mark where and how all the various sizes, styles, and attributes will appear. One code for boldface, another for italic, and so on. (A primary requirement here is a top-notch typist; the men and women who sit behind these keyboards are worth every penny of their salary, and more.)

The output of that typesetter is usually stored on a magnetic disk. It is then fed into a large [and, again, expensive] machine that transforms the data into actual galleys: words and headlines that look just as they will when the final production is in your hands.

Next comes a run through the waxer, which coats the back of the galley repro with wax. The sheets of words are then pasted onto mock-ups of pages [no crooked lines, please . . .], and then shot by a large camera that produces negatives of the entire page.
1061 - FILLET OF BEEF MODERN STYLE

Filet de Boeuf Moderne

Lard the fillet alternately with bacon and tongue, and poele it.
Glaze it just before serving; set it on a long dish, and surround it with a garnish as follows:—On either side of the fillet lay a row of small ‘chartreuses,’ made in small, hexagonal moulds.
To make these ‘chartreuses,’ butter the moulds and decorate the bottom of each with a slice of truffle, large enough to almost entirely cover it. Now line the side of the mould with various vegetables, such as carrots, turnips, peas, and string beans; each vegetable should be cooked as its nature requires.

The negative, after insertion of photos and other art, is burned onto a metal plate, which then, at last, is affixed to a printing press for final production of pages.
Which may be more than you wanted to know about printing.

The Point

But it’s easy to see that Macintosh does away with many of those early production steps. In effect, Mac is perfectly capable of producing camera-ready copy, ready to be photographed and printed.
MacWrite can easily produce a variety of printed productions: pamphlets, brochures, newsletters, and flyers. Created with all the tools of MacWrite: striking headlines, smaller body type, captions, and cutlines for photos or artwork.
The artwork can arrive via MacPaint and the Scrapbook, or from the Note Pad, or freehand additions after printing with ImageWriter, or snipped from another source. Holes can be left for later additions of photos.
Take your planned publication to a Quick-Print shop if photos or more extensive steps aren't required, or to a job printer for more elaborate productions.

The results won't be as precise as conventional newspapers or spare-no-expense brochures, but will match and exceed all of the do-it-at-home creations made with typewriters or other computers (with the single exception of Lisa).

**The Marriage of Mac**

Yet to come is the marriage of Macintosh with those large typesetting machines we mentioned earlier. That connection has already made between Lisa and Compugraphic, one of the largest manufacturers of typesetting equipment. The result enables a single person to write copy, determine format, check appearance, then blast the completed word directly to a typesetting machine, which obediently produces galleys ready to be run through the waxer and slapped on a page.

The thought gives goose-bumps to writers.

The uses of all this technology are staggering. From your spare bedroom at home, it's possible to create everything from business cards to encyclopedias, and everything in between.

**MacWrite's Place**

MacWrite lacks some of the sophistication needed for total control of text. In particular, column moves, proportional justification, and hyphenation help capabilities would be mandatory for many applications. But word processors with these features are on the way, and other word processors, designed exclusively for typesetting, are probably being written this minute.

Which leads us to the question that many people still ask: "What will these personal computers really do, anyway?"

One thing that Macintosh will surely do is help launch an armada of do-it-yourself publishing ventures.

If you'd like to come along, check out some of the books in Further Reading, at the back of this book. Unfortunately, the book *Computers and Typesetting: Home-Publishing in the Electronic Age* hasn't yet been written.

Does that give you any ideas?
A Student complained to the Master that he could find no work. The guru said:

"Do not hold that destructive thought. As a part of the universe, you have an essential place in it. If necessary, shake up the world to find your work! Don't give up and you will succeed."

Excerpts illustrating Macintosh type fonts and styles were taken from the following books:

The Illustrious Career of Col. Roosevelt

The Portable Greek Reader
W. H. Auden, ed., The Viking Press, New York, MCMXLVIII

The Complete Works of William Shakespeare

The Escoffier Cook Book: A guide to the Fine Art of Cookery
A. Escoffier, Crown Publishers, New York, 1941

Sayings of Yogananda
Paramahansa Yogananda, Self-Realization Fellowship, Los Angeles, 1968
13.
A Few Lines on MacPaint
What Is It?

MacPaint is one of the most amazing programs ever written for a microcomputer—or any computer, for that matter.

MacPaint will delight children for hours on end; it will fascinate and enthrall adults, even those who profess to dislike computers. It will stun those with even a smattering of knowledge about how computer graphics work. And it will provide a powerful set of graphic tools for people who would rather work than play.

MacPaint lets you perform graphic feats you would have never thought possible: drawing, painting, typing, moving, changing, spray-painting, lassoing, and more.

What's It Good For?

MacPaint is valuable for a number of reasons.

It is the ideal first program for people who know nothing about computers. With MacPaint, it's impossible to do anything wrong; everything behaves as a naive user would expect. What surprises there are in MacPaint (and there are many) are all thoroughly pleasant, intuitive, and on reflection, not surprising at all.

MacPaint is also the ideal rainy day program for kids of all ages. If there is a single program that will force you to buy yet another Macintosh, this is it: kids love it, can figure most everything out for themselves, and will happily sit before the screen for hours, laughing and painting [not to mention the benefits of better small-motor coordination and precise thinking.]

MacPaint gives everyone needed practice in mouse maneuvers: clicking, double-clicking, dragging, and all the other necessary Macintoshian skills.

Working with MacPaint also gives insight into how all of Macintosh works; it is a graphic primer on this most graphic of all computers: the selection rectangle will be remembered from its incarnation in the finder, where it also selects a number of objects for manipulation. The tool icons at the left of display are just more icons; pick an icon; then use it.

MacPaint, for the first time, brings Art and Computers together for people who don't have a fortune to spend on computing power. This long-overdue occurrence will hopefully begin to humanize the often forbidding binary world.

For business, MacPaint offers a range of tools: the ability to create
forms, quickly and inexpensively; fast, presentation-quality charts. Using the Scrapbook allows one to insert graphic information into documents of different types: text, spreadsheet, database, or any other document created with Macintosh.

Which isn't to say that MacPaint is not a toy. It is; one of the best toys ever created.

But it will also get you where you're going.

The art of Apple's Susan Kare, scattered throughout this book, is testimony to the range of MacPaint (and, more importantly, to the talent of Ms. Kare). Whimsy, depiction, covers, flyers, and business forms—with MacPaint, even less talented hands can create works that both please and provide function.

**How's It Work?**

It's a secret.

The sole author of MacPaint is Bill Atkinson, who is also the sole author of the QuickDraw graphics routines that underly everything presented on the Macintosh screen. In a sense, MacPaint *is* QuickDraw. Writ large, if you will.

What's so great about QuickDraw? Speed and flexibility; until Atkinson dreamed it up, no one knew how to do sophisticated computer graphics this precisely, and this fast.

**What Can It Do?**

On a higher, and less secret level, MacPaint works like this: pick a drawing tool, pick a pattern, and create.
Optionally, a number of Goodies are available for special effects, or to aid in creating certain designs.
The basic drawing tools are these:

- A paint brush, which paints using a number of brush shapes,
- A paint sprayer, which behaves as you’d expect, but also sprays in patterns, harder to find in real life,
- A pencil, which draws pencil-like lines in black or white,
- An eraser, which will bring back memories of erasing the board, and
- A paint bucket. As the name implies, the bucket will fill the document, or any enclosed portion, with the paint of your choice.

**MAC ROCK PARTY DIRECTIONS:**

![Map of directions to the event]

**LOS GATOS SWTH AND RACQUET CLUB**
14788 OSA ROAD
LOS GATOS

**SATURDAY**
OCTOBER 8TH
8PM TO MIDNIGHT

**EITHER WAY, IT'S STILL A MAC ROCK PARTY!!**

DON'T MISS THE FUN!!!!
On a still higher conceptual level are:

- The ability to insert text, in a number of different fonts, styles, and sizes, anywhere within the MacPaint document.
- A selection rectangle and a selection lasso, which allow you to snare any part of the document for further, marvelous changes.
- A hand, which grabs the portion of the document seen through MacPaint's window and moves it to a new location. The hand is delightful.

Then there are pre-defined shape and line templates [for lack of a better word]:

- Straight lines, at any angle,
- Hollow or filled rectangles (squares, of course, are also rectangles),

Anatomy of the American Bullfrog

- vomerine teeth
- eyeball
- brain
- forearm muscles
- backbone
- peritoneum
- Ilium
- cloaca
- thigh bone
- ankle bone
- web
• Hollow or filled rounded-corner rectangles,
• Hollow or filled ovals (circles, of course...),
• Freehand lines, hollow or filled, and
• Hollow or filled irregular polygons (here defined as any figure made up of a number of straight lines at various angles).

There's more. Borders can be set to five widths. If you can't find the pattern you like among the 38 choices, it's possible to quickly design...
your own pattern. A Grid aid allows you to easily line up text or figures. Another aid, Brush Mirrors, gives you a choice of kaleidoscopic ways to draw with up to four brushes at once, all in perfect mirrored unison.

Then there's the famous Fat Bits: the same square guys that appear in the leftmost desktop pattern edit window of the Control Panel. But first: what's a bit and why is it fat?

Bits are both the individual dots that make up the Macintosh screen display, and are also bits of memory within Macintosh. Because Macintosh is always in graphics mode, what you see on the screen is an exact replica of a portion of internal memory. When you look at the display, you're looking, in a sense, directly into Macintosh memory. That memory is made of bytes of information, and each byte is composed of individual bits. Those individual bits are also individual screen bits, dots, or pixels.

Fat Bits provide a blow-up of any chosen section of the MacPaint document (read: internal memory). Once enlarged, the individual pixels are at your mercy: click them one-by-one to change them from black to white [or vice-versa], or drag through them to effect sweeping changes in miniature. Other tools and aids also work in Fat Bits: the Hand is particularly useful on this level. Click the Hand, then grab the portion of the page shown in the MacPaint window and move it to a new position on the page.

Still another aid, Show Page, lets you look at a reduced version of
the complete 8 1/2 by 11 inch page—a precise way to see exactly how your work will look when printed. [Double-clicking the Hand is a shortcut way of showing the entire page.]

When your masterpiece is complete, printing is as simple as dragging down the File menu to Print and letting go of the button.

What'll It Do Next?

Are you getting tired of: "If you think this is great, JUST WAIT!!!!!!" It's the truth, though, and nowhere more true than in MacPaint. The marvelous MacPaint tools are only the tip of the ROM, so to say. Buried with Macintosh is the software that controls much of MacPaint, and much, much more.

All software designers need do is place a front end on a powerful graphics machine. The hard part's been done already.

Look for graphics programs tailored to individual professions. Maybe even tailoring. But certainly architecture, landscaping, electronic circuit design, and other fields that rely on visual design, arrangement, and graphic selection.

Not everyone works with words. Nor even with numbers.

But enough of these boring bookish words: here's more of how it looks up on the Macintosh screen:

Screen 13.1 MacPaint Introduction Screen
Screen 13.2  MacPaint Shortcuts

Screen 13.3  MacPaint Patterns in Ovals and Round Rectangles
Screen 13.4  MacPaint Rendition of Pantheon

Plan of the Pyramids at Giza

Screen 13.5  MacPaint Pyramids
14. Microsoft Wares
Multiple Microsoft Programs

Multiplan is a sophisticated, extensive, easy-to-use electronic spreadsheet. If it can be done with numbers, whether juggling, forecasting, budgeting, or quantifying, it can probably be done—and done quickly and well—with Multiplan. It also does text.

But it doesn't do graphs.

Multiplan is only one in a series of Multi programs from Microsoft. On other computers, the Multi programs (only two so far, but more to come) all share a common user interface: if you know one program, you pretty much know them all. Sounds a tad like Mac, doesn't it?

Other computers also have a version of Multiplan. One of Multiplan's failings, however, is the inability to do graphs. That failing is rectified on Macintosh with another Microsoft Multi offering: Chart.

Plan and Chart work closely together on Macintosh; for numbers, use Plan; to see the numbers in a variety of charts, use Chart. Or, use Chart alone, for quick graphs: from simple to complex. Chart will also be available for other computers, but never in the elegant form it assumes on the Macintosh.

And more from Microsoft is on the way. Microsoft Word, an extremely powerful word processing program, was being readied as this book was written. Powerful, in CompuTalk, means the program can do many things, do them quickly, and do them with little effort from the user's side of the screen.

When released, Word may be the most powerful word processor available for Macintosh.

And there's more. Microsoft was the first company to write a version of the BASIC language for small computers. Microsoft BASIC is now a standard in the field. Expect at least one version of Microsoft BASIC for Macintosh.

Also rumored to be under development for Macintosh is File: a data base management program. Details are skimpy, but Microsoft's reputation for quality software is well-known. Expect an admirable data base program.

As Macintosh sales increase, Microsoft will certainly open the gates and release versions of all their application programs, languages, and entertainment software for Macintosh. This would include the languages Pascal, Fortran, COBOL (please, not COBOL!), Assembler, C, the entertainment programs Adventure, Olympic Decathlon, and the famous Flight Simulator: a program that puts you in
the cockpit of a Cessna 180 surrounded by a 3-dimensional world to fly within.

For now, Mac owners have Multiplan, a welcome addition to the growing library of Macintosh software.

A Closer Look

Let's begin our look at Multiplan with a paragraph straight out of the Multiplan owner's manual [with thanks to Microsoft for use of their prose].

Under the title *Why use an electronic worksheet*, we find:

"Multiplan is more versatile than a paper worksheet—how you arrange it is entirely under your control. So much so, in fact, that you can restructure the worksheet by adding, changing, and deleting sections, and changing the width of columns as you work. You can copy all or parts of one worksheet onto another worksheet without the tedium of rewriting or retyping each entry. Multiplan is also a calculator—a very sophisticated one. Once you set up your worksheet, Multiplan does all the calculations automatically, exactly the way you tell it to. This lets you concentrate on thinking about the numbers rather than on calculating and entering them. And you can experiment to see the effects of different numbers. Just enter new numbers—Multiplan recalculates the entire worksheet for you."

That paragraph should be enough to convince anyone who's ever worked with paper worksheets that the computer age is here, and long overdue.

Insert the Multiplan disk, click the proper icon, and you're confronted with the worksheet—a portion of it, anyway. The initial display shows the upper left portion of the sheet: 6 columns wide, and 18 rows down the screen. Each cell within the worksheet grid can hold characters or numbers and be manipulated separately, or in accordance with the contents of other cells.

The worksheet has a maximum size of 255 cells horizontally, and 63 cells vertically; a very large worksheet. But even that massive size might not be enough for some users with complex needs. For those people, Microsoft has a solution: Multiplan worksheets can be linked to one another; a group of cells in one sheet are copied to cells in another sheet. When a worksheet is loaded from disk, it checks for changes in other linked sheets and updates the present work-
The worksheet window may also be split into two panes. Each pane can then be scrolled independently, for two views into two areas of the worksheet.

The black cell shown in Screen 14.0 is the current cell. Anything next entered into the worksheet will go into that cell. The current cell is selected by one click of the mouse.

The other Macintosh movements are also represented in Multiplan. Dragging is used to select a group of cells for manipulation. Menus are available and extensively used. Shift-clicking gives access to additional features.

Giving orders and making changes are as simple and intuitive as the Multiplan's designers could make them. Odds are, you'll feel right at home.

Functions

The contents of cells, and the changes they undergo, are guided by user-created formulas. Formulas, in many cases, are only as good as the functions within them. Multiplan supplies a large set of pre-defined functions for inclusion in your formulas.

Each function takes one or more arguments to work on. For exam-
ple, the function INT (for INTeger) returns the largest integer number that is equal or greater to the argument. INT(7.9) then, is equal to 7.

In the list below, the name of the function is given first, followed, in parentheses, by the type of argument the function requires, and a brief explanation. Glancing through the list should give you a good idea of Multiplan's power, and its potential uses.

<table>
<thead>
<tr>
<th>Mathematic functions:</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS[number]</td>
<td>Absolute value of a number</td>
</tr>
<tr>
<td>AVERAGE[list]</td>
<td>Average of the values in a list</td>
</tr>
<tr>
<td>COUNT[list]</td>
<td>Count the values in a list</td>
</tr>
<tr>
<td>EXP[number]</td>
<td>Exponentiation</td>
</tr>
<tr>
<td>INT[number]</td>
<td>Integer portion of a number</td>
</tr>
<tr>
<td>LN[number]</td>
<td>Logarithm, natural</td>
</tr>
<tr>
<td>LOG10[number]</td>
<td>Logarithm, base 10</td>
</tr>
<tr>
<td>MAX[list]</td>
<td>Maximum value in list</td>
</tr>
<tr>
<td>MIN[list]</td>
<td>Minimum value in list</td>
</tr>
<tr>
<td>MOD[number to divide, number to divide by]</td>
<td>Remainder after division</td>
</tr>
<tr>
<td>NPV(rate,list)</td>
<td>Net present value of list at rate</td>
</tr>
<tr>
<td>ROUND[number, number of digits]</td>
<td>Round number to so many digits</td>
</tr>
<tr>
<td>Sign</td>
<td>Sign</td>
</tr>
<tr>
<td>Square root</td>
<td>Square root</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>Standard deviation</td>
</tr>
<tr>
<td>Sum of numbers in list</td>
<td>Sum of numbers in list</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trigonometric functions:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ATAN[number]</td>
<td>Arctangent</td>
</tr>
<tr>
<td>COS[number]</td>
<td>Cosine</td>
</tr>
<tr>
<td>SIN[number]</td>
<td>Sine</td>
</tr>
<tr>
<td>TAN[number]</td>
<td>Tangent</td>
</tr>
</tbody>
</table>
Logical functions:

IF(logical-expression, value-if-true, value-if-false) - Chooses which value to give, depending on the logical value of the expression.
AND(list) - True if all values in list are true.
NOT(logical-expression) - True if false, false if true.
OR(list) - True if any value in list is true.
ISERROR(value) - True if value is any error value.
ISNA(value) - True if value is #N/A.
FALSE() - False.
TRUE() - True.

Functions for text calculation:

DOLLAR(number) - Dollar representation of a number.
FIXED(number, decimals) - Fixed format form of a number.
LEN(text) - Length of text, in digits.
MID(text, number-of-characters) - Middle characters from a text value.
REPT(text, number-of-times) - Repeats text a specified number of times.
VALUE(text) - Changes text value to number.

Other functions:

COLUMN() - Column number.
ROW() - Row number.
INDEX(area, subscripts) - Gets a value from an area.
LOOKUP(number, table) - Looks up a value in a table.
DELTA() - Maximum change between iterations.
ITERCNT() - Interaction count.
DATE() - The current date.
TIME() - The current time.
NA() - #N/A (error value).
PI() - The constant 3.1415926535898.

Joy or Dismay?

If you're comfortable with complex business or scientific calculations, your heart may now be all a-flutter. If you're a babe in the numerical woods, you may be thoroughly dismayed and convinced that you will never, never, buy or use one of these electronic contraptions.
For those in the first group, the above is just a hint of Multiplan's possibilities. You need not have to be convoluted indeed not to be well served by Multiplan.

For those in the more jittery group, a few additional thoughts on Multiplan:

- For simple applications, you may never use any of the more obtuse functions, nor will you miss their absence. Doing your home budget, balancing your checkbook, or planning for your next vacation will never require TAN or NPV or LOGIO. (TRUE and FALSE might come in handy, though).

- The Microsoft manuals are some of the best in the business; clear, well-written, with numerous examples. A full-blown, easy to follow tutorial is also included, along with a complete reference section and thorough index. You may be using those fancy functions sooner than you think!

- Soon to come are templates; pre-written spreadsheets that do most of the hard stuff for you. Templates are generic worksheets: the common entries and formulas are already entered. The formats are professional; sure to please bankers and bosses; the formulas are prechecked for accuracy. All that's required is a brief customization for your specific needs and the entry of your specific numbers. Microsoft now markets, for other computers, financial and budget templates. Templates for taxes, real estate, and other high and low financial purposes are surely on the way.

- As always, Macintosh makes it easy. Browse the menus, click and drag with nerve. If you're veering toward trouble, a Dialog Box will offer advice, or an Alert Box will offer a way out.

Electronic spreadsheets are one of the most popular uses for small computers. Microsoft's Multiplan will only enhance that popularity.

**Chart and Multiplan: Alone or Together?**

Microsoft's Chart will most often be considered a companion program to Microsoft's Multiplan.

No harm in that. The two programs work together to provide advanced spreadsheet capabilities, and advanced charting and graph-
Screen 14.1 Microsoft Chart Display

ing capabilities. Data from Multiplan is easily passed to Chart for graphic display in a variety of ways.

Chart, however, isn't choosy about its companions. Information from BASIC, other calcos (when they arrive), and from Macintosh-created files can also be used with Chart.

But Chart can also stand alone. Chart data can be easily and quickly entered into Chart for instant graphing. So easily, in fact, that many users will use Chart most frequently as a stand-alone program. Slam in a title and a column of numbers from the Mac keyboard or numeric keypad, pick a chart or graph from an abundant menu of choices, and let Chart do the rest: the chart is scaled, the axes are labeled, the appropriate numbers are shown along the edges.

Is it that easy? Yes, it is. The author's first version of Chart showed up one day on a Macintosh disk. No instructions, not a clue about how to work the program. Within minutes, Mac was displaying a variety of elegant charts and graphs: bar graphs, line graphs, pie graphs, and more. This was not only the author's first brush with Chart, but his first brush with any type of charting program. It is that easy. And your version will include a manual, which, if Microsoft's reputation is a guide, will be thorough, complete, and written entirely in English.
15. What To Use a Macintosh For
Write a book
Book a flight
Keep your books
Write your bookie
Draw a Wookie
Chart a function
Find a typo
Learn to type
Type a blood sample
Sample a population
Pop an address
Address an envelope
Enter a program
Program a game
Chart a gameplan
Plan a campaign
Write your congressman
Run for congress
Congratulate a friend
Find a good investment
Invent a faster-than-light-drive
Drive a car
Build a boat
Balance an account
Run a planetarium
Plan a flower garden
Guard your home
Train pigeons
Talk to dolphins
Design a bicycle
Plan a budget
Build a castle
Play chess
Print checks
Make change
Change your attitude
Attempt the impossible
Succeed
Impress your neighbors
Notify your landlord
Learn a language
Create a language
Draw on the right side
Calculate sidereal time
Time an event
Investigate math
Prove Fermat's last theorem
Catch a thief
Design a house
House some facts
Improve your spelling
Write a song
Write a resume
Make children smile
Study the world
Dial Boston
Analyze the Boston Pops
Program a PERT chart
Perfect your reflexes
Reflect on nature
Design an eco-system
Do your homework
Avoid housework
Check your nutrition
Narrow your priorities
Receive electronic mail
Manipulate some figures
Figure your taxes
Eliminate tautology
Take a break
Check your appointments
Apportion your income
Surprise yourself
Swap tales with a guy in L.A.
Sort numbers
Order silver
Check the commodities market
Market your wares
Design a wreath
Get rich at your own pace
Paginate a document
Practice your Spanish
Squander your paycheck
Design a tapestry
Tap into a discussion
Dissect an algorithm
Teach your children
Calculate the wind-chill
Store your chili recipe
Write a receipt
Reconcile a bank statement
Make a statement
State your goals
Study geography
Control a robot
List your talents
Take on a challenge
Plot a recursive function
Have some fun
Plan your vacation
Verify your suspicions
Cut your energy use
Find a better way
Keep a phone list
Draw a Christmas card
Check your mileage
Record a milestone
Navigate by the stars
Learn to fly
Run your darkroom
Read an encyclopedia
Publish a newsletter
QSL the world
Tell a story
Design a stained-glass door
Check your accounts receivable
Plan your day
Help the disabled
Dissimilate information
Write your mother
Train your pets
Control a greenhouse
Stand up for your rights
Write a play
Stop a war
Think logically
Write a warranty
Bug a corporation
Or start one
Argue persuasively
Pursue the truth
Design an airfoil
Manage swine
Learn LISP
Print a thank-you
Think-through a dilemma
Up the organization
Solve a mystery
Compare notes
Study organic chemistry
Catalog your stamp collection
Publish your novel
Start something small
Shop for bargains
Draw a map
Map your beliefs
Decide which tree to plant
Print 1,000 mailing labels
Work at home
Enter a strange world
Study for the SAT test
Make something easy
Doodle
Write your will
Play the ponies
Conduct a survey
Amortize a loan
Make a conceptual leap
List your assets
Attest to a friend's character
Hone your skills
Revise a document
Refine your position
Post a notice
Notify your creditors

Credit an account
Compose a speech
Conquer the galaxy
Write a love-letter
Get it done first
Graph it
Break it down further
Transpose columns
Figure sub-totals
Find a thread of meaning
Multiply it all by 12%
Save it for next week
Make inventory painless
Outline your presentation
Plan for the presidency
Try a new approach
Predict the future
Understand the past
Match wits with a machine
Create an index
Write your autobiography
Turn on the coffee
Water your lawn
Balance your checkbook
Study your horoscope
Improve your grammar
Name the baby
Valuate your coin collection
Learn geometry
Discover Queortfort
Design a solar collector
Do the payroll
Pay your bills
Estimate how much paint you'll need
Design a needlepoint picture
Pitch a perfect inning
Boost your income
Pilot a space shuttle
Program in Pascal
Send a memo
Improve your memory
Alphabetize a list
Run your furnace
Furnish documentation
Document your accomplishments
Keep a dairy
Manage a dairy farm
Encourage serendipity
Fight the good fight
Cross-reference it
Study the bible
Sharpen your bidding
Play a scale
Write a haiku
Now you think of a few hundred more uses for Macintosh!
Section I: Operation
Section II: Software
Section III: Programming
Section IV: Hardware
Section V: Epilogue
### Bug Report

**Macintosh Department of Entomology**

<table>
<thead>
<tr>
<th>Infested software</th>
<th>Infested documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Describe the bug. Can you provoke its appearance? If possible, provide a code fragment that reveals the problem.</td>
<td>In brief, cite the problem. Please provide any suggested revision.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Department</th>
<th>Phone + M-S</th>
</tr>
</thead>
</table>

Please return bug reports to Sandy Tompkins, x3211, M-S 3-6. Use back of page.

16. Learning To Program
Sooner or later, you’ll probably get the urge to write programs. Your motivation may be curiosity, or you may have a need that is outside the scope of Macintosh-supplied tools or purchased programs. If you’ve never written a computer program, a few comments, cautions, and some words of advice are in order.

**Problem Solving**

First, realize that computer programming is problem solving. If you enjoy solving problems, particularly tricky, complex problems, you’ll probably be entranced by computer programming.

Writing computer programs, like problem solving in general (and *writing* in general), can be laughably simple or unbelievably difficult. Whether simple or difficult depends on:

- the problem you’re trying to solve
- the language you’re using
- how well you know the language
- your ability to think logically, and
- your general computer knowledge

In reality, computers only understand one language: binary machine language; ones and zeros. Machine language programs are a sequence of instructions, in binary form, that are processed one by one. Each instruction forces the computer to perform one simple and specific action: add two numbers, compare two numbers, move a number into (or out of) memory, and so on.

Machine language programs also require data for the instructions to act upon. The data may be part of the program, or reside on disk, or be entered from the keyboard. The computer can add marvelously fast, but it needs to be told which numbers to add, and where in memory the numbers can be found.

The instructions and data depend, of course, on what you’re trying to accomplish, whether it’s computing the national debt, or blasting mutant aliens from outer space. The program logic devised to solve the problem is called an algorithm.

Algorithms are solutions that solve the problem.
Algorithms

Computer programs consist of algorithms plus the instructions and data to implement the algorithms. A well-known text by Nicholas Wirth, Pascal's creator, is titled *Algorithms + Data Structures = Programs*. The book is excellent, but the title alone is required reading.

When you become an ace programmer, you'll begin to see algorithms everywhere, not just in computer programs. You already have algorithms for getting dressed, brushing your teeth, doing your taxes, and making your bed. You just never realized that those processes were, in fact, algorithms.

Interpreters and Compilers

In the next chapter we'll glance at a number of popular, or at least well-known, computer languages. Regardless of their appearance or purpose, all of the languages can be divided into compiled and interpreted languages.

But didn't we say earlier that all programs are really machine language instructions? Yep. No matter what language you choose to program in, it all comes down to the same set of binary instructions, logical, relational, and mathematical rules. Deep within the computer, the line that appeared as [almost] English, is translated into something that looks like this:

```
0011001101001101
1001001111001011
1001001011011010
1000001111011011
```

It's best not to think about reality at this level.

All computer software is built from a few instructions designed to be understood by the computer's microprocessor. The microprocessor is the computer's central brain and, as the word says, processes the individual instructions. Mac's microprocessor, the 68000, employs only 56 fundamental instructions.

The machine language instructions can be combined in endless ways. Want to invent a new computer language? Adhere rigorously to formal logic, create a nomenclature, arrive at a philosophy of how the language ought to work, and a new language is born. (No, it really isn't that easy, but that's the general idea.)

The difference between compiled and interpreted languages results
from the method of translating the language into machine language instructions.

A compiled approach is easier to explain.

With compiled languages, the program is first written using a word processor, or some other means to enter program lines into computer memory. The program text can't be executed, or run, at this point. It must first be processed by another program, a compiler, which scans the lines, determines what instructions the words represent, and translates (or compiles) the text into machine language instructions that the computer can execute. The program usually needs to be processed twice to thoroughly complete the translation.

Speed of program execution is the primary advantage of compiled languages.

Compiler disadvantages are three-fold. First, compiling a program is often a lengthy process. One tiny error can, and usually does, result in returning to the original program text, correcting the error, then attempting to re-compile the program.

The time-consuming nature of compiling causes a second disadvantage: compiled languages aren't interactive. The sense of immediacy—of writing, then running a program, or portion of a program—is lost. As a result, it's difficult, if not impossible, to write a program on the fly when using a compiled language.

Finally, compilers are often expensive and require additional memory. With computers, memory and money are inversely proportional; the more you have of one, the less you'll have of the other.

Interpreted languages translate program lines into machine language instructions while the program is running, one line at a time. With interpreted languages, though, much of the program is pre-written rather than translated. Your program commands are hooked into pre-written machine language routines. When using an interpreted language, therefore, it's necessary to first load the language, then run your program within the language, so to speak. Not so with many compiled languages, where your program is converted into binary instructions that the computer can run, or execute, with no further help. The actual language, whether it be C, Pascal, or Fortran, doesn't need to be in the computer's memory when the program is run.

Because interpreted languages do their translating during program execution, the time-consuming compilation process is avoided. As a result, the process is more interactive, but program execution speed is somewhat slower.

Macintosh evades this potential slowness by using the MC68000 microprocessor, which runs at a remarkably fast clip, and makes everything about Macintosh fast.
Whichever you choose, interpreted or compiled, similarities abound. Once you understand the fundamental notions of one language, learning your next language will be vastly easier.

Syntax, Problem Definition, and the Joy of Hacking

If you program in assembly language, which is only a short step higher than machine language, the instruction ADD will cause the program to add something. If you program in BASIC or Pascal, the symbol + in the correct place will cause addition to be performed. If you program in C, the symbol ++ means "increment (or add 1) to the variable associated with this symbol."

If, someday, you acquire the expertise to invent a new computer language (not an easy feat), you may choose to represent addition with the symbols TAKE-THE-VARIABLE-TO-THE-IMMEDIATE-RIGHT - AND - LEFT - OF - THESE - WORDS - AND - ADD - THEM - TOGETHER.

This particular choice isn't recommended, however, too much typing involved.

The rules that make up any given computer language are called the language's syntax. Computer languages have syntax just as human languages have syntax. Certain words and symbols represent certain specific things and must be used absolutely correctly.

If you're new to computers, your first programming failures will probably happen because:

- You used the language's syntax incorrectly; or,
- You incorrectly defined the problem to be solved.

Syntax bends for no one. If you type PRNT rather than PRINT, nothing good comes of it. Even computers as advanced as the Macintosh fail miserably when it comes to guessing what the human really meant to type, but didn't.

The second common failing is problem definition, or trying to do something you haven't completely thought out.

Consider how programs are created. Although there are many methods to creating programs, the best advice is divide and conquer. First, break the problem down into steps, or modules. Then break the steps down into still smaller steps. Eventually, you'll get down to the level of computer language. If you've broken down the problem
thoroughly, the task of programming will become, if not gleefully fun, at least manageable.

The formal term for this process is Stepwise Refinement.

Some languages, such as assembly language, require an immense amount of breaking down. Others, like Basic, handle many of the pesky details for you. (Do you really care what part of Mac's memory contains your program? Most people don't.)

The problem definition problem afflicts newcomers and veterans alike. After all, if we knew exactly what we wanted, we might not need a program in the first place! Often, the easiest course is to wade into the problem and begin writing program lines, or code. We might have no idea what algorithm will solve the problem; instead, we take a few runs at the problem, and hope the solution arrives before our patience leaves.

Many programming courses frown on such heresy. Coding, or writing the actual program lines, should be one of the least time-consuming aspects of programming, they say. The majority of time should be spent on:

- precisely defining the problem to be solved,
- defining the inputs (data) that the program will use,
- determining how the computer will process the inputs, and
- defining the program's output.

Only then, they say, are you ready to actually begin typing strange symbols into the computer.

Fiddlesticks! Not that it's a bad idea. It's a wonderful idea and better programs would surely result. Fortunately or unfortunately, however, many people like to begin programming by simply sitting down and typing in lines of code.

It's your computer, do what you like. Don't feel guilty.

Another term for fearlessly wading into the midst of programming is hacking. Hacking is best defined as writing code for the sheer joy of writing code.

The Macintosh is a wonderful machine for hackers. Macintosh BASIC is a powerful language that does a lot with very little code. Mistakes, a common occurrence, are easy to spot and correct. Like all good machines, Macintosh BASIC behaves in an understandable manner.
Learning the Art

Learning to program is like learning anything. Read as much as you can, think about what you’ve read, and practice, practice, practice. Be prepared to devote not a few hours to learn programming. Some authorities believe the average person requires 300 hours, minimum, to achieve a moderate proficiency in Basic.

300 hours?!

As in art and music, however, some people are natural programmers. For them, the knack of logically solving a problem is mystifyingly easy.

You might have that knack. Or you might not. One thing is certain: all things are impossible to those who never try.

And you may discover that you enjoy programming. Library and bookstore shelves attest to the fact that a great number of people truly enjoy programming. It is, after all, satisfying to solve problems, whether the problem involves corporate finances or cosmic invaders.

Computer programming provides mental exercise and challenge of the highest order. If you enjoy a challenge, programming offers unlimited vistas. Believe it or not, some people write programs but have no desire to run them on a computer. For them, the joy of problem solving, of finding the proper algorithm, is sufficient.

It could happen to you.
17.

Programming Languages

PASCAL ON $5 A DAY

PL/1

APL

C

FORTRAN

PASCAL

LISP

BASIC

Modula

Ada

C

FORTH
Inevitably, Macintosh will be a popular computer. As Macintosh sales rise, the number of software programs available for Macintosh will increase rapidly; among them will be a host of languages—maybe not all, but surely most of those mentioned below.

Where applicable, guesses and hints will be provided concerning the future availability of specific languages.

Some of the languages that will, or may, become available for the Macintosh include:

**Assembler**—The lowest of languages, assembler works with the actual instruction set of the microprocessor. Not recommended for beginners, but essential if, for some reason, you must know how computers really work. Assembly language reveals algorithms in the stark, white light of exactness. For that very reason, it is greatly appealing to some temperaments.

Assembly language is required for applications that demand blinding speed and precise tweaking and jolting of the computer's internal mechanisms. Assembly language routines can be optimized for both speed and compactness to a degree possible in no other language. The internal software that runs Mac is written in assembler and packs a whallop into an amazingly small area of memory. Expect a Macintosh assembler package early in 1984.

**BASIC**—The ever-present, love it or hate it personal computer language. Some say it encourages poor programming techniques, others say if it can't be done in BASIC, it isn't worth doing. Many versions exist, both interpreted and compiled, and recent versions include many advanced features of higher level languages. Macintosh BASIC is one of the most recent, and best, implementations of the language.

Also, because proprietary versions of BASIC are typically supplied by computer manufacturers, the individual BASICS are much more likely to access the special features of the computer. In Mac's case, Macintosh BASIC provides simple means to access advanced screen display functions found only in the Macintosh. Those same features could be handled by other languages, but the time and effort necessary to write the access routines from scratch would be prohibitive, unless you truly enjoy a challenge.

Like it or not, the world of personal computers revolves around BASIC—the de facto standard language for small computers. Because of its popularity, programs of all shapes and sizes are available in BASIC, many free for the asking from users groups or electronic bulletin boards. Although different makes of computers employ different versions of BASIC, the process of conversion is usually more
tedious than difficult. At least a smattering of BASIC is mandatory for most users. For details on Apple's latest and greatest version of BASIC, see Chapter 18.

**LOGO**—The successor to BASIC? Favored by educators to teach computer literacy, LOGO is also a flexible tool for grown-ups. Excels at easy to create graphics and other mathematical manipulations (make no mistake, computer graphics are mathematical in nature). LOGO, at heart, is a close cousin to LISP, another unstructured and recursive language. Recursion is a difficult concept both to explain and understand, but here goes: recursive functions are functions that call themselves recursively.

How dare I define something in terms of itself? That's just how recursion works. It's the math equivalent of a snake nibbling its own tail. The trick is that the functions and definitions may only call simpler versions of themselves. As you might imagine, endless loops are to be avoided.

It is sometimes difficult to think about not thinking about thinking about recursion. At least I think it is sometimes...

Apple made a strong commitment to LOGO, as have other manufacturers. LOGO is interpreted and strongly interactive; a learn-as-you-go language that is just beginning to be widely used. Expect a version of LOGO sometime in 1984.

**Pascal**—Good news for Macintosh owners! A superb, new, interpreted version of Pascal has been written exclusively for Macintosh: Macintosh Pascal, written by Think Technologies and marketed by Apple Computer. Macintosh Pascal is officially sanctioned by Apple, and provides access to all the Macintosh features. More on Pascal in Chapter 19.

From the marvelous, to the mundane, here's some background on Pascal in general...

Pascal, typically, is a compiled language. Pascal was originally created to teach proper programming, and it shows. The language gently forces you to write programs in a correct, block-structured and understandable manner—strictures that are great virtues or irritating straight-jackets, depending on your point of view. Pascal is fairly easy to learn, quite powerful, and popular. Many different versions exist, differing in speed, flexibility, capabilities, and price. A good meat and potatoes language for serious beginners.

Pascal compilers convert program text into either machine language, or an intermediate type of code known as p-code. How the
program is written isn't affected by either method, only the form of code that the compiler produces.

UCSD Pascal, developed at the University of California at San Diego, is the best known of the p-code compilers.

Both methods have advantages and disadvantages. Because the machine language Pascal compiler produces true machine, or object code, the programs that result usually execute extremely fast. This is because a good compiler is precisely tailored to the computer microprocessor's instruction set.

P-code, on the other hand, is neither fish nor fowl. It isn't machine code and it also isn't code that looks anything like the original Pascal program that was typed into the computer. Instead, it's termed an intermediate code. The Pascal program first compiles the program into this intermediate code. The intermediate code must then be interpreted while the program is run, much the way BASIC programs are interpreted during execution.

Although not yet available, expect, eventually, some form of p-code Pascal for the Mac.

Many good books on learning Pascal are available. A few of them are listed in Further Reading at the back of the book.

Ada—Pascal writ large. Compiled. The official language of the Department of Defense, made to order. COBOL's governmental replacement. If you need to conduct a nuclear war with maximum efficiency, this is the language for you. Otherwise, less complex languages will do the job just fine. Ada is also beginning to replace COBOL on college campuses, and some see it also replacing Pascal. Never happen.

C—A compiled language, C is similar in many respects to Pascal. The language is widely used as an alternative to assembly language and usually produces very fast code. A sophisticated, advanced language, again not recommended for beginners. Also touted as a replacement for Pascal; don't bet on it. Expect a C compiler for Macintosh early in 1984.

PL/I—A heavy-duty, high-level language, similar to both FORTRAN and COBOL. Compiled. Like C, first written for impersonal computers. A classy, not-for-beginners language with a wealth of features. Usually fairly expensive.

Modula 2—A new language from the author of Pascal. Compiled. Allegedly cures the many ills of Pascal. Modula 2 receives good press
FORTH / FORTRAN / APL / COBOL / PILOT / LISP / PROLOG

and is just beginning to rise in popularity. This one's a contender, all it needs is a good manager and the right PR agency. Worth investigating.

FORTH—A powerful, yet quirky, language. Interpreted and quite fast. Somewhat like assembler and somewhat like high-level languages. Devotees call it the best language for beginners and everyone else. Others aren't so sure. Versions are typically inexpensive, compared to other languages. If you have an affinity for Reverse Polish Notation, this language may call you FORTH (the language is noted for bad puns). Read up on it before plunking down your money.

FORTRAN—The old war-horse of the scientific community. Compiled, faded, and dated. If you truly need to crunch massive amounts of numbers, look instead to APL.

APL—A powerful scientific language. Interpreted. Difficult to learn, primarily because of a non-standard, yet powerful, set of programming symbols. If you need to perform sophisticated math operations on large matrices (tables) of data, take a close look. APL's power, like that of C, can be used unwisely and result in code that is extremely impenetrable; APL is famous for one-liners that accomplish much, at the expense of understandability. The villain here is the programmer, not the language.

COBOL—The old war-horse of the business community. Always compiled. A massive, cumbersome language. Cobol running on a Macintosh would be an anomaly too terrible to contemplate. Which doesn't, of course, rule out someone offering it for sale. Don't buy.

PILOT—A language developed to create CAI, or Computer Aided Instruction lessons. Interpreted. Interesting, easy to learn, but declining in popularity.

LISP—The darling of the artificial intelligence community. Interpreted. Difficult to learn, yet extremely flexible. First read Godel, Escher, Bach (see Further Reading). If you like the book, you'll love the language. A good understanding of Logic goes a long way toward understanding LISP. One of the few languages that merits the adjective fascinating.

PROLOG—Another artificial intelligence language, PROLOG has tree-structured roots (an inside joke) in LISP. Interpreted. The Japanese intend to base their Worldbeater fifth-generation computer on PROLOG. The language behaves more like a sophisticated database
program than a programming language. If you enjoy studying programming languages, study this one.

Confused?

Don't be confused. Computer languages come in all shapes and sizes. It would be a boring world without LISP, FORTH, Chinese food, or mud-wrestling.

Your choice of language will depend on your bank account, temperament, and the nature of your programming problems. LISP is not the best language for writing General Ledger programs, nor is Pascal the ideal tool to investigate intelligent programs.

Programming, like all exploding and infant fields of endeavor, is ripe with opinions. Read a few books, try out some languages, and you'll be perfectly entitled to become as opinionated as the next person.

As mentioned earlier, bookstores and libraries now carry numerous books about computers, programming languages, and programming in general. Some suggestions for further reading can be found, not surprisingly, Further Reading.

Users Groups

Another good source of information is users groups. If a Macintosh Users Group isn't yet organized in your community, you have an opportunity to be both the group's founder and first president.

What do users groups have to do with learning computer programming? Quite a lot. Users groups are an excellent source for a wealth of free public domain software, written by hobbyists and users of all stripes: from technically pristine programmers, to rough and ready hackers. Using, exploring, and modifying these programs are excellent ways to learn. It's also nice to make friends who can offer advice or help during those inevitable times when you're stuck in the midst of a programming project.

User groups also provide computer owners a chance to swap stories and software, commiserate, receive bulk-buying discounts, test-drive software, and generally mingle with other computer owners who made equally savvy computer selections.

Enjoy.
18. MacBASIC: A Preview
Welcome to a sneak preview of Macintosh BASIC—hereafter called MacBASIC.

Created by Apple's Donn Denman, MacBASIC is scheduled for release sometime during the summer of 1984 (sorry, that's the best information available).

MacBASIC is a new, terrific, outrageously wonderful version of the BASIC language. Does that sound like un-objective hype? It isn't. In many ways, MacBASIC is also a study in what may seem to be contradictions.

It is both a very simple BASIC, and a very extensive BASIC. It will impress people who know BASIC, but it will also be easily learned by people who know nothing about BASIC.

If you also program in Pascal, MacBASIC will seem, in many ways, to have a Pascalish feel to it. Pascal cures many of BASIC's ills, but MacBASIC cures the many of those same ills without resorting to an entirely new language.

By now, you're probably tired of hearing this, but, like everything else about Macintosh, MacBASIC is also extremely fast.

What follows is merely a quick glance at MacBASIC. As this book was being written, MacBASIC, also, was being written. If anything, however, MacBASIC will be even niftier than this short summary indicates.

Some notable features of MacBASIC are:

- Concurrency
- Optional line numbers
- Labeled GOTOs and GOSUBs
- Advanced flow of control structures
- A rich set of data types
- Graphic abilities
- Access to the Macintosh User Interface

Let's take those features one by one.

**Concurrency**

What is concurrency? Imagine this: you write a MacBASIC program (in a window, of course). Then you select Run (or possibly Do it) from a menu. Up pops a second window—and there's your program, happily running away in the window.
**Concurrent Tasks**

While the first program is running, you nonchalantly open a second window, and write a second program. Or, maybe you load a program from the disk drive. Whichever, you run the second program. Up pops another window. And there's your second program, also doing its thing in its own window.

And all the while, the first program is running undisturbed.

That's concurrency. A number of separate programs running simultaneously in different windows.

Not only can a number of programs run concurrently, but the programs can pass messages between themselves, to jointly attack a problem, make short work of lengthy matters, or entrance you in multiple ways, all at the same time.

It gets better. Not only can two or more programs be running at once, but it's possible to make changes to a program **while the program is running**. An amazing feat; no more LIST, RUN, HALT, LIST, EDIT, RUN, STOP, etc. With MacBASIC, it's possible to edit and run a program at the same time.

The ultimate beginner's BASIC. The ultimate programmer's BASIC.

**Optional Line Numbers**

MacBASIC program lines may be numbered or un-numbered, whichever you prefer. In fact, if you do choose to use line numbers, the numbers need not even be sequential!
The three following program segments are all, believe it or not, absolutely correct.

10 PRINT "First line"
20 PRINT "Second line"
30 PRINT "Third line"
   PRINT "First line"
   PRINT "Second line"
   PRINT "Third line"
20 PRINT "First line"
10 PRINT "Second line"
30 PRINT "First line"

Each segment will print the lines in the same order. To MacBASIC, program lines are labels without colons. Program lines will always execute in sequential order, unless referenced by other lines.

**Labeled GOTOs and GOSUBs**

A label is always the first word of a line, and is always followed by a colon. LABEL: is a label. This alone tremendously enhances the readability of programs. Program lines such as:

```plaintext
LABEL: PRINT "This line is not numbered or labeled"
```

when run, produce:

This line is not numbered or labeled
This line is labeled
This line is labeled
This line is labeled

(and so on, forever...)

Labeled GOSUBs are a Godsend for creating and understanding long, complex programs. Instead of cryptic statements such as GOSUB 1255, programs can be written:

```plaintext
GOSUB Titlepage:
GOSUB Getwishes:
GOSUB Mainprogram:
```
Because keywords may be used as labels, with no restrictions, other perfectly legal lines are:

GOSUB Gosub:

or,

GOTO Loop:

**Flow of Control Structures**

MacBASIC implements the familiar FOR-TO-NEXT loop, with an optional STEP parameter. Also available is the handy DO-LOOP, with an optional EXIT:

```
LEAVE_TIME = FALSE
DO
  PRINT "Here we are in the DO-LOOP"
  IF LEAVE_TIME = TRUE THEN EXIT
  LEAVE_TIME = TRUE
LOOP
  PRINT "And now we've gracefully EXITed"
```

This masterpiece, when run, produces:

```
Here we are in the DO-LOOP
Here we are in the DO-LOOP
And now we've gracefully EXITed
```

Another highlight of MacBASIC is the extended IF control, accomplished by the IF-THEN--(ELSE)-ENDIF construction:

```
IF TIRED_OF_EXAMPLES = NOPE THEN
  KEEP_READING = TRUE
  TIRED = FALSE
  GOSUB NEXTPAGE:
  PRINT "Way to go, reader!"
ELSE
  KEEP_READING = FALSE
  TIRED = TRUE
  AUTHOR$ = SADS
  GOSUB QUIT:
ENDIF
```
IFs may be nested within other IFs. Just remember that each IF can have only one ELSE, and each IF must have an ENDIF.

```plaintext
IF (condition) THEN
    IF (condition) THEN
        [statement]
        [statement]
    ELSE
        [statement]
        [statement]
    ENDIF
ENDIF
```

The extended IF-ENDIF will be hailed by BASIC programmers who, till now, had to create often unreadable and convoluted code to test and set conditions.

### Data Types

MacBASIC data types and the symbols associated with each are given in Appendix E. All the well-known BASIC data types are included in MacBASIC. All characters in variable names are significant, and variable names can be any length.

Some of the more notable types are:

- **Double precision real.** The default data type. FIRST_VARIABLE would be treated as a double precision real variable, significant to 15 digits of accuracy.

- **Boolean data types.** Boolean types are identified by a tilde (~) as the last character of the variable name. A boolean variable, such as BOOL_VARIABLE~, can only be either true or false, 1 or 0. Boolean arrays are also allowed.

- **Extended precision real.** Ideal for scientific or mathematical calculations. Variable names that end with a backslash, EX_PRE_VAR\, are accurate to 18 + digits.

Variables of different types may share the same variable name. Although probably not advisable, it's permissible to write a program that contains the variables:

```plaintext
THIS_VARIABLE%
THIS_VARIABLES and,
THIS_VARIABLE\`
```
Graphics

Yes, MacBASIC has a wealth of powerful graphic commands. Good thing; if graphic commands were omitted, BASIC programmers would probably storm the gates of Apple (if Apple had gates!).

But let's backtrack for a moment.

Underlying the Macintosh screen display is a collection of amazing routines that together form the QuickDraw graphics package. Each procedure or function does a particular duty: set a point, move an image, and so on. Procedures that perform graphic operations on ovals, for example, are: FrameOval, PaintOval, InvertOval, and FillOval.

Without getting too specific, let's just say that there are lots and lots of QuickDraw routines.

MacBASIC won't allow you complete access to the full power of QuickDraw; to do that, you'll have to use Pascal (or assembler, but Pascal's easier). Instead, MacBASIC offers a subset of commands; a healthy subset that allows you to perform the most needed, and most requested graphic manipulations.

At present, MacBASIC allows easy manipulation of three types of shapes: ovals, rectangles, and round rectangles. Apple calls these:

```
OVAL
RECT and
ROUNDRRECT
```

In general, there are four ways to manipulate the three shapes:

```
PAINT
ERASE
FRAME and
INVERT
```

So far, we have shapes and manipulators. Now all that's needed are coordinates—where the shapes appear on the screen.

All images are made up of points, or pixels, on the Macintosh screen. The pixels can be thought of as points on an invisible grid, 512 pixels wide, and 384 pixels high. Coordinates begin at the upper-left corner of screen, and reach their maximum at the screen's lower-right corner.

Screen coordinates are always given with the horizontal position first, followed by the vertical position. 0,0 is the upper-leftmost position, and 511,383 is the lower right-most position. (Why not 512,384? Because we started at 0, so we're 1 number shy after we've counted all the way up. You did count, didn't you?)
All shapes have a minimum of four coordinates: two horizontal, and two vertical, divided into two sets:

HorizontalPoint, VerticalPoint; HorizontalPoint, VerticalPoint

(Notice the semi-colon, a necessary symbol between pairs of coordinates.)

Learning to think in terms of coordinates is often difficult. Practice and you'll get it eventually, however. Remember that a set of two coordinates identifies one point on the screen. Here's how to specify a rectangle (RECT) that begins at the upper-left corner of the screen:

0, 0 ; 20,10

Notice that only two points of the rectangle are given: the upper-left corner, and the lower-right corner. MacBASIC draws the lines that create the rectangle using only those two points (if we specify RECT, which we must).

Now we've got the necessary graphic equipment: shapes, manipulators, and horizontal and vertical position coordinates. It may be helpful to think of manipulators as verbs, and shapes as nouns; then graphics become a matter of specifying:

VERB NOUN Horizontal pixel, vertical pixel ; horizontal pixel, vertical pixel

Let's change horizontal pixel and vertical pixel to HP and VP, respectively. Then we have:

VERB NOUN HP,VP ; HP,VP

Verbs, of course, are action words. The available graphic verbs are FRAME, PAINT, ERASE, and INVERT.

FRAME outlines a shape.
PAINT fills in a shape with the color black.
ERASE doesn't actually erase; it paints the shape white. Since the screen is white to begin with, this appears as erasing, but should give you some good ideas.
INVERT also paints; in this case the paint is the reverse of the existing color: black becomes white, and white becomes black.

To outline our example rectangle, we enter:
FRAME RECT 0,0 ; 20,10

That’s all there is to it.

To draw a filled-in rectangle, we don’t need to use FRAME; the verb PAINT is all that’s necessary:

PAINT RECT 0,0, ; 20,10

Rather have a rectangle with rounded edges?

FRAME ROUNDRECT 0,0, ; 20,10

Strange shapes and clever line drawings are the province of PLOT, which uses coordinates in much the same manner, only a number of points may be specified, not merely two sets.

Hey, this was supposed to be an overview of MacBASIC, not a tutorial!

Sorry. It’s easy to get carried away about this language.

---

Inside the User Interface

All the uniquely Macintosh characteristics: menus, objects, buttons, and dials, are components of the Macintosh User Interface, a consistent set of visual rules that govern Mac’s display.

Many of the most useful elements are available to MacBASIC programmers. The two powerful pairs here are SET/ASK, and SHOW/REMOVE.

SET/ASK refers to a number of system parameters that can be set, tested, and changed by MacBASIC programs. The parameters are either SET to give a value, or ASKed, to find out what the current value is.

Most BASICs allow a cursor to be positioned on the screen. MacBASIC allows you, instead, to either PLOT one or more points, or use a Pen, much like the MacPaint pen. The pen has a position, a size, and a mode:

PENPOS
PENSIZE
PENMODE

PENPOS either positions the pen on-screen, using SET PENPOS, or places the pen position into two variables, by use of ASK PENPOS.
SET PENSIZE allows specifying how wide (in pixels) and how high (in pixels) the pen is to be. Want a big pen? No problem:

SET PENSIZE 30,30

is a big pen. Like all Set-options, PENSIZE can also be ASKed for its parameters.

PENMODE uses Boolean logic (OR, XOR) and clear, copy, and inverse. Each mode is represented by a number, from 8 through 15, to choose the desired effect.

SET PENMODE 11

for example, draws in white, for an eraser effect over black areas. The other set-options are:

VPOS
BOUNDS
PORT
VIEWPORT
WINDOW
PATTERN
DIAL
BUTTON
DIALSTATUS
BUTTONSTATUS
MENUSTATUS

Again, all set-options may be SET, or ASKed, which allows programs to set and change parameters on the fly during program execution.

Buttons, dials, and menus are special cases. All may be SET or ASKed, as listed above, but for these favored few objects, programmers may also SHOW or REMOVE, when needed, which simplifies pulling down menus and making dials disappear.

In Conclusion

Those are only some of the highlights of MacBASIC. For more information about MacBASIC, Appendix E lists MacBASIC keywords, data types, operators, and the order of precedence for expression evaluation in MacBASIC.

Learn the language, enjoy, and do great things. But force yourself
to get outdoors once in a while, if only for the vitamin D in sunshine. Or take vitamins, load MacBASIC...
19.

Macintosh Pascal
Well...here's another landmark program: Macintosh Pascal.

Insert the proper disk, click the proper icon, and you're presented with something similar to what's shown in Screen 19.0.

This Pascal limits on-screen windows to two: a program window and an output window. Like most Macintosh windows, they may be re-sized, overlapped, or otherwise shifted around at will. If you prefer to write your own program, and then re-size the output window to fill the entire screen, go ahead.

Macintosh Pascal is one of few interpreted Pascal on the market; most versions of the language are compiled, and make the user slog through the drudgery of edit-compile-test-reedit-compile (again), and run. In this slick, interpreted version of the language, all that's necessary is dragging down to the word Go and releasing the button.

The menu Check option is especially noteworthy. Drag down to Check, then release: the program instantly races through your program, indenting, formatting, block-structuring, and otherwise making your efforts look as Pascalish as possible.

Syntax errors are also checked during the process. If such errors as a missing semicolon are found, the program gives a clear error
message, immediately following the offending characters, and offers hints, when possible, as to what was expected—both a great convenience and often a lifesaver.

Another aid to Pascal education and program debugging is the Step option, which single-steps through program execution. A hand appears to mark which line is currently executing. Watch the hand and watch the program output screen for a classic illustration of how Pascal works—and a marvelous debugging tool when things don't work as expected.

If you've Stepped enough, simply drag down to Go and let the remainder of the program execute at lightning speed.

Pascal is often touted as the best first language to learn; it's powerful, high-level, and opens the door to a number of other sophisticated languages available for small computers.

Until now, though, learning Pascal was a long, hard process. Mac Pascal makes the learning process painless; it's possible to start with a small, already functioning program, and slowly build it into something that you can justly say is yours.

As your program listing is displayed, Pascal's reserved words are displayed in boldface type—a textbook example of Macintosh Pascal's educational value.

The feeling is more one of editing a program, than of writing a program. Editing is easier.

What's more, programmers will soon find that the tough programming tasks have already been completed by Apple programmers: almost 500 pre-written functions and procedures that can be included in any Pascal program. To achieve the desired effect, merely throw in the correct procedure or function and specify the proper parameters.

As an example, you might wish to move or size Macintosh windows. If so, you could use the procedures:

```
MoveWindow,
DragWindow,
SizeWindow,
or the function
GrowWindow;
```

after giving each the proper parameters, not at all difficult to do. [Note: not all of the Macintosh functions and procedures will be available in the first release of Pascal, but most users will never even miss the ones omitted. A second, more complete version will be available later this year for software developers.)

There are very few things that Macintosh can do that aren't easily available from Pascal. That includes all window manipulations,
Screen 19.1  Pascal Error Message

Alert Boxes, Dialog Boxes, QuickDraw routines, operating system calls, text manipulation (Mac has a text editor built into ROM, remember?), font selection and definition, menu management, keyboard mapping, string manipulation, operations on bits, even routines for printing on color printers. The works: all pre-written and ready for use.

This represents a giant step forward in low-level programming of microcomputers. No longer, as in Apple II days, does anyone need to know pokes, peeks, or hexadecimal addresses buried deep within the computer. All the magic of Macintosh is available to anyone who knows the right names and the right parameters.

If you wish, you may program Macintosh in assembly language (with the aid of an extremely good assembler, soon to be offered), but it isn't really necessary; Pascal will suffice for all but the most demanding programming feats.

Which should help fuel the coming explosion in Macintosh software.

Macintosh Pascal was created by Think Technologies, and is marketed by Apple Computer.
Section I: Operation
Section II: Software
Section III: Programming
Section IV: Hardware
Section V: Epilogue
20. Taking Care of Mac
Routine Care

Computers require little routine maintenance. You may want to clean your disk drive once in a great while, but that’s about it, other than dusting or wiping off fingerprints.

Essentially, nothing’s needed to keep Mac running fine.

The Macintosh was engineered to take a lot of abuse. So was the Apple II. Many Apple II keyboards have been hammered on for years. They still work fine. So do the other Apple parts.

Macintosh keyboards are, if anything, even more rugged. Just don’t spill orange pop (or any other beverage) inside.

Not satisfied? Hardly seems enough, does it? But it’s true. Computers do not wear out. After all, there aren’t many moving parts. The disk drive is the only major exception.

It’s unlikely that billions of electrons flying through Mac can wear down the circuitry one tiny bit. Actually, some hobbyists have the firm belief (superstition?) that using a computer is the best way to keep it in top condition.

They may be right.

What We Have To Fear

One fear, common to almost all first-time computer users, is the fear of Breaking The Computer. This fear is also known as: “I spent X thousand dollars on this and I hit the wrong key and IT BROKE!!!”

Don’t be afraid. It won’t happen. You might break Mac by dropping it down the stairs, or using it in the shower (not recommended), but you will not break Mac by pressing the wrong keys, or clicking at the wrong time or place.

Macintosh is designed to forgive. If you press the wrong keys, or click the wrong spot, probably nothing will happen. Certainly nothing bad will happen.

You won’t accidentally erase your disk, you won’t accidentally erase one of Mac’s internal programs, and you will not create either clouds of smoke, or rivers of melting plastic.

Attitudes toward computers range all the way from let me at it, to the fear that touching the keys (or that mouse thing) will cause something untoward to happen. At its most extreme, the fear of computers becomes computerphobia: a morbid fear of computers.

Computerphobia often strikes people brought, against their will, into computer relationships; in particular, office-workers who sud-
Suddenly, a powerful machine arrives; a powerful machine that can magnify tiny slips into ENORMOUS SCREW-UPS.

The best advice is to relax. Try to remember that Macintosh has undergone brutal and devious testing; tests performed with people like yourself, and designed to make the computer as easy to use as possible. If there were ways to cause total destruction by pressing a few keys, they would have been discovered, and eliminated, long ago. Don't worry. Relax. Enjoy. If you do happen to press a potentially disastrous combination of keys [or mouse clicks] a Dialog Box will appear to quizzically inquire: "Do you really want to black-out all street lights on the west coast?"

Odds are, the button reading "Naaah, just kidding" will be bold-faced as a suggested response.

Remember, even computer scientists, hot-shot programmers, and Professional Writers slam the wrong keys fairly often. If those errors caused disaster, we would neither create, program, nor write about computers. We'd get rid of them. Or worse.

And so would you. But disaster, at least in this form, won't strike, so relax. Hit a few wrong keys. Feel like a professional.

Or go one step further: actively court disaster; see if you can make Macintosh crash and burn [as it is euphemistically said]. It should be impossible; the Macintosh software was designed to expect and recover from all potentially silly, thoughtless, and reckless inputs from users. Deliberately pushing the system is a good way to see Dialog and Alert Boxes that you would, otherwise, never run across. Good practice both as a confidence-builder, and as a hands-on exercise in understanding how Macintosh behaves.

While we're on the subject, here's one additional thought. As time goes by, you'll probably find yourself buying additional software tools. When you do, let's hope that every package is as well-designed as the software tools that came with Macintosh. But they may not be. If so, remember: if any piece of software is difficult to use, or hard to understand, it is not your fault: it is the fault of the software. A few lousy programs are certain to surface for use with Macintosh. Try to avoid junk at the appropriate time: before purchase. Chapter 25 offers a few tips on buying software. Even in computerdom, it's sometimes necessary to be a wise shopper.

**Screen Care**

Mac's most sensitive spot is the picture tube, which is similar to a television picture tube. Don't abuse Mac's body any more than you
would an ordinary television, and you should have no problems.

You may want to clean the screen periodically. Display screens, like all glass, seem to attract dust. Computers and televisions, when on, also produce a slight magnetic field near the screen. That attracts more dust.

Most manufacturers caution against using cleaning fluids, such as Windex, to clean the display screen. Most people go right ahead and use glass cleaners anyway.

If you do, don't use too much, and don't spray the cleaner directly on the screen (it shouldn't drip down into Mac or accidently be squirted into the drive opening). Instead, spray the cleaner onto the proverbial clean, dry lint-free cloth, then wipe the screen.

If you're a purist, special CRT cleaners are available, but the results will be about the same.

Dust and Fingerprints

Dusty mouse? Dirty keyboard? Well, time to buy a new computer...this one's obviously had it.

Not necessary. The Macintosh, being fully encased in the modern, space-age material called plastic, may be cleaned with anything used to clean other objects encased in plastic. Again, use your favorite cleaning fluid, dampen a soft cloth (dampen, not soak), and carefully wipe away the offending dust or blemish. We're not talking turpentine here, only the normal spectrum of 409 or Fantastic cleaning solutions.

Mouse-ball cleaning instructions are found in the Macintosh users manual. It's time to clean the mouse when the pointer seems to jump a bit, instead of smoothly following your hand motions. Be on the lookout; heavy mouse use may predicate cleaning your ball about once every two weeks; rolling the mouse on a dusty or cigarette-ash covered surface may lead to cleaning every other day. Just be ready to clean when the pointer ceases to move smoothly. And remember that the ball should be dry before reassembling the mouse.

Mac's keys may also be cleaned; use the same damp cloth and be careful not to let anything drip inside the keyboard. A dusty keyboard may also be vacuumed (use a narrow nozzle), or blown clean with a can of compressed air. Check your art supply store for cans of air.

If you're a truly fearless sort, the keyboard keys may be removed by gently prying them up with a small screwdriver. The key word here is gently. Make sure the keyboard is unplugged from Macintosh (just to be on the safe side) before prying. Once off, the keys can be lovingly cleaned; a task that should appeal to fastidious users.
Reinserting the keytops also requires a gentle touch; you'll be able to feel the tops snap back into place. Try to put them back where you found them. Appendix B will help with that task.

Dust and Disks

Dust, along with smoke and static electricity, is the plague of computers. Dust is most devilish in disk drives. Small dust particles, when hit by drive heads, cause effects similar to your sportscar hitting a boulder at 360 miles an hour.

(The above paragraph is typical of those found in many computer books, although certainly more picturesque. In reality, many users compute for years, chain-smoking and generally raising dust all the while, with nary a head crash to their credit. Which doesn't mean heads don't sometimes crash. They do. People are also killed by falling bridges, struck by lightning, and smacked by oncoming mete- orites. It just doesn't seem worth losing too much sleep over.)

Although not yet available, disk drive head cleaners will be appearing in the finer shops any day now. The package consists of a head cleaning disk that is inserted into the drive. The disk is covered with a cleaning solution (alcohol) that wipes the drive head as the disk
spins. It's a lot like taking alcohol and Q-tips to the head of a conventional tape recorder.

If this sounds like a swell idea, buy a disk cleaning kit, read the instructions, then use the cleaning disk *half as frequently as recommended*. You'll still, probably, be cleaning your head *twice as often* as is necessary.

### More About Dust

By design, the Macintosh is somewhat less prone to dust problems than other computers. The reason is Mac's fan: it doesn't have one. Due to the design, it's not necessary to prevent overheating by wafting air over Mac's innards. As a result, dusty air (and *all* air is dusty) is not constantly gathering inside Macintosh.

Still, dust is dust; one remedy to exterior dust is purchasing dust covers for the computer and keyboard. And mouse, possibly. At this writing, dust covers weren't yet available, at least not from Apple. By the time you read this, they probably will be.

---

### Ventilation

Good ventilation is necessary and often overlooked. Because Macintosh doesn't contain an internal fan, ventilation is especially important. In particular, never place manuals, magazines, books, papers, or anything else directly on top of the Macintosh system unit. Hot air rises, and the vents located on top of the system unit must be kept uncovered. This especially applies to the top left vent (as viewed from the front); a circuit board is mounted vertically below this vent. After a few hours of work, the left vent area is hot; don't cover the vent, the board may object strenuously. Also keep a few inches of clearance free on both sides of the computer (and don't scrunch the back panel into walls, books, or anything else).

In theory, Macintosh may be switched on and left on, unattended, for weeks or months at a time, with no harm done. If the computer is unattended for long periods, however, be sure to turn down the screen brightness; after many, many hours, a constant, bright image might conceivably etch itself permanently into the screen, resulting in ghostlike images forever more. Not likely, but possible.

The above may seem trivial. It is relatively trivial. But there isn't much else to talk about concerning routine maintenance.

Precautions are a different matter.
Sensible Precautions

There are only a few of these. Very simply, they are:

- Don't compute during an electrical storm.
- Don't compute in a foreign country without first determining that the electricity supplied is U.S.
- Plug the Macintosh power cord either directly into a socket or use a heavy duty grounded extension cord.
- Avoid extreme temperatures, both hot and cold. A good rule of thumb is this: if you're frozen or sweltering, so is Macintosh.
- Never plug anything in Mac's rear connectors that isn't designed for use with the computer.
- Never put anything into the disk drive except a disk.
- Don't place disks on or near anything that uses electricity; televisions and other appliances create invisible magnetic fields that can do damage to information stored on disks. This includes placing disks on printers, modems, Macintoshes, or any other device that is on.
- Don't open up the system unit to see what's inside. If you want to see inside Mac, look elsewhere in this book—that's a safer (and more informative) way to examine Mac's innards.

The final precaution is vital. Like ordinary televisions, computers often have dangerous voltages present even when the machine is not plugged in. Now go back and re-read the previous sentence.

If you still must see for yourself, the best advice is to visit your local Apple dealer. The resident technician will [probably] be glad to rip off Mac's cover and allow you a personal inspection.

These are all fairly common sense precautions. Again, treat Mac about the way you'd treat a [highly prized] portable television and you'll probably have no problems.

Insuring Mac at Home

Mac will probably never break, but it may be stolen, consumed in fire, whisked away by a tornado, or washed away by a flood. If so, let's hope that Macintosh is covered by your insurance.
This is an area you should check. If your Macintosh is used at home, it is considered a home computer and likely covered by the property coverage of your homeowner's policy. Although not specifically mentioned, the computer would be covered along with your television set, furniture, and other belongings. Your insurance agent can tell you if additional coverage is needed.

If the computer is stolen or damaged, you'll hopefully be reimbursed. The reimbursement payment will either be based on a depreciated value, or like replacement of goods. Find out which.

Another insurance possibility is a floater: an extension to your insurance policy that specifically covers Macintosh. Floaters are used to cover jewelry and other expensive items. Some policies have floaters tailored for computers; the general form will usually suffice, however.

Expect to pay a slight additional premium for specific computer coverage. The rate may range from $.60 to $.80 per $100 of value.

Insuring software is more difficult. Most insurance companies won't reimburse software destroyed through carelessness. Don't expect a check if you spill coffee in your system unit (or any sympathy from your dealer)!

The simplest way to insure software is not to insure software. Instead, religiously make backups of all your valuable software, then store the backups in a safe place outside your home; a safety deposit box, say.

---

**Insuring Mac at Work**

If you use Macintosh at work, the insurance picture is somewhat clearer and somewhat more expensive.

Most insurance companies now have policies for computers and computer software. The policies can be extremely detailed and cover everything from theft to loss of profits. Unfortunately, many of these policies are geared toward large, expensive mini and mainframe computers. Certainly, many agents will be glad to cover Macintosh under one of these policies. But the premium you pay may give you pause.

Some of the companies that offer coverage for small business computers are Fireman's Fund, Hartford, Unigard, and Safeco.

Because coverage of small business computers is still in its infancy, be prepared for a wide variance in premiums. A few insurance companies are now beginning to realize that computers used in business don't always cost $50,000 and up, and are setting the premiums accordingly. Shop around and don't buy more coverage than you need.
A Final Thought

Although Mac is a rugged, low-maintenance machine, it is nonetheless a major investment. Treat it like one.
21.
A Look at Mac’s Back
The hardware hobbyists are going to be disappointed. The breadboard fanatics are going to be disappointed. The companies that make add-on hardware cards are going to be disappointed. But most Macintosh buyers will be only relieved: there is nothing complicated about the Macintosh back panel. There are no optional cards to plug in anywhere, and only a short list of desirable accessories: a printer, a modem, and a second disk drive or hard disk drive. None of these accessories are at all difficult to properly attach to Macintosh.

And nothing—nothing—ever needs, or even can be added within Macintosh. The box is sealed and will hopefully remain sealed.

This simplicity was carefully planned by Apple. By making a slot-less machine with only one standard configuration, Apple has fashioned a dream come true for software developers: every software company is assured that its products will run on every Macintosh sold by Apple. Only one version of a software package need be marketed; not a color version, and a monochrome version, and an expanded memory version, and a low memory version. One size fits all.

The Macintosh software buyer will have the satisfaction of knowing that every software package will run on his or her machine; a comforting thought amid the complexities of software buying decisions.

---

The Back Panel

Let’s begin at the back. On the top right is a small removable panel that gives access to a 4.5 volt battery. Press down the small tongue at the panel's top and gently pull the panel toward you to reveal the battery. The battery keeps the Macintosh clock updated, even when the computer is switched off or unplugged. Periodic replacements can be made with Eveready 523s or equivalents.

Directly below the battery panel is the Macintosh on/off switch and, below that, the plug for the AC power cord.

From left to right, along the bottom, we find:

- a security kit attachment inset,
- the mouse socket,
- a socket for an external disk drive,
- a socket marked with a telephone icon,
- a socket marked with a printer icon, and
an audio out jack, marked by a musical note.

The optional Apple security chain attaches both into the back inset and into another connector found on the Macintosh keyboard [next to where the keyboard cord is plugged in]. The security kit was unavailable for review when this chapter was written; we can assume it somehow foils those that would otherwise walk away with your Macintosh.

Next comes the mouse socket. Not much of interest here: plug in the mouse and roll away.

The next connector is for the optional second disk drive. The drive is a functional duplicate of the Macintosh internal disk drive. For more driving information, see Chapter 10.

Things become more interesting with the next two sockets: the printer icon socket, and the telephone icon socket.

Both are gateways to what are called RS-422 serial interfaces. To understand the significance of those numbers, we must digress...

---

**Serial Considerations**

Typically, computers communicate with the outside world by one of two methods: serial or parallel interfaces. Serial interfaces are both simpler to construct and simpler to explain: in serial transmissions, the individual bits of information are blasted down the line, one after another, like peas blown through a straw.

Parallel transmission communicates one chunk at a time; each bit of the chunk gets a wire all to itself. As a result, parallel interfaces are usually faster, but more complicated to construct and interface with other parallel devices.

Of course, there are serial interfaces, and then there are serial interfaces. The same for parallel interfaces. Each type has many variations, as anyone who has ever shopped for a cable to hook up the printer can tell you.

Fortunately, a serial standard emerged: RS-232. In general, most things that employed RS-232 serial ports could talk with most other things that employed RS-232 serial ports.

Time passes...

And the time came for a new serial interface standard. Why? Because the RS-232 could no longer cut it in today's world of fast computers, fast printers, and—most of all—networking.

Networking is the achievable dream; cable together a slew of computers and let them all communicate with each other; send inter-
office mail, memos, financial projections, even voice communications. Corporate executives become giddy at the thought of networking.

But networks require serial, not parallel lines. So the standards makers, the Electronic Industries Association (EIA), came up with a number of new, high-speed conventions for data transmission, among them the RS-422.

The Macintosh is one of the first recipients of this advanced communications standard. Macintosh has not one, but two RS-422 serial ports: the modem port, and the printer port. In reality the two ports are completely identical.

Although networking for Macintosh hasn't yet arrived, it will soon. When it does, connecting Macs will be primarily a matter of some additional software, and plugging a cable into the back panel.

And checking the In box on your screen, periodically.

The main advantage of the ports is speed: both can throw out information faster than the fastest printer or modem can match [but don't worry, Macintosh will wait, when necessary]. In theory, the ports are fast enough to handle even voice communications.

Although one of the ports will be occupied by a cable to the ImageWriter printer, any serial printer should easily work with Macintosh, provided the proper cable is obtained and the printer dip switches are properly set (usually a job for your computer dealer).

The serial ports are even speedy enough to service a high-capacity hard disk drive. Another possibility for Macintosh is an expansion box, which would provide a number of expansion slots for hardware add-ons.

But didn't we say, at the beginning, that this was the kind of thing that Macintosh was designed to avoid? Sure. But the possibility is there, and some eager capitalist is almost certain to exploit it, so why hide the fact? Someday there will be expansion cards for Macintosh.

---

**Sound**

Last, we come to the Macintosh audio out jack.

Granted, the Macintosh already has an internal speaker. Evidently, though, the prospect of four-voice sound through the living room stereo was too much for Apple engineers to resist: hence the jack.

The jack can be used in one of two ways:

- To connect an external speaker, or
To use Macintosh as a preamplifier.

The first use is the easiest. Get a 4 to 10 ohm external speaker and plug it in. Speakers of various sizes should work, but remember that you've only got 3 watts of output to work with; no massive woofers, please!

Massive woofers are certainly possible using the second approach, which involves using Mac as a pre-amp. In essence, the Mac audio output is boosted by another amplifier, usually a quality stereo with lots of input jacks on the back.

Find a knowledgeable friend to select the right cord and jack on the stereo, and prepare yourself for some potentially ear-splitting decibel levels.

Enjoy.
22. Inside Macintosh
A Peek Inside

We’ve noted the superb Macintosh design, and stated that if built less cleverly, Mac would be vastly different in appearance. We’ve also cautioned against removing the Macintosh case to see for yourself [and again: don’t do it. You could receive a lethal shock, even if the computer is unplugged].

But we’ve said nothing about what’s within the Macintosh case.

So what’s in there, anyway?

Two gerbils, a round wire cage... Kidding.

But you might be surprised. To the untrained eye, the interior of Macintosh looks quite spartan—there’s just not much there.

Here’s what’s inside:

• A digital motherboard,
• An analog board,
• The Macintosh video tube, and
• The Apple/Sony disk drive.

That’s it. Hardly seems like enough, does it?

Mr. Smith Goes to Town

This is all Burrell Smith’s doing. Like the Apple II and the Apple III, the Macintosh is a one person design. The credit for the Macintosh hardware design belongs to Apple’s Burrell Smith, who was ably assisted by Brian Howard.

Burrell is both a whiz at Defender and a magical hardware designer. As he explains it, designing computers is easy—if you’re satisfied with an ordinary hardware design.

What’s ordinary? Ordinary is simply attacking a design problem by using more integrated components, or chips. Each chip holds the equivalent of hundreds or thousands of transistors. To solve a problem in hardware, or to add features, the easy way is to add more chips. A bunch of chips here, another bunch there...

Good hardware designers turn up their noses at engineers who simply add more chips. It’s the easy way out. It’s also bad design, for a number of reasons. Higher chip counts have a number of results, all of them bad:
Larger computers. With conventional design, Macintosh would have been much, much larger.

Higher manufacturing costs and, therefore, higher retail costs to buyers.

More heat produced by the computer, which leads to internal fans (which cost money and make noise) and necessitates design changes to get rid of the heat before it damages the computer.

Larger power supplies to feed all those power-hungry chips. Again, more size, and higher cost.

Reduced speed. The more components for electrons to fly through, the longer their trip will take. Performance suffers.

Lower reliability. The more components, the more likely that one or more will fail. Not a fun occurrence.

A good hardware designer is an artist, not merely a technician. To minimize the number of chips in a design, without sacrificing capabilities, is a tough task. The hardware artist does this, and more: he (or she) uses fewer chips, but uses them to produce more capabilities.

This is less is more with a vengeance.

There are a wealth of chips to choose from in creating a design. Each chip has certain abilities and certain characteristics. The chip manufacturers publish data sheets that detail the characteristics of each chip. Designers use that information in creating boards.

At Apple, the idea has always been taking the spirit, rather than the letter of the data sheets.

In many cases, this translates into using chips in new ways, for new functions, and by combining chips with other chips in spectacular ways. It also results in vastly greater reliability and often much greater speed.

Steve Wozniak is legendary at getting the most from every chip. With the release of Macintosh, Burrell Smith is also suddenly legendary, at least in the engineering community (down the road about 10 miles, take a left...).

Another mark of a good designer is knowing what to do in hardware and what to do in software. Often, the same function can be accomplished either way. But there are, as always, trade-offs. Hardware is simpler, and often faster, but brings you back to more chips. Software is more easily changed, but often slower. And software consumes memory that might be used for other purposes. And more chips needed to hold on-board software is, well, more chips.
These are the kind of questions that hardware designers agonize over.

Sound generation is a good example. The easy way is to add a big glop of chips somewhere on some board. The hard way is the perfect combination of software and a minimum number of extra chips. But there's a best way: come up with an entirely new method to produce sound, without slowing up any other functions of the computer, at a minimum of cost, with a maximum of function.

As always, Smith found the best way. The result is that Macintosh can produce four-voice sound; four notes played simultaneously, without slowing down other operations, or adding a mess of expensive chips. Cheaper, faster, better.

**The Motherboard**

What we think of as the actual computer can be found on one board placed horizontally at the bottom of the Macintosh case. The connectors on the back panel run directly off this board.
The motherboard is where all the digital processing takes place. Near the center is the MC68000 microprocessor, the heart of the machine. Nearby are two chips that hold the 64K bytes of Read-Only Memory, or ROM, where all the internal programs are stored. Unlike RAM memory, programs stored in ROM can't be erased—they're there for good, or until replaced with new chips.

The ROMs contain a storehouse of tightly compacted routines that control the disk drive, the Macintosh display, windows, text editing, fonts, and a wealth of other functions. In the months and years to come, hackers will surely spend thousands of hours exploring the routines packed into ROM. They'll be well-rewarded for their efforts.

Mac's RAM memory is also on this board: 16 chips that hold over 130,000 bytes of Random-Access Memory; abbreviated, by convention, as 128K. The RAM memory chips are late-model, high capacity, expensive memory chips.

Two other chips, a Serial Communication Controller (SCC), and a Versatile Interface Adapter (VIA), provide a variety of functions in two small packages. The SCC is the brains behind the two serial ports on the back panel; both sophisticated high-speed ports capable of communicating with printers, plotters, modems, networks, or other yet-to-be-developed serial devices.

The VIA also works for a living, mediating between the 68000, the Macintosh keyboard, the mouse, and Mac's real-time clock.

Our PALs

Of all the clever hardware features of Macintosh, possibly the cleverest is Burrell Smith's use of PALs. You may remember PALs from the book *Soul of a New Machine*. In that book, PALs were a necessary item in creating a new design of computer.

The PALs in Macintosh mark one of the first time PALs have been used in small computers.

In essence, PALs (for Programmable Array Logic) are equivalent to entire circuit boards on a single chip. In one sense, PALs are programmed much like ROM chips are programmed. In another sense, PALs are designed much like entire circuit boards are designed. PALs are a blank slate that can be written-on in an almost endless number of ways.

The trick here is placing a maximum of function on each chip. Doing so lowers the number of other chips that would otherwise waste space, make heat, and cost money.

Clever designers can radically lower chip counts by using PALs.
In Macintosh, the six PAL chips replace, conservatively, 35 or more conventional integrated circuits. That's a lot of chips.

The PALs do things in Macintosh that only computer engineers can understand and love: bus management, timing signal generation, analog signal generation, linear address generation, and more. But let's not go into it.

As Smith explains it, the PALs can be thought of as fast servants to the wise old man of the board: the 68000 processor. Six additional PAL processors, each racing along at millions of operations per second. Not as smart as the old man, but blindingly fast.

In all, the board is deceivingly simple in appearance. It's only on reflection that you realize that this one board contains:

- All system processing and memory (memory is often an additional card).
- Two advanced serial communications ports (often one or more cards).
- Additional interfaces to the keyboard, mouse, and external speaker (often another card).
- All sound generation circuitry (and don't forget: voice is sound, too!).
- All disk controller circuitry, including everything necessary for use of an additional disk drive (often...).

The Analog Board

Only one board remains: the less exotic analog board. The board is mounted vertically within the Macintosh case, directly under the top left row of vents. If you've noticed that more heat flows through the left vents than through the right vents, here's the reason. Don't cover the vents.

The analog board holds the Macintosh power supply, the internal speaker, and circuits necessary to control the video display.

Analog circuits are often overlooked when judging computers, but they're extremely important. In particular, the power supply design determines how long, and how well a computer functions. Typically, power supplies are large, bulky items within a computer. Macintosh, instead, uses an advanced, lightweight switching power supply that delivers a clean stream of power to all other components, and sips a mere 60 watts of power—about the same as one average lightbulb.
Power supplies have other important duties. Unfortunately, Apple engineers aren't controlling the nation's electrical generators. As a result, the power available at your socket might be a little higher than advertised, a little lower, or not quite as clean as most computers would prefer.

Macintosh has built-in protection for each possibility, and a number of power filters for still more protection.

Macintosh can be used with power that isn't perfect, yet still perform flawlessly; a comforting thought if your electricity isn't the best.

The analog board also contains circuits to control the video display; in many respects the same circuits found in many more traditional monitors. Everything found in a stand-alone monitor is squeezed into the analog board, and designed in a manner that allows Macintosh to write information to the screen much faster than other computers.

Most computer designers cheat a bit when it comes to video displays. Because power comes from the socket at 60 hertz, the cycling of the AC current is used as a reference for display purposes. A fine idea if the current really is 60 hertz.

Macintosh, instead, controls the video display using a rock-steady crystal. The result, again, is dependability, and a crisp, black-on-white image without the annoying flicker or haze found on other monitors.

Because the display is crystal-controlled, International Macs need few changes to sparkle with different forms of power, and none of those changes have anything to do with the video display.
Building Macs

The above would be all just engineering showing-off if the computer couldn't be manufactured cheaply, and used reliably.

Manufacturing considerations were weighed and enhanced during every step of Mac's design. So were reliability considerations.

The Mac motherboard is a four-layer board: an advanced design that puts the electrical ground and power within the board itself, assuring an even ground potential over the entire board, and immediate, even power to components that need the juice.

This design results in vastly lower RF emissions than found in conventional designs, and greater ruggedness.

When the FCC certified that Mac wouldn't disturb other electrical devices, the computer sailed through every test.

Absent on the Macintosh motherboard are dip switches, potentiometers, and other controls often necessary to set or tweak computer behavior.

Macintosh needs none of these. There is nothing inside the cover that will ever need initial setting, or subsequent adjustment. When Macintosh leaves the factory, it's ready to be used by you, with no configuring or adjusting, or optimizing. Just turn on the switch.

Ready to "Go out and live life as a Macintosh," as Smith says.

Details Forthcoming

The above is only a quick look at what's inside. For many readers, it should be enough; it's not necessary to know anything about what's inside Macintosh to use the machine to the limits of its (and your) ability.

For those who want more, look for an in-depth reference guide to be published soon by Apple. And expect many books to detail the inside of Macintosh in mind-numbing specificity.

If you're a hardware buff prepare to be thoroughly amazed. If you're not, it's enough to know that Macintosh's design is both wondrous and utterly reliable.

Just as you would have guessed.
23. Inside Macintosh ROMs
The Macintosh Read-Only Memory contents have been referred to over and over again within this book. The references are unavoidable; if Mac has a personality, this is where the personality is stored.

**How It Was Done**

Everything in ROM levered off one sneaky idea. The sneaky idea was made possible by Motorola, manufacturers of the Mac MC68000 microprocessor.

When the 68000 was designed, engineers at Motorola had to create an instruction set: a set of machine language instructions that would ADD, SUBtract, MOVE, and do all the other things that programmers needed and wanted.

But Motorola didn't specify every instruction. Instead, they left two instructions undefined.

In normal operation, the processor looks at each instruction, and then does what the instruction calls for. But if the instruction begins with the certain arcane, undefined, hexadecimal characters, the processor throws up its hands (literary license), and deems the strange characters to be an exception.

Exceptions are similar to what other processor manufacturers call, interrupts. The processor is interrupted by something, deals with it, then goes back to what it was doing prior to being interrupted (you know the feeling). Many times, individual keystrokes on a keyboard are used as interrupts.

So the processor says "I don't know what this is" and hands the instruction over to something called a *system dispatcher*.

In effect, the chip designers left a trap door for programmers to create their own instructions.

The Apple programmers weren't satisfied with creating a measly two new instructions, so they blew the trap door open and sailed a fleet of new instructions through.

Something over 480 new instructions. Each one fast, small, and dedicated to doing one thing as well as it can be done, in a completely Macintoshian way.

When an undefined instruction appears, the system dispatcher uses a *look-up table* to determine which of the new instructions is being requested.

The look-up table scheme will also make it easy for Apple programmers to make changes in the future. New routines can be placed (from disk) into RAM memory. An entry in the look-up table can then
be changed to point at the new routine in RAM, rather than the former location in ROM.

### Accessing Routines

Because each separate ROM program has a name, life is greatly simplified for programmers. Programmers need not re-write every tricky piece of program code that manipulates the Macintosh screen. Instead, they write their programs and, when needed, make calls to the routines in ROM. To make a ROM call, all that programmers need to know is:

- The name of the routine to call,
- The information that needs to be supplied, and
- What value is returned (or what happens) when the routine executes.

Imagine a little kid named Andy. Andy is a go-fer: go for this, go for that. Andy isn't the brightest kid in the world, but he's good for getting things. If you want coffee, here's what you do:

- Call Andy.
- Tell him what you want: coffee.
- Give him the necessary parameters. In this case, let's say the parameters are: cream and sugar.
- Wait a brief time (Andy's amazingly fast).
- Take the coffee, creamed and sugared, from Andy.

Essentially, this is all that programs need do to access most Macintosh features. Add the proper name to the program, then pass along the necessary information. No need to memorize a list of byte addresses where each routine can be found—knowing the name is enough. Slick, isn't it?

### Breaking Down ROM

The ROM programs extend from location 0 to location 65,535. Over
65,000 separate bytes of read-only memory (also known as 64K of ROM); a large amount in comparison to other microcomputers.

For now, the new instructions are contained in ROM. The routines in ROM can be grouped, loosely, into three areas of endeavor:

- The Macintosh operating system,
- QuickDraw routines, and
- The User Interface Toolbox.

Each group of individual assembly-language routines takes up about one-third of the total 64 thousand bytes of ROM room.

The Operating System

The operating system routines in ROM do many of the traditional operating system tasks, with a few nifty wrinkles.

Low-level device drivers provide the information and means necessary for the computer to communicate with the keyboard, mouse, disk drive, and other devices. To the operating system, almost everything is a device, whether it is an actual physical device (like a disk drive), or a conceptual device (like a file of addresses). This consistency probably won't be appreciated by typical Macintosh users, but provide programmers with easy ways to write applications that work with a number of different devices.

A memory manager controls what's in RAM, and allows pieces of programs to be relocated, when needed, to different areas within memory: a powerful feature that guards against file fragmentation, a condition that is both messy and which slows performance.

Then there's Input/Output: the mass of routines that make sure that information gets where it's supposed to go. The plus here is asynchronous I/O. Roughly speaking, asynchronous means that the processor can say "Get me this from the disk," then go on about its business until the requested information shows up. Faster operation results from not having to wait for I/O calls to be processed.

A fast and simple file system is also part of the operating system. The file system permits use of long file names, and adds a tag to every block of data on the Macintosh disk, insuring that a file directory can be re-built, even if destroyed.

The file system permits use of various sizes of volumes (usually disk drives), and files. File and volume sizes may be as large as 32 megabytes—a tremendous boon when hard disks arrive.

Also within the operating system is the Macintosh Event Man-
To Mac, everything is an event, whether it’s a mouse button click, a double-click, a drag, or a keystroke. Your pressing of a single key is, to Macintosh, an Event!

Another Event Manager, this section found in the Toolbox, serves as an intermediary between the low-level Event Manager in the operating system, and the application program currently underway.

**QuickDraw**

Everywhere you look in Macintosh, there’s QuickDraw. This pervasive set of routines fill about one-third of ROM. QuickDraw has a hand in drawing everything you see on the Macintosh screen, and, since everything is drawn, it’s impossible to escape QuickDraw when using Macintosh.

Not that you’d want to. The existence of the QuickDraw routines inside every Macintosh, and the fact that they’re easily accessible by programmers, ensures speed, consistency, and well-behaved screen displays.

All that’s necessary, for programmers, is delving into the final third of ROM.

**The Toolbox**

The Toolbox is, truly, a Toolbox for programmers. This is where the good stuff can be found, all nicely arranged, labeled, and laid out for easy programming. The Toolbox includes:

- The Resource Manager
- The Font Manager
- The Toolbox Event Manager
- The Window Manager
- The Control Manager
- The Menu Manager
- The Dialog Manager
- TextEdit
- The Desk Manager
- The System Error Handler
If this weren't enough, the Toolbox also contains a number of handy utilities that help with arithmetic, manipulation of alphabetic strings of characters, and logical operations on single bits within memory.

Each Manager is a complete set of procedures and functions that together manage one facet of Macintosh. The Dialog Manager, for example, sets the sound level of the beeps associated with the Dialog Box, creates and disposes of Dialogs, handles Dialog events, and also provides the same services for Alert Boxes.

The text editing features we discussed earlier are also found within the Toolbox.

Although it's difficult to single out any particular Toolbox manager for extra praise, the Resource Manager deserves much of the limelight. The Resource Manager allows programs to break out character text and program data from the main application program. As a result, to create a French version of an application program requires only changing the information controlled by the Resource Manager, not re-writing the entire program. A tremendous aid for programmers.

The Toolbox also is the hand behind data transfer within Macintosh, whether it appears via the Clipboard, the Note Pad, the Scrapbook, or merely from Cutting information off the desk accessories.

Two data types, ASCII text and QuickDraw pictures, are defined by the Scrap Manager. Programmers with more specific needs are able to create their own custom data types, which can then be Cut, Copied, and Pasted, as easily as other Macintosh data types.

### Digging Into ROM

The contents of ROM deserve, and will surely get, entire books devoted to their uses.

For now, programmers itchy to get their hands on the Tools will need either Lisa Pascal or an assembler. Instant Pascal provides most of the more common, and most used features within Macintosh ROM, but not everything.

Full-featured Pascals that encompass every ROM routine as pre-defined functions and procedures should arrive soon. Also rumored is a C compiler that would give C programmers easy access to ROM routines. MacBASIC, when it arrives, will also provide a sub-set of ROM routines.

The contents of the Macintosh ROMs are reason enough to learn a high-level language, if you haven't already. But the existence of the Macintosh ROMs ensures that you'll never need to know a language to use Macintosh.
24. The Motorola MC68000 Microprocessor
Like all devices, microprocessors are designed to be used for certain purposes by certain groups of people.

The earliest true one-chip microprocessors, like the Intel 4004, lacked both the size and the speed for general purpose computer use. They were fine for hand-held calculators and other less demanding functions.

The early microprocessors also had limited instruction sets. They were designed, essentially, to replace hardware that consisted of logic circuits. Possibly because of this, they were hardware oriented and awkward to program. They were limited in the amount of memory they could address, and the number of other devices they could work with.

In many ways, more is better when it comes to microprocessors. Computer manufacturers want more functions on a single chip; smaller overall size and reduced power consumption. Hardware designers want more flexible methods to work with circuits and peripherals such as keyboards, disk drives, and printers.

Programmers are the most difficult group to please. They want more of everything: more instructions, more registers to manipulate the instructions, and more memory to store the instruction results, program data, and the programs themselves. And they want speed, speed—the more the better.

Programmers also want size. In this case, size refers to binary groups that the microprocessor works with; how large each instruction or chunk of data is, expressed in groups of individual bits. Here's a bit: 1. Here's another: 0. (It should be remembered that bits aren't numbers, only states; one state is on, the other off, an important distinction. The states may be combined to represent numbers, but they may also represent alphabetic characters or other symbols.)

The now ancient 4004 microprocessor worked with 4-bit units; most 8-bit microprocessors, such as the 6800, 8080, Z80, and the 6502 found in the Apple II (and IIe and III), work with 8-bit bytes of information.

Here's a typical street-variety 8-bit byte:

00101101
A larger unit size has many results, almost all of them favorable to programmers. Instructions can be more complex and powerful. More memory can be quickly accessed. Very large (and very small) numbers can be conveniently manipulated, and overall speed is greatly increased. Often the net increase in speed is more than double that of 8-bit microprocessors.

The success of Motorola's MC68000 microprocessor can be laid to the fact each of those groups did, indeed, get what they wanted. Programmers got the most of all.

The discussion that follows won't tell you everything about the MC68000, or even enough to begin programming the 68000 in assembler. Hopefully, though, it will give you a fix on the device, instill an appreciation of its power and convenience for programmers, and prepare you for a more in-depth study of the 68000. (A number of excellent books covering the 68000 and programming the 68000 in assembly language are now available.)

Now let's take a closer look at the MC68000. Readers with limited technical backgrounds may continue at their own discretion.

**Architecture and Design**

The Motorola MC68000 was designed to be a flexible, general-purpose microprocessor. As we noted, it was designed, from the beginning, by programmers with the needs of other programmers in mind. Some areas of particular interest are:

- Architecture
- Addressable memory space
- Data sizes
- Instruction set
- Number and type of registers
- Speed

The 68000's general purpose design resulted in a device that can be used in a variety of ways.

- The 68000 supports large applications that require vast address space (over 16 megabytes directly addressable).
• The 68000 provides fast instructions that make the most of small numbers and memories.

Other special features support multiprocessing, multi-tasking, and memory management units. Separate user and supervisor modes are included.

The general purpose design is also reflected in Motorola's decision to employ microprogramming rather than random logic. Although the reasons behind that decision are many, the effect is a design that can be easily improved, updated, or extended.

The topic of microprogramming is beyond the scope of this book. If you're interested in this truly low level topic, an excellent place to begin is with the Scientific American article listed in Further Reading.

Consistency or orthogonality is one of the 68000's strongest features. The terms mean that the processor's form and functions are similar. There are few special cases. If one operation works one way, other operations usually will behave the same way.

All data registers function identically. All address registers function identically. Any data register, or any address register, may be used as an index register. The registers are not dedicated and limited to use by certain instructions. Instead, there are two general register types: data registers and address registers.

Arithmetic and logic operations are also general. They all operate in a similar fashion, with consistent results. If the ADD instruction does certain things, the SUBtract instruction behaves in a similar manner. (100% consistency, however, is difficult to achieve. Not surprisingly, many of the Apple programmers would design the chip a bit differently, if they could. They admit, though, that the 68000 is an excellent processor for many reasons, consistency among them.)

Operations can be performed on the following data types:

• 4 bit BCD digits
• 8 bit bytes
• 16 bit words, and
• 32 bit long words

Instructions are also provided to test, set, clear, or change individual bits quickly and simply.

Despite a set of addressing methods that appear formidable, working with memory space is accomplished in a straightforward manner. Memory space is not segmented, paged, or banked. The entire address
space is a linear sequence of 8-bit bytes (words and long words are merely series of bytes). Any location can be simply and directly accessed. I/O is memory-mapped, with address locations like other memory locations. Here again, there are few taboos.

Other microprocessors, such as Intel's 8086/8088, use a segmented memory scheme. Segmented versus flat memory is hotly, almost religiously debated. Here's one opinion: flat is better, if only because it simplifies the programming of what is already a complex device. Feel free to agree or disagree.

68000 instructions, like register use, are also fairly general. In particular, the MOVE instruction replaces and enhances a variety of instructions used by other processors. Data or addresses are not POPed or PUSHed or PULLed or LOADed or STOREd. Instead, quite simply, they're MOVEd. Essentially, MOVE moves anything anywhere. Like other instructions, MOVE works with operands of either 8, 16, or 32 bits.

Other instructions were also tailored to more powerful, multi-function forms. Programmers benefit from a smaller instruction set with fewer rules to remember.

That last sentence may be somewhat misleading. True, the basic instruction set contains only 56 instructions. When combined with five data types, however, and further combined with the 14 addressing modes available, the number of actual instruction combinations tops 1,000! That should be more than enough general instructions to satisfy the most dedicated programmer.

Registers

Figure 24.0 details how the 68000 appears to the programmer. If you're familiar only with register designs of 8-bit processors, it might be well to gaze at the diagram for a few minutes.

Your first conclusion is that the 68000 is not a 16-bit processor. Correct. Internally, it's a 32-bit machine. All registers are 32 bits wide. The 16-bit status register is the lone exception.

Data registers can store 8-bit bytes, 16-bit words, or 32-bit long words. Specifying the desired data size is easy. Using the assembler opcode instruction ADD as an example we have:

\[
\begin{align*}
\text{ADD.B} & = \text{ADD 8-bit byte} \\
\text{ADD.W} & = \text{ADD 16-bit word [the default size]} \\
\text{ADD} & = \text{ADD 16-bit word [default size, remember?]}
\end{align*}
\]

\[
\begin{align*}
\text{ADD.L} & = \text{ADD 32-bit long word}
\end{align*}
\]
All opcodes are one or more 16-bit words. Complex opcodes may contain ten or more 16-bit words.

Address registers handle either 16 or 32 bit addresses. In reality, all addresses are 32 bits wide, but are sign-extended if only the lower bits are specified. The registers may be used as base address registers, point to stacks in RAM memory, or hold temporary address values.

The combination of eight data registers and seven address registers often results in simpler, more compact code. Data seldom needs to be shuffled to free up a register for other use. Also, more operations can take place internally, reducing memory accesses that slow program execution.

The status register wasn't overlooked during the 68000's design. The register consists of two bytes: a system byte and a user byte. The user byte contains the expected bits to indicate Carry, Overflow,
Negative, and Zero results. It also contains an Extend bit for use with multiple precision arithmetic. Some instructions set the Carry bit, others the Extend bit; the result is more precise, less troublesome programming.

The system byte contains three bits that serve as an interrupt mask for up to 7 levels of prioritized interrupts. Another bit indicates supervisory mode for multiprocessing or multitasking applications, and one bit enables a single step trace during program execution. The trace mode is one of many debugging aids available to the 68000 user.

The 32-bit program counter is responsible for the vast addressing range of the 68000. A little binary math determines that the maximum number of bytes addressable with 32 bits is 4 gigabytes (4 billion bytes). That should be enough for most purposes.

Actually, only 24 of the 32 bits are presently brought to the outside world through address signal lines, although the entire 32-bit address is held in the program counter. That smaller number allows 16 megabytes of memory to be directly addressed. Still enough for most purposes.

Two 32-bit stack pointers are also shown in Figure 24.0. Only one of the pointers is active, at any given time, as either a user or supervisor stack pointer. Never both at once. The user stack pointer holds addresses for subroutine returns; the supervisor stack pointer saves needed addresses and the contents of the status register.
Speed

The 68000's impressive speed comes from a number of factors.

One obvious factor is clock speed. The 68000 operates at a fast 8 MHz. For comparison, the Apple IIe's 6502 trudges along at about 1 MHz. Most Z80's operate at 2 or 4 MHz.

Another reason is the 68000's 16-bit wide data bus, which transfers twice the data of 8-bit processors and often doubles processor throughput.

Still another factor is the 68000's semi-intelligent "pre-fetch" queue, which readies or fetches the next opcode to be processed while the current opcode is being executed. The pre-fetch is smart enough to keep the data bus busy about 90% of the time. Having the next instruction queued up to be processed greatly increases program speed.

We're not done yet. The large number of fast internal registers play a starring role in the 68000's speed. With clever programming (and even not-so-clever programming) typical programs zip along.

It's difficult, though, to generalize about instruction speed. The minimum time for a 68000 instruction to be executed is 4 clock cycles. Complex instructions may use over 100 clock cycles. That still represents a great savings in time over older instruction methods, which require a number of separate instructions to accomplish the result of a single 68000 instruction.

Benchmarks are frequently used to compare the performance of microprocessors and the computers within which they reside. A test program, usually one that gives the processor a thorough work-out, is run, timed, then compared to other systems. The results are sometimes fair, sometimes grossly unfair, sometimes revealing and sometimes meaningless.

They're also irresistible.

Some of the most extensive sets of benchmarks have been published in BYTE. The test program was The Sieve of Eratosthenes, a prime number generator that requires intensive processor operation. The 68000 was speedy indeed. Here are some selected results:

<table>
<thead>
<tr>
<th>Processor</th>
<th>Language</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>68000 8 MHz</td>
<td>Assembly</td>
<td>0.49</td>
</tr>
<tr>
<td>Z8001 5.5 MHz</td>
<td>Assembly</td>
<td>1.10</td>
</tr>
<tr>
<td>8086 8 MHz</td>
<td>Assembly</td>
<td>1.90</td>
</tr>
<tr>
<td>8088 5 MHz</td>
<td>Assembly</td>
<td>4.0</td>
</tr>
<tr>
<td>Z80</td>
<td>Assembly</td>
<td>6.8</td>
</tr>
</tbody>
</table>
Before you start telling your friends that the 68000 is the fastest of them all, it should be noted that some of the giant mainframe computers (with prices in six figures and beyond) come in with times like this: 0.0078 seconds.

Nevertheless, the 68000 is fast, for all the reasons given above. At full bore, the 68000 can process over 1 million instructions a second! That speed can be used by good programmers to good advantage. Superb programmers can use the 68000's speed to wondrous advantage, as demonstrated whenever a Macintosh is switched on.
Section I: Operation
Section II: Software
Section III: Programming
Section IV: Hardware
Section II: Epilogue
25. Hints, Tips, and Advice
This chapter is a grab-bag of thoughts about Macintosh, unsolicited advice, hints, and a few tips. At the top of the unsolicited advice queue is:

**Attitude**

What is the proper attitude to take in relation to an Apple Macintosh computer? Zest!

Granted, Macintosh is the easiest to use computer ever marketed. Still, as you’ve found, there is a lot to the machine; a variety of ways to do things, and an almost bewildering variety of things to do.

The process of becoming adept at computers is not unlike learning to fly, handling a fine sports car, sailing, or learning the ins and outs of a full-featured stereo system. Unlike flying, though, it’s always possible to walk away from a Macintosh crash, should one occur (though it probably never will).

Reading this book, or any book, will never give you a precise understanding of Macintosh. For that, it’s necessary to use the machine on a frequent and thoughtful basis.

The best advice is this: try out everything. Pull down every menu and try out each command. Practice maneuvering the mouse. Experiment with windows and clicking. Practice Renaming, Removing, and every possible way of moving anything to everywhere.

Fool around with the keyboard. Display and print the alternative characters that the keyboard produces. Discover where Enter works and try out Tab in different situations.

In particular, learn the shortcuts. Spend a bit of extra time using keyboard commands rather than selecting from menus. If needed, write down the command combinations and place the paper near your keyboard. The time spent memorizing the command letters will save time later, if you’re a frequent Mac user.

Double-click, whenever possible, rather than using the menu. Practice Shift-clicking to make extended selections.

When editing, use double-clicks to select words instead of dragging. Use drag-backspace with abandon.

For serious experimentation, always first make a duplicate of your disk. Then, perform disk experiments without a care. See what happens when the disk becomes full. Practice saving to and loading from disk, in as many ways as possible.

Experiment lustily with the Finder. Practice throwing icons in the Trash and then retrieving them. Put icons in other icons. Duplicate a raft of Empty Folders, then re-title them all, open some of them,
and practice shuffling windows from front to back. Drag icons into windows to make subdirectories, then close the window and drag that icon into still another icon, making a sub-subdirectory.

In word processing programs, give each font a workout. What fonts look best for what purposes? What size looks best for each font? What attributes best fit each font? What justifications look best for which documents? What headers and footers look particularly elegant?

To give your eyes a break, process words in a large, bold font size. When you’re finished writing, select the entire document and bring the font size down to the optimum size for printing (your choice of size).

In MacPaint, give everything a try. Make a drawing, then refine it with Fat Bits. Change a few patterns. Use the selection rectangle with the Command key, and with the Option key, and with the Shift key. What are the differences? What’s a good use for each? Become adept at lassoing and determine when best to use your skill.

Try Brush Mirrors; try each brush size with a variety of patterns; give each tool a workout.

Try your hand at creating pre-printed forms. Volunteer to make the next newsletter for your favorite organization (they’ll all probably rave at the results).

Again, try every feature in every program. Don’t spend all your time working with Macintosh; allot a good amount of time merely for exercising the machine.

Always be on guard for insights. Notice how everything really does work like everything else. Use that knowledge to good advantage; to speed your work, and give you new ideas.

---

**Workstyles**

Many applications programs will give you two or more windows at once to work with. Often, you have the choice of having one or more open at any time.

If so, consider keeping all windows open. Then re-size your main window to fill almost the entire screen, leaving just a sliver to let windows underneath peek through. To switch active windows is then only a matter of a single click.

Your desktop is uncluttered, but other windows are always available, fast. One large window also takes maximum advantage of the nine-inch Macintosh screen.

Or, you might prefer two windows, side-by-side, splitting the display into two; the other windows could be open but out of sight.
Don’t worry about having too many windows open at once; it’s not a problem. If anything, it’s a convenience. Just try to avoid display clutter by keeping only one or two windows large enough to be useful.

The same applies in MacWrite. It’s possible to use Show Clipboard and Hide Clipboard from the Menu Bar, but it’s much easier to keep the Clipboard open and underneath the MacWrite window. Just keep a narrow edge of the Clipboard window visible, for fast access. In time, you’ll probably find that looking at the Clipboard window is unnecessary—you know what’s there.

### Traveling with Mac

Is Macintosh a portable computer, a transportable computer, or what? Can you travel with Macintosh? More important, should you travel with Macintosh?

As always, Mac is in a class by itself. It’s too large to be anyone’s idea of a lap computer, and it’s a bit bulky to be termed a portable.

Maybe Mac is a transportable, a catch-all term for any computer that is sort of portable. After all, the machine is only 20-some pounds—about the same weight as a sewing machine or portable television. And you’ve probably seen the variety of Mac packs available for totting Mac around.

The simple answer is: yes, Macintosh is portable. But traveling with Macintosh won’t be the most pleasant experience of your life.

The ruggedness of Macintosh is the least of your worries. Just don’t drop Mac, and you’ll probably be okay. Everything inside Macintosh is firmly secured, and can take a fair amount of jiggling and shaking. The monitor tube is the delicate component, even the tube may be more sturdy than you’d think, but don’t go out of your way trying to prove that statement.

Macintosh is not sturdy enough to safely check through airline baggage, unless the machine is carefully re-packed in the original cartons, with all the original protection carefully in place. Even then, the prospect of airline handling is a trifle unnerving.

Good thing Mac fits under airline seats, huh?

Wrong.

Macintosh will fit under some airline seats. Not all, some. Nor will it fit into the overhead bins on most airlines. The Mac Pack is almost exactly one inch too wide to fit in the bins. (Sounds like the voice of experience, doesn’t it?)

The truly unpleasant aspect is this: you won’t know if your Macin-
tosh will fit until it's too late; usually about the time they're closing the doors. Soon after, a Flight Attendant will come strolling down the aisle to find you struggling to fit your Mac under the seat. And failing.

Not a pleasant scene. When this happens, the Flight Attendant may have mercy on you, stick the Mac in a suit closet, or an empty galley bin, or let you strap the offending Mac Pack in alongside your seat (if it's empty, that is, even though it's Against The Rules).

At the worst, you may have to leave the plane, check your Mac through the dreaded baggage department, or leave Mac behind, or stay behind yourself: Macintosh in hand.

The worst can happen. It probably won't, but it can.

There are no good solutions to all this. The best is the original carton scheme, accompanied by much nervousness and trepidation.

Oh well. You don't really need to travel with Macintosh, do you?

---

**Extracurricular Activities**

On to a more pleasant subject. Work at increasing your knowledge of Macintosh. We noted users groups earlier as a source of programs and programming information. Their value, though, extends beyond those areas. See what other Mac owners are doing with their machines. Pick up hints, get and give advice, get involved with Macintosh.

Subscribe to one or more Mac magazines. And don't just read—write. Share your thoughts with the editor, other readers, or consider writing an article; review a product, rate an accessory, or relate your experiences, pleasant or unpleasant, with Macintosh. You might even get paid! You might even get paid well!

It's been estimated that each computer owner buys an average of 11 computer books. Try to keep the average up; you learn, authors earn; everybody wins. If you must, go to the library and check out a few of the books listed in Further Reading.

Drop in on your Apple dealer once in a while. See what's new, get a few demos; bring your checkbook or Apple credit card.

---

**Change of Setting**

Where is your Macintosh? In the study? At work? In the basement? On the kitchen table?

A change of location might be good. If your Mac is at work, bring
it home; give yourself time for experimentation, and give the rest of your family a chance at the machine.

Remember the footprint; Macintosh requires little desk or table space. Consider plunking your Mac on your end table and relaxing in your favorite recliner while computing. If the table isn't a handy location for the mouse, place a thin board or pad on your lap and roll away; with practice, an area only a few inches wide is room enough for all mouse motions. (You can even use your leg as a mouse pad in an emergency.)

Have an outdoor porch or deck? Run an extension cord out and compute while the sun sets. [But stay away from the Hot Tub.]

---

**Telecommunications**

If you've got the time, and the extra money, invest in a modem. At the very least, get a friend to give you a demonstration of the wealth of bulletin boards, data bases, and other services that lie only a phone call and a phone bill away. If you're sold, get on-line; your first chat via Macintosh with another Mac owner half a continent away will convince you that the money was well spent.
mac presents:
THE MEDFLYs
at a
PUt'NK ROCK
PARY

SATURDAY, APRIL 9
LOS GATOS SWIM + ACQUET CLUB
14700 ORA ROAD
7-12

music starts at eight. as ice picks, bottles, cans, or malathion, please. bring your own mice.

Other Hobbies; In the Future

If you're a typical computer owner (is there such a thing?), you have a number of interests; not merely computers.

Look for ways to involve Macintosh in those other areas. Stamp
collecting is a prime candidate for databasing; many other hobbies also lend themselves to cataloging, sorting, and selection.

Use MacPaint to plan your next garden. Or chart your family tree. Or create designs for stained-glass works of art.

Interested in money: keeping what you have or getting more? Look into spreadsheets and fancy budgeting programs.

In the not too distant future, home controllers will be available for Macintosh. Protect your home, switch on the coffee, run your electric trains, or dazzle the neighbors with your automated Christmas decorations. All with the aid of Macintosh.

In the more distant future, the storage capabilities of Macintosh will be virtually unlimited. Laser disks and other yet-unknown wonders will allow you to store entire libraries for instant access and manipulation. Text-readers will allow you to quickly plug text information into your computer, bypassing both keyboard and mouse. Last year's tax forms? Throw them in Macintosh for help with this year's tax bite.

The potential of computers in general, and Macintosh in particular, are astounding. Explore those capabilities; use Macintosh to its fullest.

The rewards are great. And imminently achievable.

---

**When Mac Is Old and Gray**

Will Macintosh eventually be obsolete? Should you put off buying a computer because they'll be cheaper soon, or they'll be better but no more expensive?

No and no.

The Macintosh will never be obsolete. Nor will any computer that continues to function properly. Antiquated, yes. Obsolete, no.

The difference is this: if a machine does something useful, it has value. A 1961 Ford in running condition has value because it works—it gets you where you're going.

Ten years from now, maybe even five years from now, Macintosh will surely be antiquated. Computers will be smaller, faster, possibly even easier to use (though it's difficult to imagine how that will be accomplished). Maybe even cheaper.

No one can deny that the computer price/performance ratio is improving all the time. Still, it seems that prices, in one sense, haven't gone down at all; you just keep getting more for your money. Same amount of money, though.

People who bought their first computers in 1978 are probably glad
they did. They learned about computers, and the computers performed tasks that served them well. When they bought their next computer (and many did), they carried their earlier knowledge with them.

The same is true today. The Macintosh may be your first computer, or your second, or even your third or fourth. And it's likely that someday you'll buy another. When you do, I doubt that you'll look back on Macintosh, or even Macintosh's price, with regret.

Instead, you'll remember that your Macintosh helped you process words, manage information, learn programming, communicate across town or across country, figure your taxes, balance your books, and just generally make life more manageable, and more fun.

The folly of waiting to buy a computer is that while you're waiting, you don't have a computer. Not having a computer means that life, in many ways, is more difficult than it needs to be.
The People Who Made Macintosh
<table>
<thead>
<tr>
<th>Collete Askeland</th>
<th>Susan Kare</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill Atkinson</td>
<td>Guy Kawasaki</td>
</tr>
<tr>
<td>Bob Bailey</td>
<td>Larry Kenyon</td>
</tr>
<tr>
<td>Steve Balog</td>
<td>Betti Kenyon</td>
</tr>
<tr>
<td>Susan Barnes</td>
<td>Barbara Kosikin</td>
</tr>
<tr>
<td>David Beaver</td>
<td>Dan Kottke</td>
</tr>
<tr>
<td>Bob Belleville</td>
<td>Jan Krappel</td>
</tr>
<tr>
<td>Brian Berkeley</td>
<td>Kelly Laccabue</td>
</tr>
<tr>
<td>Mike Boich</td>
<td>Dan'l Lewin</td>
</tr>
<tr>
<td>Mike Boys</td>
<td>Scott Love</td>
</tr>
<tr>
<td>Jeff Bradley</td>
<td>Liz London</td>
</tr>
<tr>
<td>Debbie Bress</td>
<td>Sam Lyall</td>
</tr>
<tr>
<td>Peggy Brown</td>
<td>Ivan Mach</td>
</tr>
<tr>
<td>Bill Bull</td>
<td>Jerry Manock</td>
</tr>
<tr>
<td>Pete Burnight</td>
<td>Scott Marquardt</td>
</tr>
<tr>
<td>Tom Burt</td>
<td>Tina Marquez</td>
</tr>
<tr>
<td>Steve Capps</td>
<td>Bob Martin</td>
</tr>
<tr>
<td>Steve Chernicoff</td>
<td>Mary Ellen McCammon</td>
</tr>
<tr>
<td>Debbie Coleman</td>
<td>Blanche McCoy</td>
</tr>
<tr>
<td>Bud Colligan</td>
<td>Joanne McManus</td>
</tr>
<tr>
<td>Mike Collins</td>
<td>Vicki Milledge</td>
</tr>
<tr>
<td>Jerome Coonen</td>
<td>Rosemary Morretta</td>
</tr>
<tr>
<td>George Cossay</td>
<td>Mike Murray</td>
</tr>
<tr>
<td>George Crow</td>
<td>Terry Oyama</td>
</tr>
<tr>
<td>Diane Dalton</td>
<td>Ben Pang</td>
</tr>
<tr>
<td>Donn Denman</td>
<td>Linda Patterson</td>
</tr>
<tr>
<td>Roger Donatien</td>
<td>Steve Quento</td>
</tr>
<tr>
<td>Gene Dunham</td>
<td>Jeff Raskin</td>
</tr>
<tr>
<td>Dave Egner</td>
<td>John Rizzo</td>
</tr>
<tr>
<td>Bonnie Endress</td>
<td>Brian Robertson</td>
</tr>
<tr>
<td>Chris Espinosa</td>
<td>Dave Roots</td>
</tr>
<tr>
<td>Eva Fasano</td>
<td>Caroline Rose</td>
</tr>
<tr>
<td>Gail Fauber</td>
<td>Alain Rossman</td>
</tr>
<tr>
<td>Frank Fennelly</td>
<td>Sam Sanford</td>
</tr>
<tr>
<td>Bill Fernandez</td>
<td>Steve Scheier</td>
</tr>
<tr>
<td>Peter Foley</td>
<td>Rick Schutt</td>
</tr>
<tr>
<td>Hans Gerke</td>
<td>John Scull</td>
</tr>
<tr>
<td>Pete Griece</td>
<td>Rony [Veronica] Sebok</td>
</tr>
<tr>
<td>Dennis Grimm</td>
<td>Hasmig Seropian</td>
</tr>
<tr>
<td>Barbara Grisier</td>
<td>Pat Sharp</td>
</tr>
<tr>
<td>Brad Hacker</td>
<td>Joe Shelton</td>
</tr>
<tr>
<td>Martin Haiberli</td>
<td>Burrell Carver Smith</td>
</tr>
<tr>
<td>Meaghan Hardy</td>
<td>Pam Stanton-Wyman</td>
</tr>
<tr>
<td>Andy Hertzfeld</td>
<td>Lynn Takahashi</td>
</tr>
<tr>
<td>Pam Hillhouse</td>
<td>Paul Tavenier</td>
</tr>
<tr>
<td>Barry Hochfield</td>
<td>Bobby Tilley</td>
</tr>
<tr>
<td>Joanna Hoffman</td>
<td>Denise Tolan</td>
</tr>
<tr>
<td>Diana Hogue</td>
<td>Sandy Tompkins</td>
</tr>
<tr>
<td>Dave Holzer</td>
<td>Joe Valente</td>
</tr>
<tr>
<td>Toni Homewood</td>
<td>Leslie Vanwinkle</td>
</tr>
<tr>
<td>Bruce Horn</td>
<td>Stephanie Wall</td>
</tr>
<tr>
<td>Brian Howard</td>
<td>Bill Wathen</td>
</tr>
<tr>
<td>Carol Jinks</td>
<td>Denise Wells</td>
</tr>
<tr>
<td>Steve Jobs</td>
<td>Linda Wilkin</td>
</tr>
<tr>
<td>Lynneea Johnson</td>
<td>Stan Wilkison</td>
</tr>
<tr>
<td>Carol Kaehler</td>
<td>Tricia Wilcoxon</td>
</tr>
<tr>
<td>Scott Kamins</td>
<td>Gary Williams</td>
</tr>
</tbody>
</table>
Accessories Anything that attaches to the Macintosh except the mouse and keyboard. Accessories are also called devices or peripherals. Accessories available from Apple, at this time, are a numeric key­pad, dot-matrix printer, and additional disk drive.

Accessories (desk) See Desk Accessories.

Activate event [Technical] An event generated by the operating sys­tem program Window Manager when a window changes state between active and inactive.

Active selection The selection now in force. Usually a menu selec­tion, but could be an icon. Highlighted. See Selection.

Active window The window that is open for use. Always the front window. The title of an active window is highlighted, and the title bar and scroll bars are visible when the window is active.

Alert A warning or a report of an error. May be a sound from Macin­tosh's speaker, the appearance of an Alert Box, or both. There are three types of Alert: Stop, Note, and Caution, each associated with a specific icon that appears in the upper-left corner of an Alert Box.
Alert Box  A window that appears on the screen signalling an important message that requires action from the user. Typically, two choices are given: Okay, and Cancel; each within a command button. The recommended choice is boldly outlined. Click within one of the command boxes to select. [Technical] The boldly outlined button is the default button.

Alice  A demonstration program that shows off the lightning-fast graphics capabilities of Macintosh.

Alignment (text)  The appearance of text on a page or screen. Refers to how text, numbers, or data are lined-up in vertical columns. Often used synonymously with Justification, where right-alignment means the same as right justification. [Technical] Also refers to the positioning the read/write head in a disk drive. If the head is out of alignment, the drive functions poorly, if at all. Dealers who service drives typically align drive heads. If your Macintosh is having trouble reading or storing information, bad alignment of the drive may be the problem.

Alpha Lock key  [Technical] A hi-tech way to say Caps Lock key.

Application program  A software program that is used for a particular application, e.g., writing or data management.

Arrow  The usual on-screen representation of the mouse position. The pointer shape is an arrow when over menus, button and dial areas, inactive data, and other cases with the exception of text, structured [cell] documents, graphics, and while long operations are in progress. The arrow always points north-northwest, handy if you're lost in the woods with a Macintosh and a power outlet is nearby.

Ascent line  [Technical] A conceptual horizontal line across the top of upper-case letters that marks the upper limit of characters in any given font and point size. Also called cap line. A typographer's term.

Ascenders  [Technical] The upper, vertical portions of letters such as d, l, and b. A typographer's term.

ASCII  [Technical] ASCII is the agreed upon convention for representing numbers, letters, and other symbols using binary chunks of information. In ASCII representation, for example, the letter A is translated from the binary byte 0000 0001.
Assembly language  A precise but painful way to program computers. See Chapter 17 for details.

Athens  A Macintosh type font. Sans serif. Looks good fairly large and larger; looks not-so-good very small. As always, let your good taste be your guide.

Automatic scrolling  See Continuous scrolling.

Back  Refers to windows that are covered by other windows; they're in back. Being in back also means that a window is inactive, or off.

Back-up  As a noun, a disk that contains information duplicated from another disk; good protection in the event you lose, break, or inadvertently erase [or otherwise foul up] the original disk. As a verb, the process of making a copy of a program, or entire disk. If the disk is valuable, consider keeping the back-up somewhere other than with the original disk—safety-deposit boxes are seldom consumed by flames.

Backspace key  A key on the Macintosh keyboard. Pressing the Backspace key moves the insertion point one position to the left and erases any character located to the right of the new position. After material has been selected by dragging, hitting the Backspace key will delete the selected material. Unlike Cut and Copy, removing material this way may mean it's gone for good; or, you may be able to retrieve the text through a menu selection: Undo Typing. Check your program for details. The key repeats if held down. Also known as a destructive backspace.

Bar (menu)  See Menu bar.

Base line  [Technical] The imaginary line on which the base of capital letters rest. Lower-case letters with descenders, such as the letters p, g, and y, extend below the base line and are bounded on their lower limits by the Descent line. A typographer's term.

BASIC  Acronym for Beginners All-purpose Symbolic Instructional Code. The most popular language for small computers. See Chapter 17 for more details.
Baud  (Technical) A term that refers to the speed of data being transmitted between two computers, or between a computer and another, external, device, such as a printer. Typical data transmission rates employed by modems are 300 baud and 1200 baud.

Bit  (Technical) The smallest practical unit of information within a computer. The elemental atom of computer programming. Bits can be either off or on—1 or 0. Eight bits form a byte, the next larger chunk of information easily manipulated by computers.

Body type  Refers to the size, in points (see definition) of text used in columns of print, such as newspaper columns. Body type is usually 14 points or smaller. May also be called book type. Typographer's term.

Boldface  A visual way to highlight a character or section of text by increasing the density of the letters.

Boot  (Technical) All the necessary things that happen when a computer is first turned on are lumped together in the phrase booting-up, or booting the computer. Often, computers require a particular boot disk to begin operation. With Macintosh, this is known as a start-up disk.

Buffer  (Technical) Refers to an area in memory, or on disk, where information is stored, usually prior to being moved somewhere else. Typically an area in user memory (RAM), where data is stored; when the buffer is full, the data is automatically stored on (written to) disk. This simplifies things considerably for designers and programmers. The process is invisible to users. By the way: the Clipboard is a buffer.

Bug  (Technical) A flaw in a machine or program. Often difficult to find. The first bug goes back to the early days of computing and was, truly, an insect (a moth, actually) that found its way into a computer and died a messy death, causing subsequent agony and consternation. Today's bugs are generally of a more abstract nature, but equally frustrating.

Bulletin board  Electronic message services, usually operated by computer hobbyists. Often full of programs that can be downloaded into your computer, tips, and computer gossip. You'll need a modem, a communications program, and the appropriate phone number (and sometimes a password). Hobbyist bulletin boards are free, except for the phone bill.
**Button**  Two kinds. The button located on top of the mouse selects and gives commands. Buttons found on the display screen allow a variety of commands to be given. Most screen buttons are outlined, with explanations of the command within, or outside the button. To perform the action (or to select the option), place the pointer on the button box and click.

**Byte**  (Technical) A group of eight bits (see definition) that represent one chunk of information in a form easily processed by computers. Memory is also represented in terms of bytes. The Macintosh contains about 65,000 bytes of memory filled with internal programs, 128,000 bytes of user memory, and about 400,000 bytes of storage available on disk—all healthy amounts. The number of bytes available is seldom a consideration until you run out of space.

**C (language)** A modern, popular, advanced programming language. See Chapter 17 for details.

**Cable**  (Technical) Cables usually consist of a number of wires, which may each carry different signals, that are enclosed in a rubberized covering. Often about the same heft as a heavy-duty extension cord. To attach a printer, for example, you'll need the proper printer cable. Unlike many brands of socks, one cable won't fit all.

**Calculator** One of the Desk Accessories supplied with Macintosh. Functions as an everyday four-function handheld calculator. Like all Desk Accessories, the calculator may be used while another window or document is active. Numbers from the calculator display may be Cut or Copied, then Pasted into other applications.

**Call**  (Technical) Procedures and subroutines are called by the main program when their names appear beneath the gaze of the microprocessor instruction pointer. Often, parameters must be passed to the procedure that is called, or unpredictable values are returned and strange things happen.

**Cancel** A button that appears in Dialog or Alert Boxes (see definitions). Clicking Cancel halts a process, or aborts an operation that was about to begin.

**Caps Lock key** Key on the Macintosh keyboard. When depressed, displays subsequent alphabetic characters in upper-case. Caps Lock will not give upper symbols on punctuation or number keys; for that, you need Shift.
Cell  Either 1) a form of document typically containing numeric information, such as a spreadsheet, or financial statement, or 2) an individual area within a document where information is entered, changed, or otherwise manipulated on a cellular grid. The grid may be visible or invisible. Cell documents are one of the three primary Macintosh document types. The other two are Graphic and Text.

Centered  A way to position or print text. Lines are centered when an equal amount of white space appears to the immediate left and right of the line. Good for titles or attracting attention.

Character  An individual unit in a type font. Text can be thought of as a sequence, or list, of characters.

Character style  A consistent set of type font variations, such as italic, bold, underlined, shadow, or outline.

Check box  A control that allows setting of [usually] one of a number of options. The Check box is either X'ed [on] or empty [off]. The selection is made by clicking or pressing within the check box. Current settings are noted by an X within the box.

Check marks  Appear on menus to the left of the item explanation. Used in menus where more than one choice may be made. The MacWrite menu, for example, allows selection of one or more styles of type: italic, bold, and so on. As each style is selected, a check mark appears on the menu next to the selection, and the new style [or styles] is used for subsequent characters. If you have selected both Bold, and Italic, the next text entered will be both Bold and Italic, and the respective items in the Style menu will display check marks.

Chicago  A Macintosh type font. Chicago is used as a system font, the font used on all Menu Bars and Title Bars.

Choose  To make a selection. See Selection, Clicking, Double-clicking, and Dragging.

Clean up  A selection found in the Finder [see definition] menu bar under Special. Arranges the icons displayed in the frontmost window in neat tidy rows.
Click  A way to perform actions or choose selections. The mouse pointer is first positioned at the desired screen location. The mouse button is then briefly pressed once. That is a click. Also see Double-clicking.

Clipboard  An area of memory, represented in a window on-screen, where information can be held before being transferred to a new position in the current document, or in a different document. Used with Cut and Paste operations.

Clippings  Anything found on a Clipboard, naturally.

Close Box  A small box in the upper-left corner of a window. Positioning the pointer within the Close Box and clicking once closes the window, which is then represented by an icon.

Column  Two definitions. Either a horizontal position within a document (as in Row 5, Column 11), or a vertical block of text, such as a column of type.

Command  A way to control the behavior of a program or a computer. Commands are given either by mouse actions, or by use of the Command key and an associated keystroke.

Command key  A key on the Macintosh keyboard marked with a strange cloverleaf symbol. Used to give commands without using the mouse. The Command key is held down while a second key is pressed. The second key invokes a command that is, usually, associated with a menu item. Menu items often have single letters or symbols to the right of the item description. These, when entered via the Command key, produce the same result as the Drag-release method of selecting from menus.

CompuServe  An Information Utility similar to The Source [see Source, The]. Offers a variety of services for communications-minded computer users willing to pay for the service.

Computer  In common usage, an electronic machine that processes digital, electronic information in the form of inputs to produce output. While the actual processing of the information is often exceedingly complex, one hopes that the output is helpful and the input reasonable.
**Condensed** Typeface expressed in a narrow version. Typefaces often come in a variety of versions: condensed, gothic condensed, medium, bold, extra-bold, ultra-bold, etc. The essential characteristics of the face are preserved, but the characters differ in width and boldness.

**Continuous scrolling** Moving text, line by line, forward or backwards, through a window. To begin continuous scrolling, press and hold the scroll arrow at the top or bottom of the Scroll Bar. Scrolling ends when the mouse button is released.

**Control panel** A desk accessory used to tailor speaker volume, desktop pattern, menu item twinkle, time needed between clicks that make up double-clicks, keyboard repeat speed, mouse response, and more.

**Controls** Objects on the screen that you may manipulate, using the keyboard or mouse, to make selections, or otherwise control the action or behavior of the program or computer.

**Copy** Two definitions. 1) The process of selecting portions of a document for duplication within the same (or different) document, through use of the Clipboard. 2) Transferring information, in a more general sense, from the computer to disk drive, disk drive to disk drive, or computer/disk to printer, modem, or any other output device.

**Cursor** (Technical) The Macintosh doesn’t have a cursor, even as an expensive option! You must be looking for Pointer, Pointer shapes or something like that.

**Cut** The act of removing information from a document. What is Cut is placed on the Scrap (or in the Clipboard) and displaces any previous contents of the Clipboard.

**Data** Data is a plural noun, datum is singular. Data is information—and information is usually a preferable word unless specifically referring to the contents of memory locations or areas on disks. Do not use this word in everyday conversation. Also please don’t use the words feedback, interface, or input; there are better choices in each case. People don’t get input; computers do. People get advice, comments, feelings, suggestions, thoughts, opinions, etc. Dismissed.
Decimal tab  A form of tab used to align columns of numbers. Setting a decimal tab ensures that the decimal points all line up in a vertical line, regardless of how many digits are found to the left or right of each decimal point. The decimal tab is found in the MacWrite decimal tab well which appears on the ruler to the right of the well for more conventional tabs.

Default  [Technical] A selection made for you by the computer or program. Format settings in force when MacWrite is opened, for example, are the default settings. Defaults may be changed, but are usually the most common settings, or configurations, chosen when a document is opened, or an application begins. Defaults are used to save the user the bother of making obvious, or common selections before work can begin. Apple prefers the less threatening term Presets.

Descent line  [Technical] A conceptual horizontal line across the bottom of lower-case letters that marks the lower limit of characters with extenders in any given font and point size. See also Base line. A typographer's term.

Desk accessories  Mini-Tools available during use with other documents. The Calculator, Note Pad, Clock, Puzzle, and Scrapbook are Desk Accessories. Information from Desk Accessories may be Cut or Copied, then Pasted into documents.

Desktop  The primary visual metaphor displayed by the Macintosh screen, and the concept underlying all Macintosh software: a desktop containing documents, papers, and accessories such as calculators, clocks, and a telephone.

Device  [Technical] In general, any piece of physical equipment within or attached to a computer; typically external. See Accessories.

Dial  A way to represent, or for the user to control, a setting that is continuous, such as speaker volume. Dials come in different representations, usually a horizontal bar with a white rectangle inside. Positioning the pointer on the rectangle and dragging results in a new setting of the dial. A dial may also represent information about the progress of a task being performed; a document being printed, or being transmitted over a phone line, for example.
Dialog Box  A window that appears on screen when Macintosh requires information, or clarification of a command. May contain a variety of messages, program controls (see definition), or check boxes (see definition). In general, a milder variation on an Alert Box (see definition).

Digit  A single number. Also any finger except a thumb.

Dimmed item  A selection displayed in gray rather than black. Dimmed items may not be chosen or selected. This is for your own good. Usually a menu title or menu selection.

Disabled  Unable to be chosen. Dimmed. See above definition.

Display type  Refers to the size of type, given in points (see definition), that is used for headlines, titles, etc., rather than use in the body (column) of text. Display type is usually considered to be 17 points or greater.

Disk  A circular disc within a rigid plastic case, used to contain information that is accessed by the disk drive within the Macintosh. (Technical) Apple uses a Sony 3 1/2" disk drive that is single-sided and holds 437.5K bytes, unformatted.

Disk drive  An electro-mechanical device within Macintosh that stores information onto (and retrieves information from) removable disks. Additional external disk drives may be connected to Macintosh.

Disk Eject  End the application program. Return to the Finder. Pull down the File menu and choose Eject. Or, in emergencies, switch off Macintosh, hold down the mouse button, and switch on Macintosh, keeping the mouse button depressed. This will always get you the disk.

Display  The Macintosh screen. (Technical) The display is a nine-inch (diagonal measure) monochrome monitor. Images are software controlled, on a grid 512 dots wide by 342 dots high. 80 dots per inch are displayed. The screen is refreshed 60 times a second, a rate that results in a stable, flicker-free image.

Document  Documents are any representation or collection of information or text. Documents are accessed through Windows and come in three primary flavors: text, graphics, and cell (see definition), depending on the application, and the tool required for use.
**Dot-matrix** Refers to the Apple ImageWriter dot-matrix printer. A form of printing where separate pins form a square matrix on a printer print-head. Each pin can produce one small dot when activated. The pins are software controlled and, in various combinations, form characters or graphic symbols. [Technical] Because the Macintosh is always in graphics mode, all characters are graphic representations and the dot-matrix printer always performs in a consistent fashion, regardless of the document printed.

**Double-click** A method to invoke a command using the mouse button. The pointer is placed over the object desired and the mouse button is pressed twice in rapid succession. [Technical] Unless altered on the Control Panel, the downstroke of the second click must follow the release of the first click by no more than 700 milliseconds, or will be considered a separate click by the Event Manager.

**Drag** Moving the mouse while holding down the mouse button. Performs a variety of functions, usually moving a shape or window to a new location, or selection from a menu. Releasing the mouse button then causes the window or shape to reappear in a new location or, in menus, to select an item. If the pointer is dragged through, then outside a menu, nothing is selected. Hint: a quick way to scan the contents of menus on a Menu Bar: drag the pointer, horizontally, across the Menu Bar.

**Drive** [see Disk drive]

**Edit** To change text that has been previously entered into Macintosh.

**Edit menu** Contains commands to act upon text that has been previously entered: Cut, Paste, Copy, Undo. See definitions for each.

**Elevator (scrolling)** The gray area within a Scroll bar. The home of the Thumb: a small, white rectangle that controls what portion of large documents is displayed. Because the thumb travels up and down within it, the gray area is called an elevator. If the scroll bar is horizontal, it's still called an elevator, which shows you the trouble metaphors can lead to.

**Enter key** A key on the Macintosh keyboard. Pressing Enter confirms an entry and, usually, allows advancing the pointer to the next line of text, or cell [in cell documents]. Also duplicates the function of clicking Okay in a Dialog or Alert Box. [Technical] The Enter key sends the control code End of Text.
Extending a selection An alternate form of selecting a portion of a document for further manipulation. The pointer is positioned, then clicked to set an insertion bar. Next, reposition the pointer, hold down either of the keyboard Shift keys, and click the mouse button again. This causes the area between the pointer and the previously selected area to become selected and highlighted.

Face See Typeface.

Fat bits A selection on the MacPaint Goodies menu. Fat bits enlarge a portion of the screen to allow precise manipulation of individual screen elements. Good for precision work, secret messages, font designing, and showing off your Macintosh.

Feature Something an application is capable of doing. Conventional wisdom suggests "The more features the better," which isn't always true; too many features confuse the user and present options that are rarely, or never, used. The key, for software developers, is choosing the right features to include, not an easy task.

Field-motion keys Keys found only on the optional Numeric keypad, which allows pointer movement within Cell documents such as spreadsheets and financial applications. (Technical) This is as close as Macintosh comes to the dreaded cursor key. Field motion keys make sense here, but only when used with highly structured grid-like documents.

Find A command in the MacWrite Search menu that allows a series of characters to be found and, optionally, changed. Choosing Find creates a Dialog Box, which queries with FIND WHAT: for the characters to search for.

Finder Term for the Macintosh operating system. A visual representation of a desktop, with icons that represent files of information, and a Menu Bar available for commands.

Font A complete set of characters in a consistent and unique typeface. Fonts supplied with Macintosh are Chicago, New York, Geneva, Monaco, Toronto, London, and Venice. Cute, huh?

Footer The bottom margin of a page of text. The word margin is only used to refer to the right and left areas of a page.
Format The arrangement of text within a document. Line spacing, justification, margins and tabs all contribute to the Format of a document. In MacWrite, format attributes can be changed by changing settings that appear on the text Ruler (see definition). [Technical] To create track and sector markings on the surface of a disk in preparation for use. Apple refers to this process as Initializing (see definition).

Format menu In MacWrite, contains commands to arrange the appearance of text displayed and determine how text will be printed: Insert Ruler, Hide Ruler, Show Header, Hide Header, Show Footer, Hide Footer, Set Page, Insert Page Break, and Title Page.

Front Refers to the Active window, which is always the frontmost window. Clicking anywhere in a window covered by another window will bring the clicked window to the fore, and make it the Active window.

Geneva A Macintosh type font. Sans serif. Looks good in 12, 14, and 18 points. Also looks good in boldface. A plain looking, all-purpose font.

Graphic A primary type of document used by Macintosh. A graphic document is pictorial in nature, but may include text or numbers. The other types are Text, and Cell (for numerical applications on a grid). [Technical] Just between us, everything is graphics with Mac—a paradigm with many interesting ramifications. Think of a couple.

Gray A color that, in menus or on menu bars, denotes a menu, or selection, that cannot be made. An inappropriate selection. Also, various shades of gray can be represented on the display screen; and printed, if desired. Who says Macintosh doesn't display in color?

Grow Region (Technical) A more precise definition of a size box; more properly thought of, however, as a region. Regions are basic parts of the QuickDraw routines that underlie all objects displayed on the Macintosh screen.

Gutter The white area between the edge of text and the binding of a book; or, the vertical white space between columns of text. A typographer's term. Good newspaper design dictates that gutters between columns should never extend from the top to bottom of a page—headlines or photos should stop gutters.
Hard-disk A form of external disk storage. Hard disk typically store data on non-removable platters rather than removable disks. Storage capacities of hard disks vary, but generally begin at 5 megabytes (the equivalent of about 10 Macintosh disks), and proceed upward to capacities of 10, 20, or more megabytes. Expensive but worth it for large applications such as inventory, accounting, or large databases. See your dealer.

Hardware (Technical) The physical components of a system, as opposed to software, the magnetic form of programming information within the computer (or programs that you purchase). You may also run across the term Firmware for programs that reside, permanently, within the computer. Then there's Vaporware, used to describe products that are announced, but may not (or ever) exist, and Coldware, for stuff you keep in the basement and never use.

Header The top margin of a page; the word margin is only used to refer to the right and left areas of a page.

Highlight Displaying anything on the screen in a visually distinctive manner. Usually done by inverting (reversing black-and-white representation in a given area), or by making the outline of an object, or text, larger.

Hollow cross The pointer shape automatically chosen and displayed during work in a structured document. Structured documents are characterized by cells of information; all spreadsheets and most accounting and data management documents are cell documents.

I-beam Another term for Insertion Point, the distinctive pointer shape displayed during work with text. Text on screen is always entered at the insertion point, which progresses to the right as text is entered.

Icon A graphic (pictorial) image on the Macintosh display that represents a message, an object, or a concept. From a Greek word meaning to resemble, which makes sense. Icons are pictorial abbreviations, if you will. If you study icons, you will be an iconologist, but not necessarily an iconoclast. (Technical) A 32 by 32 bit screen image.

Inactive window A window not in use. Any window not in front is inactive. Therefore, moving a window to the front makes all other windows inactive.
**Indent** To begin, or move, text a specified number of positions from the left edge of a page or margin.

**Indentation** White spaces found at the beginning of a line of text; often denotes the beginning of a paragraph.

**Initialize** To prepare a disk to receive information. Initializing places track and sector markings onto the surface of a disk, in preparation for receiving data. Disks must be initialized before their first use, but need not ever be initialized again. See your Owner's Manual.

**Insertion Point** The position at which text is entered into a document and displayed on screen. The insertion point is moved by means of the mouse. Click once to set a new insertion point. Also called an I-beam because of its characteristic shape. The I-beam shape is automatically chosen and displayed in text documents or areas.

**Interface** (Technical) An electronic boundary between devices, or the electronic circuitry necessary to mate devices, or components.

**Invert** To highlight text or objects by reversing the black or white on-screen display. When a pointer is dragged through a menu, the selections are inverted as the pointer passes through them. To make a selection, release the mouse button while the selection is inverted.

**Italic** A typestyle (see definition). Italic text is slanted to the right, say, 20 degrees.

**Item** A selection within a menu is a menu item.

**Justification** A parameter of text appearance within a window or on a page. Documents may be left justified (straight left edge of text, ragged right edge), right justified (straight right edge, ragged left edge), or fully justified (straight edges left and right). The justification abilities of traditional typewriters begin and end with left justification. Some people think that full justification leads to a more professional appearance, while left justification adds a more personal feeling to text representation.

**Key** As a noun, the various separate pads on a keyboard, each of which transmits one, or more, unique characters or signals. As a verb, to enter in, or key information into a computer.
Keyboard  A necessary component of the Macintosh computer. Used primarily to enter text, but may also be used to deliver some commands and make some selections. See also Modifier keys.

Keypad  An optional accessory to Macintosh. See Numeric Keypad.

Language  A consistent set of rules that, when known and agreed upon, allow communication between beings, or between humans and machines. For the latter usage, see Chapter 17.

Leading  [Technical] The vertical distance between the maximum lower limit of a line of type and the maximum upper limit of the next line. Goes back to the linotype days of printing, when lead strips were inserted to separate lines of type. Pronounced as in "He led the way," rather than in "Leading a horse to water." Typographer's term.

Left justification  The arrangement of characters that displays a clean, straight edge of type down the left vertical column of a page or screen.

LISP  A programming language favored for list-processing and artificial intelligence programming. See Chapter 17 for details.

LOGO  A relatively easy-to-learn computer language. See Chapter 17 for details.

London  A Macintosh type font, otherwise known as Old English. Ornate and heavily stylized; good for infrequent and selective use. Looks best in 18 and 24 points. Great for Christmas cards.

Macintosh  An affordable, advanced computer manufactured by Apple Computer, Inc.

MacPaint  A sophisticated, amazing, fun graphics program for the Macintosh computer.

MacWrite  A word processing application program for the Macintosh computer.

Margin  The number of spaces between the right or left edge of a page (or window) and the beginning of text.

Matrix  (see Dot-matrix)
Memory  The area within a computer, or on disk, where information is stored.

Menu  A listing of items that may be selected by the user. Menus are displayed by pointing to the menu title (on the Menu Bar) and pressing the mouse button. The menu then appears. Keeping the mouse button pressed, drag down through the menu. As you do, the items will be inverted (or highlighted). When the selection you want is inverted, release the mouse button. The item is then selected (and may twinkle at you) and the menu disappears, in a flash.

Menu Bar  A horizontal strip at the top of the display screen, usually covering the entire width of the screen, which contains the names of individual menus. The Menu Bar appears in front of all windows and cannot be covered; only the pointer can appear over a Menu Bar. Menus that may not be chosen will appeared in a dimmed font (see definition). Technical: The Menu Bar is 20 pixels high, white, and surrounded by a black border. Menu titles are always displayed in the Macintosh system font: 12-point Chicago.

Menu Item  A choice in a menu. Possibly text, maybe an icon, maybe both. To select, see Menu definition.

Message  Text found in Dialog and Alert Boxes that offers counsel, advice, or warnings; usually prior to an optional, or necessary selection or command from the user.

Microprocessor  (Technical) A single chip that controls processing of information within a computer. The primary microprocessor within Macintosh is the Motorola MC68000. See Chapter 24 for details.

Missing symbol  (Technical) An undefined character within a particular type font. Often, many symbols are missing and may be defined by advanced users. The presence of missing symbols also provides a convenient method for the operating system to continue operation when confronted with unknown characters.

Modal Dialog  (Technical) A Dialog that requires response from the user prior to doing anything else on the desktop. See Mode.

Modeless Dialog  (Technical) A Dialog that allows the user to respond before, or after, performing any other actions on the desktop. See Mode.
**Mode**  [Technical] A system state that determines how subsequent characters or commands will be treated. Macintosh has been designed explicitly to expunge the difficulties caused by modes. Although Macintosh modes exist, they are short-lived, spring-loaded, and typically not apparent to the user.

**Modem** A device that can be connected to the Macintosh to transmit or receive information or data through phone lines, or through a cable attached to the Macintosh RS-232 serial port. Modems are popular desk accessories. See Chapter 11 for details.

**Modula 2** One of the newer programming languages. Similar to Pascal. See Chapter 17 for details.

**Modifier keys** Keys on the Macintosh keyboard that, when pressed and held down, transmit pre-defined commands or signals to the computer. Macintosh modifier keys are both shift keys, both option keys, Caps Lock key, and the Command key. See individual entries for details of each.

**Monaco** A type font supplied with Macintosh. Sans serif. Looks best when 18 point or larger. An all-purpose font.

**Monitor** The CRT within the Macintosh. At this time, there is no provision for use of an external monitor with Macintosh. Also see Display. [Technical] A small program kernel that works in conjunction with the Macintosh operating system.

**Monospaced font** A method of printing or displaying characters that allows each character an equal amount of space. Thus, a w and an i each are positioned in an equal area. Monospaced fonts are not usually considered as attractive as proportionally spaced fonts, which provide a more typeset appearance. Macintosh fonts are generally proportionally spaced. Numbers, however, are monospaced—which keeps vertical columns of numbers in line, where they belong.

**Motion**  [see field-motion keys]

**Mouse** A necessary component of the Macintosh computer; used to position a pointer on the screen, to see and make various selections, and to perform other actions on the Macintosh desktop.
**Mouse button** A rectangular switch on top of the mouse which transmits commands to the Macintosh. The button is used to click, double-click, or drag the mouse pointer. Each use initiates a different command or action. See definitions of each for details.

**New York** A type font supplied with Macintosh. Looks uncannily similar to the well-known Times Roman font; one of the all-time favorite type faces. Seen everywhere. With any luck, this book will be printed in some variation of Times Roman. Times Roman—oops, New York looks good in a variety of sizes and styles, though is used most often for body type.

**Note Pad** A Desk Accessory. Note Pad allows you to enter text while working with another application. The pad contains eight pages, which may be flipped through; only one font, style, and size are allowed. Text entered on the Note Pad can be Cut or Copied, then Pasted into other applications. In all, a convenient, small-scale word processor.

**Numeric keypad** An optional accessory. Connected to Macintosh, it allows easy entry of numerical information. The preferred method of entering information into cell documents such as spreadsheets.

**Open** The process required to begin work with a document. To open a document, position the pointer over the desired icon and double-click, i.e., press the mouse button twice in rapid succession. Or, click the icon once to select, then choose Open from the File menu. Either method creates a window into the document and makes the window active and ready for use. Opening a document also makes the open document the top or front window on the desktop.

**Operating system** [Technical] Programming code, partially contained in ROM, which performs low level tasks such as communicating with hardware, disk I/O, and interrupt handling. The portion of the operating system accessible to the user is called the Finder [see definition].

**Option key** One of several modifier keys on the Macintosh keyboard. When held down, it gives a different interpretation to characters next typed.
Orphans When a line that begins a paragraph appears as the last line on a page, it is said to be an Orphan. Considered undesirable; better that the bottom footer should be a bit larger and the entire paragraph appear on the following page, yes? The word-processing world has no love for orphans, or for widows either (see definition).

Output [Technical] Computer term for any information or data transmitted from the computer to anywhere else, including the display screen. Commonly, a hard-copy print-out of information.

Overlapping What objects or windows on the display screen often are. The important points are that: the front window is always active or on, and that windows can be hidden from view. Closing a window brings windows further back into view.

Pane The term for each of the windows that result from splitting a single window.

Paradigm A fundamental conception that underlies a possibly complex structure. The central kernel within a concept. New paradigms result in new conceptions. The Macintosh is the result of innovative paradigms concerning computer operation and computer interaction with users. A popular buzzword among computer designers; the original Greek word meant merely an example, or pattern; a fairly pedestrian definition.

Pascal A modern computer language. Popular and readily available. See Chapter 17.

Paste To place information previously Cut from a document into a new position. With some qualifications, areas of text, cells, or graphics may be cut from a document, saved on a Clipboard, then Pasted into a document overseen by a application program of a different nature.

Peripheral [Technical] Any device or accessory externally attached to and used with a computer. Printers, modems, and hard disks are common peripherals.

Pica [Technical] A typographical term; a pica is a unit of measurement of about 1/6 inch. Twelve points [see definition] also equal 1 pica. Headlines and columns of text are often measured in picas. Newspapers that use a six-column format usually set type in columns 16-19 picas wide.
Plain Text  Text that is displayed without use of various typestyles (see definition). Plain Text is how MacWrite assumes you prefer your text, unless told otherwise. Plain Text is text that is not bold, italic, underlined, outlined, or shadowed.

Plane  (Technical) The position of a window, front-to-back, as displayed on the Macintosh desktop.

Point  (Verb) The act of positioning the pointer by rolling the mouse across a flat surface. As a noun, a unit of typographical measurement. Type fonts sizes are given in points, where 72 points equal 1 inch of height, measured from the bottom reach of lower-case letters with descenders (like y), to the upper reach of capital letters (like T).

Pointer  The shape on the display screen that marks the relative location of the mouse. The pointer may be one of several shapes, but can never be moved off the screen. The pointer is the primary symbol used to make selections and otherwise interact with the document or application program.

Pointer shapes  The pointer shape changes, automatically, depending on the nature of the document, or action underway. In text, the pointer becomes an Insertion point, or I-bar; in graphic documents it is a Cross, or Plus sign; in cell documents, it is a Hollow cross, or Hollow plus sign. During periods when a lengthy operation is in progress, the pointer becomes a wristwatch icon, notifying the user to wait with further actions. In all remaining cases, the pointer is arrow-shaped. (Technical) Applications written by third-party vendors may freely redefine pointer shapes to convey unique or specific information to users. And, of course, in many senses the Pointer is the familiar cursor, well-known to millions of computer users.

Press  The act of pushing down and holding the mouse button. Used, most often, to drag the pointer.

Print  As a verb, transferring information (text, data, or graphics) from the Macintosh to a printer, usually the Macintosh Dot-Matrix Printer. The information may, or may not, be visible on the display screen.
**Principal tool** The tool most strongly associated with a given document. Generally, the tool that originally created the document. Users need not worry about which tool is needed by various documents; opening a document automatically selects the required tool—i.e., opening a text window makes MacWrite available for use. [Technical] One of the finest aspects of Macintosh is the intentional blurring, for the user, of distinctions between programs and data. As with all software in Macintosh, the routines necessary to accomplish the magic are readily available, documented (or soon-to-be-documented), and easy to employ by application programs.

**Procedure** [Technical] A relatively small, self-contained, segment of a larger program that performs one or more tasks based on parameters passed to it. Much like a BASIC subroutine. Procedures can be called from the main program, which, among other things, saves typing in all those lines of code each time the procedure is needed. One of the primary concepts in Pascal.

**Processor** [see microprocessor]

**Program** [Technical] What a delight to term this term technical. Programs are collections of instructions that, when executed, act upon data. But you knew that.


**Prompt** [Technical] The concept is done away with in Macintosh, where prompts have become Alert Boxes, Dialog Boxes and such. If you're writing applications for Macintosh, try not to think in terms of prompts; it leads to no good.

**Proportional spacing** A method of displaying, or printing text, that allows varying amounts of white space to characters of varying sizes. Thus, the letter w takes more space than the letter i. Results in a professional typeset look.

**Protect** To shield against harm. Here, usually means write-protecting a disk. To do that, use your fingernail to move the red button toward the edge of the disk. Information can now not be written to your disk, which protects whatever is currently on there from accidental erasure or overwriting (same thing, right?).

**Protected** See above.
Pull-down Menus are. See Pull-down menu.

Pull-down menu A list of selections that appear when the pointer is placed over a menu title, and the mouse button is pressed and held. Sometimes referred to as pop-up menus, though pull-down is more accurate and is Official Apple Terminology.

QuickDraw [Technical] A collection of fundamental routines that allow creation and manipulation of graphic images on screen. Programmers may make calls directly to QuickDraw routines, or call other routines that, in turn, often make calls to QuickDraw.

RAM [Technical] Acronym for Random-Access Memory. The Macintosh is equipped with 128K of RAM, a figure that is not easily expandable, but quite sufficient for most purposes.

Radio button [Technical] A type of control, similar to a Check Dialog Box. As on car radios, only one Radio button may be currently selected. Making a new selection clears the previous choice. A handy tool for programmers.

Random-access memory See RAM.

Read [Technical] [Verb] The process of retrieving data from disk.

Read-only memory See ROM.

Reformat [Technical] To erase all information on disk and lay down track and sector markings in preparation for continued use of the disk. Apple uses the term Re-Initialize rather than reformat.

Release Used here to refer to releasing the mouse button, usually after a drag. Releasing the button causes the action to be performed—when moving a window, the window doesn’t assume its new position until the mouse button is released. If you change your mind, keep your finger on the button and return the pointer to where you began the move, then release the button.

Resource file [Technical] Programs or data, stored on disk, for use by applications programs, but stored separately from the application program.

Return key A key on the Macintosh keyboard. When pressed, Return adds a new line to a document or ends an entry.

Right justification A method of displaying or printing text characterized by a smooth, straight, vertical edge of type along the right side of the text. Right justification leaves a ragged-left edge of text, unlike full justification, which straightens both edges of text.

ROM (Technical) Acronym for Read-Only Memory. A non-volatile type of memory where systems programs and necessary data are stored and made available to the computer. Macintosh has 64K of ROM, which holds the operating system, QuickDraw graphics routines, and enough other goodies to make programmers and hackers coo with delight.

Routine (Technical) A short segment of program code that performs a specific task. Typically heard in reference to assembly-language programs.

Ruler The area under MacWrite's Menu Bar. The ruler is numbered, looks like a yardstick, and has modifiable settings for tabs, line spacing, margins, and text justification. Additional rulers may be inserted into text and cause subsequent text to take on new characteristics; different line spacing, for example.

Run (Technical) To execute a program.

RS-232 (Technical) A standard form of electronic interface which allows memory contents to be transmitted in a serial fashion to peripherals, usually printers and modems. See Serial interface.

RS-422 (Technical) The convention used for the two Macintosh serial ports. A new, fast way to send electrons through wires. The chapter on the Mac back panel has more thrilling information about RS-422 ports.

Sans serif Letters of typefaces without serifs. Okay—serifs are widened ends of individual characters that, in part, give a typeface a distinctive appearance. The New York font, for example, is serified, while the Geneva font is san serif.

Save Storing information onto the Macintosh disk. Often a menu item. Portions of lengthy documents may also be saved and work then continued on the document.
Scroll  Moving the display of a document within a window. Necessary for documents larger than the windows in which they're displayed. Scrolling is accomplished by use of the Scroll bars, which are found on right edge, or bottom, of windows that hold large documents.

Scroll arrow  Found at the ends of Scroll bars, the arrows allow you to move through text a bit at a time. Click once, while the pointer is positioned over either arrow to scroll one line of text, up or down, depending on the arrow clicked. Press for continuous scrolling.

Scroll bar  A vertical or horizontal bar found at the right edge, or bottom, of a window that contains a large, or lengthy document.

Scroll box  Also called a thumb. Used to quickly find and display portions of large documents. See Scrolling.

Scrolling  To move a large document through a window to display the document's contents. Three ways to scroll: position the pointer over a scroll arrow (see definition) and click; this scrolls a line at a time. Pressing, rather than clicking the arrow causes a continuous scroll. Or, click in the gray area within the scroll bar on either side of the white rectangle (the Scroll box, or thumb); this causes the document to scroll a single page forward or backward, depending which gray area was clicked. The last method is thumbling; to thumb, drag the Scroll box to a new position and release the mouse button; the document then scrolls to a position relative to the new position of the Scroll box. Try it!

Search menu  MacWrite menu that contains commands to locate text.

Select  To choose. Information, options, and choices are generally selected by positioning the pointer over the item to select and clicking, or double-clicking the mouse button. Larger areas or actions are selected by Dragging the pointer over, or through, text, numbers, or graphic areas.

Selection  Often a menu item. Sometimes making a selection replaces a previous selection. Working with information means making selections. In all cases, the possible selections, and their consequences, should be understandable. Other definitions (Radio buttons, Alert Boxes, Menu) give more precise information.
**Serial interface** (Technical) One way to connect a computer to an external device. Serial interfaces are used to transmit information in a stream of bits. The other common interface is a parallel interface, which is used to transmit chunks of data. Serial interfaces are often used with printers and modems. See RS-422.

**Serif** Difficult to explain. A typographer's term for the ornate, widened bases and tops seen on some characters of some type fonts. If, for example, the capital letter H is represented with the two vertical bars of equal width, from top to bottom, then the character is Sans Serif. If the bases of the bars (stems) are widened, the character is said to have Serifs. Serifs are also called finishing terminals. The London font (also known as Old English) is extremely serifed.

**Shadow print** A typestyle (see definition). Shadow print is created by widening the black outline on the right and bottom sides of a character. This is the same technique used to give a three-dimensional feeling to certain objects and windows displayed on the Macintosh display.

**Shift key** Two Shift keys are found on the Macintosh keyboard. Either shift key is held down while a second key is struck. The key(s) struck are displayed upper-case (if alphabetical) or the upper character of two-character (number) keys is displayed. For entry of repeated capitals, pressing the Caps Lock key once displays all subsequent letters upper-case, until the Caps Lock key is again pressed, but has no effect on number keys.

**Size box** Two small, overlapping boxes often found in the lower-right corner of windows. When the pointer is positioned within the size dialog box and the mouse button is pressed and held, the window can be Dragged to a larger or smaller size.

**Size (of type)** See Typesize.

**Software** A general term for any program in memory or on a disk. Not very specific. Application software refers to programs that perform a specific task, or tasks. MacWrite is an example of application software. System software are the internal programs within Macintosh.
Sound procedure (Technical) A procedure called from Pascal to produce one of up to four sounds from the Macintosh speaker. The necessary parameter is an integer from 0 to 3.

Source (The) An Information Utility service available to subscribers; allows users with computers and modems to play games, access databases, check flight schedules, post messages, receive and send electronic mail, read newspaper wire services, and a host of other things. Currently, The Source charges a one-time fee of $100 and a per-hour rate based on time-of-day and speed of modem.

Space bar The long, horizontal bar on the Macintosh keyboard. When pressed once, it causes a space to be placed into text at the insertion point. Repeats if held down.

Split bar A symbol found at the top of vertical scroll bars (see definition), or at the left of a horizontal scroll bar. Split bars won't be found on many windows; their presence notes the capability of splitting the window into different panes, each of which will then display a different portion of the document. The separate panes may be scrolled independently of each other. See Splitting a window.

Split window A window that has been divided into two or more panes. See Splitting a window.

Splitting a window The act of dividing a window into two or more panes. For windows to be Splittable a Split bar must be present within the Scroll bar. Split bars are located at the top of vertical scroll bars and, though less seldom seen, at the left of horizontal scroll bars. To split a window, position the pointer on the split bar, then drag the bar to a new position. When the mouse button is released, the window will split into two panes, each with one or more scroll bars. Handy for large, especially numeric, documents.

Stage (Technical) Refers to Alert levels. Each Alert has four possible stages, and a different response may be required for each stage.

Style See Typestyle.

Style menu MacWrite menu that allows text to be displayed in various styles and sizes. The styles (more properly, typestyles) are Italic, Bold, Underline, Outline, and Shadow. The sizes are given in Points (see definition).
 SUSPEND  Halting a process in a manner that allows resumption. Pausing between insertion of single sheets of paper, during printing, is an example of suspending operation of an application. Games often have a suspend feature that allows bathroom or hamburger breaks without giving up a chance at an all-time high score.

SYSTEM FONT  A type font (see definition) used by Macintosh for document titles and Menu Bars. The font is Chicago. (see definition.)

SYSTEM SOFTWARE  (Technical) The software, primarily in ROM (see definition) that coordinates the Macintosh operating system and manages windows, documents, and other aspects of the Macintosh Desktop.

TAB  (Verb) To Tab means to move the Insertion Point to the left in jumps of pre-defined increments. As a noun, a tab is a modifiable place marker within text; setting tabs is especially useful when entering numbers, or text with varying indentations.

TAB KEY  A key on the Macintosh keyboard. Acts much like a typewriter tab key: changes the position in which text will appear to some number of spaces from the left margin.

TEXT  Any group of alphabetic characters. Also, the general name for one of the primary Macintosh document types. The others are Graphic and Cell.

TITLE BAR  All documents have titles. The title appears in the Title Bar, located in a thin bar at the top of a window. Windows are repositioned by dragging while the pointer is positioned within the Title Bar.

TORONTO  A type font supplied with Macintosh. Serifed. Attractive and all-purpose. Looks good in 12 point and larger sizes.

TURN ON  To turn on Macintosh, press the switch on the left rear of the machine, about one-third of the way up. The Owner's Guide tells all you need to know.

THUMB  (Noun) The white rectangle found in Scroll elevators. Also called the Scroll box. See Thumbing.
Thumbing (Verb) An easy way to move through large or lengthy documents. Because windows only show portions of large documents, thumbing was invented to select portions of the text to be displayed in the window. To thumb, place the pointer over the white rectangle (the Scroll box or thumb), which is located in the Scroll bar at the side, or bottom, of the window. Drag the thumb to a new position and release the mouse button. The window now displays the text located in a position relative to the thumb position; i.e., if the thumb is in the middle of the elevator, you're shown the middle of the document. On vertical scroll elevators, the thumb is at its topmost position when the beginning of text is displayed.

Tool Tools are computer programs, or, more precisely, Applications software. Tools allow you to work on, or with, documents. The correct tool (program) is selected by Macintosh when you open a document icon. For example, if the document is text, MacWrite tags along. Expect a great number of software tools to be made available for Macintosh.

Toolbox (Technical) See User interface toolbox.

Top You mean front. The front window is always on, or active, or able to be used. Moving a window to the front automatically opens it for use.

Top-of-form (Technical) A control found on printers that advances paper one page forward. Used with connected, fan-fold sheets of papers. The printer counts and spaces the equivalent of one page, but you must begin at the page separation to end on the page separation. On the ImageWriter, the button marked Form Feed will do the trick.

Type Refers to a collection of printed letters. From the Greek and Latin typus, which means image. There are types of type, sizes of type (given in points), styles of type, etc. All are markings that convey information, often in a striking or pleasing manner, depending on the typeface, style, and wisdom of the typographer.

Typeahead A Macintosh feature that prevents individual keystrokes from being lost. Keys pressed, even if not immediately visible on the display, are stored with the computer and displayed when the machine catches up. The typeahead buffer (see definition) is large enough to eliminate concern about filling it to overflow. It's also possible to clickahead.
**Typeface**  In the old days, printing was done with small blocks of wood (later metal). On top of the blocks was a raised relief of a letter or symbol. Because the relief was higher than other areas of the block's top, it picked up ink and printed a, hopefully, clear image of the relief. The relief was called the face of the block, hence the term typeface. Today, a collection of letters, numbers, and symbols that share a distinctive appearance are called a typeface.

**Typesize**  The size of type, and type fonts, is given and measured in points. A point is about 1/72nd of an inch. Points give an approximate measure of the vertical size of type. On the Macintosh, text may be shown and printed in sizes ranging from 9 points (very small), to 72 points. If the Martians land, expect newspaper headlines of 72 points or more. Because of the nature of showing things on a video screen, text on the Macintosh screen isn't exactly equal to the real typesize in points, but it's close.

**Typestyle**  Characters in any given font may be transformed into a number of styles. As an example, the font may be New York, the size may be 12 point, and the typestyle Italic. The six typestyles available on Macintosh are: plain text, bold, italic, outlined, underlined, and shadowed. The styles may be used within documents with wild abandon, but don't hold Apple Computer responsible for your personal aesthetic blind-spots.

**Undo**  Often a timesaver, sometimes a lifesaver, Undo reverses the consequences of the last change made to a document—usually. Undo only affects the previous action; you can't Undo yourself back to the first of fifteen drafts. [Technical] At least not with tools now available. In theory, programmers are perfectly able to write applications that support undoing back to the Middle Ages.

**User**  A term for you, the owner and user of computers. Does this mean that we're using while we work with computers?

**User Interface Toolbox**  [Technical] The collection of routines and data types that allow programmers to implement features of the standard Macintosh user interface in their application programs. Commonly called Toolbox.

**Venice**  A type font supplied with Macintosh. A cursive font, which looks good in larger sizes (above 12 points).

**Video**  [Technical] See Display.
Widows  A typographical no-no, widows are created when the last line of a paragraph inadvertently appears at the top of the next page, all alone, widowed from the remainder of the paragraph. Widows lend a displeasing appearance to letters. Better to start a new page with at least two lines of text from the paragraph that began on the previous page. Also see Orphan, if you can’t guess the definition.

Window  A fundamental Macintosh metaphor. All documents are displayed in windows, and all windows share many characteristics. By learning how to operate windows—opening, closing, moving, scrolling, etc.—you learn skills to manage a variety of documents, e.g., what owners of other, less sophisticated computers call programs.

Word Wraparound  A text term. A benefit of computerized word processing, word wraparound frees you from pressing the carriage return at the end of lines. When a line becomes too long, the next words are deposited at the beginning of the next line. The ability to word-wrap is a built-in text editing feature of Macintosh.

Write-protect  A method to protect the contents of a disk. Macintosh disks have a small tab that can be repositioned toward the edge of the disk to prevent information from being stored on the disk. The disk still works fine—information can be retrieved and used—but new information can no longer be stored. Until you again move the tab. Good protection for purchased programs, or for creations of your own that represent hours of work (or fun).
Things in General


Computers are just one more system, and simpler than most other systems that swirl about us. Everything in this book applies. Here's what Weinberg says about mastering tools: "If you cannot think of three ways of abusing a tool, you do not understand how to use it." With computers, one person's abuse is often another's "undocumented feature." Weinberg also wrote *The Psychology of Computer Programming*.


$325 and worth every penny. St. Gall was a ninth century Carolingian monastery, and more complex and fascinating a system than anything filled with transistors. Anyone who loves books will love these books. What are *these books* doing *here*? Simple: they're wonderful books! Save your money.

Zen, motorcycles, technology, and values. Good reading for anyone who owns or plans to buy a machine, especially a computer. Includes the best-ever digression on "being stuck." For Monday: read the book, then determine the relationship between Macintosh and "Quality."

**Macintosh**


Not another Macintosh book! Do we really need two! Probably.
Actually, there will be a glut of Macintosh books. Cary Lu's book will undoubtedly remain one of the best, if not the best book about Macintosh. That's okay, I can take it.

Although the book wasn't available for review as these remarks were written, we can assume it's a thorough and interesting look at Macintosh. Cary Lu is the Editor of *High Technology* magazine and admittedly one of the best technical writers now writing about microcomputers. This guy is good.

I said I'd mention his book if he mentioned mine. Do I get a write-up like this? We'll see.

**Type, Graphics, Art, and Publishing**

It's difficult to use the Macintosh without being seduced by graphics and typography. Typography covers everything dealing with the style, arrangement, and appearance of type. Typography as Art began before Gutenberg and continues every time you switch on the Macintosh. The possibilities are exciting. It looks like the publish-it-yourself revolution is dawning. To get in on it, the following books can help.

A worthy first course in the art of typography. Now that you know a bit about fonts, type styles and such, you might want to know a bit more. Invaluable for designing your own fonts, or teaching the Macintosh some new typographical tricks.


44 pages on how it's done.


The Macintosh is the equivalent of newspaper production equipment worth tens of thousand of dollars. Soon we'll see titles like: *How to Produce... with the Macintosh*. Producing newsletters and newspapers is still easier said than done; but with Macintosh, it is easier (and less expensive and more fun) than ever before.


But then, why settle for newspapers? Macintosh-created books are a logical progression. After you've written your book, you may wish to publish it yourself. If so, read Poynter. If it's a computer book, and you don't wish to self-publish, consider Softalk Publishing, they may be interested. (The preceding self-serving advertisement was brought to you without suggestion or coercion from Softalk—but I'm sure they won't mind the plug.)


People with Scott Kim's talent don't come along every generation. The talent? Drawing words that read the same (or differently) backwards, upside down, sideways, uh... pick up a copy and see for yourself. A visual treat.
Computers in General


Maybe *the* single most important book about computers, artificial intelligence, and the nature of the beast; authored by a brilliant pioneer in computer science. You may disagree strongly with Weizenbaum's conclusions (many do), but you'll be enriched by their consideration. Read this prior to picking up any gung-ho book about computers, including this one.


The best opening sentence in any computer book. Something like: Any nitwit can learn to program a computer, and many have. Actually two books: read from the beginning and it's Computer Lib. Start from the back (?) and it's Dream Machines. The book shows its age, but is still a refreshing look into a complex subject.


Good reading and a superb doorstop. This weighty tome lives up to its all-inclusive title. The selections aren't long, but then, after all, there are 550 selections. 1,664 pages, including index. Not a book for the beach, but great for desert islands.


An engrossing history of computers, which concludes in the late seventies. Justifiable attention is given IBM; small computers are mentioned only in passing. Not just a history of computers; a revealing look inside America's corporate giants.

The four volumes are titled, respectively, The Beginner's Book, Basic Concepts, Some Real Microprocessors, and Some Real Support Devices. Together, they offer a comprehensive education in microcomputers. After you turn the last page of volume 3 you'll never again be intimidated by a computer book. Like most Osborne books, these are accurate and well-designed. Still, a good amount of stamina is required to finish the entire series. First ask yourself: do I really need to know this? With Macintosh, you don't; but you might wish to, nevertheless.


A popular, complete introduction to microcomputer hardware.

**Artificial Intelligence**


A landmark book on artificial intelligence and thinking in general (and specific). Winner of the Pulitzer Prize. Not easy going, in parts, but always challenging and frequently delightful. Like all great books, this is really about everything. The author also writes a widely-read column in Scientific American. (Godel's verbose and informative bibliography was the model for the notes you're now reading.)


Hofstadter this time teams up with Daniel Dennett, a philosopher, to consider the nature of consciousness. The result is sure to remain in print for many years, but never make the best-seller list.

The best introduction to artificial intelligence, although good books covering the field now appear regularly. A good companion to Gödel by Hofstadter. The latter half of the book deals with the language LISP. You might want to skip that half, or devour it. If the latter, next read LISP, also by Winston, and here listed under LISP.

**Computer Programming In General**


Very rough going without a thorough background in math. These three volumes are the professional programmer's bibles. If you do have a good math background, you'll find these books absorbing and enjoyable. If you're just starting in computers and programming, don't even consider these books; go to the next reference, quick!


Also not a book for beginners, but worth the effort.

**Computer Languages**

**BASIC**


How to write BASIC programs that other people, and more importantly, that you can understand. The authors courageously claim that "Programmers can and should write programs that work correctly the first time." Good advice given simply and clearly. Also available are *COBOL With Style*, *FORTRAN With Style*, and *Pascal With Style*.

One of the better first books of BASIC. The typographical philosophy here seems to be: Education as Vaudeville.


One of the first computer book best-sellers, this remains a delightful and instructive book.


Essentially an update of *BASIC and the Personal Computer*. Recommended, but don’t waste your money buying both books; get this one. The version of BASIC here is quite unlike MacBASIC, which shouldn’t deter you: this is a very helpful book.

Note: Look for excellent books on MacBASIC from Dr. Scot Kamins, author of the Apple II Reference Manual and Apple Backpack. Kamins is one of the better writers now tilling the binary fields, and also played a role in development of MacBASIC. Kamins’ books will be published by Hayden Book Company.

### FORTH


Books about FORTH are proliferating. This, one of the earliest, is still the best of the lot. When entering the realm of FORTH, be prepared for an avalanche of puns. Some of the puns are so awful that it’s tempting to forthsake the language altogether. *Starting FORTH* gives worthwhile insight into all programming languages, not only FORTH.

FORTH is a threaded interpretive language. If FORTH appeals to you, this book might also. Read *Starting FORTH* first.

LOGO


Advertised as being for children as young as nine as well as adults, this is one of the better introductions to LOGO.

Pascal, C, Modula 2, and Other Structured Languages


A classic. Recommended for those going beyond BASIC, or for serious beginners with an eye on C, Pascal, Modula 2, or PL/1.


Another classic. Page through it next time you visit your local bookstore. Not for beginners, but don't write C without it. Elements and Structured programming are enduring entries in the bibliographies of computer books. Why break tradition?


The definitive book on C. Dennis Ritchie authored the C language while employed at the prestigious Bell Labs, and he knows what he writes about. Surprisingly, for a definitive text, the book is neither pompous nor wordy; like the C language, it's clear and
conce. Many examples are included, and most will run on most C compilers. The book proves that at least some programmers can also write English.


The first book written about Modula 2, authored by the language's author. Precise and to-the-point; a familiarity with Pascal will help the reader greatly. Expect many more books about Modula 2.


Bowles, although not the author of Pascal, is responsible for UCSD's version; to date, the most popular flavor of Pascal.


Like the title says, a good introduction. Dr. Zaks is one of the better and most prolific of computer book authors. SYBEX has one of the better collections of microcomputer books.


A handy reference book for Pascal programmers. Thoroughly covers the nuts and bolts of most major Pascal versions.

**LISP**


A complete introduction to LISP. May, however, be tough sledding for beginners. The book's bibliography is extensive, covering not only LISP but the entire field of artificial intelligence. Winston is one of the best-known artificial intelligence researchers and one of the field's clearest writers.

A short, engaging, non-threatening introduction to LISP. If you think you might want to learn LISP, but aren’t sure, this is a good place to begin.

**PROLOG**


During the preparation of this bibliography, there was only one version of PROLOG available for microcomputers: Logic Programming Associates’ Micro-PROLOG. *A Micro-PROLOG Primer* is a well-written tutorial that accompanied that package. Versions of PROLOG for 68000-based machines (and, hopefully, also for Macintosh) should be available now (or soon); it’s likely that Logic Programming Associates’ PROLOG will be one of the versions. I’ve included the firm’s full address because this book may be difficult to find. Worth a look if you’re interested in this new language.

**Smalltalk**


The first book about Smalltalk. Big, clear, definitive, with a hefty price. Get it from the library, first. Look for future Smalltalk books from Addison-Wesley.
Assembly language, the MC68000 and Technical Topics


The best book for learning 68000 assembly language. Prior knowledge helps; the waters deepen quite fast. Nonetheless, a model for other publishers of what technical tomes can and should be. Extremely thorough and readable, considering the subject. Many well-selected examples.

Wall Chart: M68000 Instruction Set. Order from: Micro Programs, Inc., 251 Jackson Ave., Syossett, NY 11791 (516)/921-1351.

A detailed chart that lists each instruction, assembler syntax, allowable addressing modes, condition codes, and more. Nice to have on the wall, if you're sincere about programming in 68000 assembly language.


About 100 pages of no-holds-barred technical information. Professionals and very advanced programmers will find it a handy, well written reference. All others beware.


Copyright by Motorola, Inc, this is similar to Kane's handbook, listed above, but with Motorola's stamp of authority.


As promised, a book on the ultimate low-level language; anything much lower than this involves quantum physics.

If you're not up to *Foundations of Microprogramming*, try this on for size. You do want to know how computers compute, don't you? The issue also contains an interesting article titled *The Future of the Universe*. Useful reading to keep computers in their proper perspective.

**Unclassified**


This is a spoof dictionary, and a welcome change from the turgid computer glossaries found in most computer books, including this one. Look up Computer Science here and you'll find: "A study akin to numerology and astrology, lacking the precision of the former and the success of the latter." The truly funny aspect of this book is that many of the definitions are truer than their actual counterparts. What better definition of random file than this: "one in which records can get lost in any order."


Fascinating and well-written exploration of how the brain works, and how its workings can be approximated with digital electronics. Assumes you know something about computers and programming, but not necessarily anything about biology.


Things you should have learned in high school, but didn't.


A great title for your coffee table. Also good to actually read; it's about the union of computer graphics and music. 50 color photos. Fun for computer composers and experimenters.

Dr. Zaks strikes again. Good advice, well-given.


If you’ve read the first two, you may think this one will be dated. It isn’t. If you’ve never glanced at one of these creations, you’re in for a treat, and a possible change in attitude. Includes a fair section on computers; by itself, a poor reason to buy the catalog. Life is too short to investigate all, or even a fraction, of what’s noted by Brand and company. A delight for booklovers and anyone with a speck of curiosity.

Brand recently received a huge publisher’s advance of 1.3 million dollars to create a Whole Earth Catalog of Software. Whether Brand can translate his considerable skills in integration and communication to computer software remains to be seen, however. Let’s hope he can.

Magazines

Macintosh Magazines

ST.Mac. P.O. Box 7041, North Hollywood, CA 91605

A Macintosh magazine from Softalk Publishing. For a few sentences on Softalk, see the last entry (beginning with BYTE, the magazines are in alphabetical order). As usual, sending the serial number of your Macintosh to Softalk will result in a free one year subscription to the magazine—one of publishing’s better (and canniest) deals. No strings, just twelve free issues, then a decision to pay up or lapse. Most pay up. Those without Macs can receive 12 monthly issues at the regular price: $24. Advance information indicates a Macintosh magazine that’s both fun and thorough, fast and deep; in all, good news for Macintosh owners.
CW Communications, the organization behind *InfoWorld, PC World, ComputerWorld*, and other magazines, will also have a Macintosh magazine: *Macworld*. Edited by Andrew Fluegelman, *Macworld* promises to be another excellent Macintosh magazine. Macintosh buyers will find a *Macworld* subscription coupon in their Macintosh box. Send in the coupon and you’ll receive two free issues of *Macworld* and a bill for a year’s subscription: 14 issues for the price of 12. Not a bad deal. Subscription rates are $30 for one year (12 issues), with a charter offer of $24 for 12 issues. The charter offer is good through July 1, 1984. You want advice? Subscribe to both magazines. After dropping two grand on a Mac, 50 bucks is a trivial price for two magazine subscriptions, and a good value. Go for it.

In the not-too-distant-future (in computerdom a time frame usually less than two months) expect another Mac magazine; this one from Ziff-Davis Publishing. Ziff publishes *PC* magazine and, it seems, every other title on the newstand. Check it out.

**General Computer Magazines**

*BYTE*. 70 Main St. Peterborough, NH 03458

The standard by which other microcomputing publications are judged. Written for the serious hobbyist/user. Not specific to any one computer, or area of computing. You may not understand some, or even most, of the articles, but reading the ads alone is educational. Never throw out a copy of *BYTE*. Someday you may understand some of the more technical articles and want to read them. The largest in circulation of all computer magazines, with over 200,000 subscribers.

*Creative Computing*. Box 789-M, Morristown, NJ 07960

A hodge-podge of articles, tutorials, columns, reviews, programs, and some unclassifiable pieces, Creative is the “liberal arts” of computer magazines. Usually features a number of honest, first-person product reviews written by hobbyists. Recently, has been slanted toward “arcade-cadet” types.
Dr. Dobb's *Journal of Computer Calisthenics and Orthodontia (Running Light Without Overbyte)*. People's Computer Company, P.O. Box E, 1263 El Camino Real, Menlo Park, CA 94025

A magazine as interesting as its title. Community-oriented (the community consists of those who like computers—a lot). The subject matter covers much ground. Expect assembly language listings, opinions, some beginner material, and much liveliness.

*InfoWorld*. 530 Lytton Ave, Palo Alto, CA 94301

A weekly magazine devoted to the exploding small computer industry, from a user's point of view. Newsy, easy to read, informative, often opinionated, *InfoWorld*, of late, has become the Time of computer magazines. Also featured, on a regular basis, is an column written by this author.

*Popular Computing*. 70 Main St., Peterborough, NH 03458

Less technical than *BYTE*, and less creative than *Creative Computing*. Articles, features, and reviews in a typical mix. Good for newcomers. Excellent production and wide distribution; this is one you may find at the supermarket, unlike other periodicals listed here.

*Softalk (Apple)*. 7250 Laurel Canyon Blvd., North Hollywood, CA 91605

More “down-home” and good-humored than *BYTE*, *Softalk* also includes many good technical articles, along with a wealth of people-oriented features. *Softalk* focuses on users who want to understand and use their machine, rather than on gadget-dilettantes with bottomless bank accounts. Also, small computerdom's longest running Hit Parade of best-selling software programs. As always, Apple IIe and Apple III owners may send in their serial numbers to *Softalk* for a free trial subscription. The company is also rumored to publish books about computers, computer languages, and such.
Specialized Computer Publications

FORTH Dimensions. FORTH Interest Group, P.O. Box 1105, San Carlos, CA 94080

Bigger than a newsletter, smaller than a magazine, this is where FORTH-folks take their stand. A mass of black-and-white pages stapled together—don’t expect the National Geographic. Because FORTH is a language in transition, lively debates are common. "Dimensions" is indispensable to FORTH devotees, incomprehensible to everyone else. Just as it should be.

Robotics Age. P.O. Box 801, La Canada, CA, 91011

Founded and edited by Carl Helmer, famous former editor of BYTE. Helmers believes that robots are "what’s next." He may be right. Most of the articles still concern themselves with industrial applications. That may change soon, if the Home Robot industry swallows (or emerges from) the Personal Computer industry. And it might, but probably not this year. Robotics Age is a relatively slim magazine, due to the lack of robot-related manufacturers and, hence, lack of advertising. This, too, may change soon.
Everyone, it seems, is busy writing software for Macintosh, or busy developing hardware additions for Macintosh.

The question becomes: who to include within this section? It's a difficult decision; the products are in various stages of completion, and some products may appear radically different than now supposed, and some products may not appear at all—which won't help the integrity of this book at all.

After a few minutes of rather deep thought, it was decided to list only hardware and software products slated for release in the first two quarters of 1984: from January to June.

Having arrived at this simple rule, it was decided to break the rule when dealing with firms known for straight dealing, Penguin Software, for example.

Explanations complete: let's look at some hardware and software that will soon be available for Macintosh.
Software

Hayden Software
600 Suffolk St.
Lowell, MA 01853

Hayden markets an extensive line of game and educational software. Their bestseller is Sargon, a chess program that's hard to beat. Hayden will make Sargon available for Macintosh sometime in the 2nd quarter, with educational programs to follow, followed by their full lines of games.

Living Video Text
100 Elwell Court
Palo Alto, CA 94303

This company markets a program called Think Tank. It's hard to describe. Think Tank bills itself as an "idea processor." Idea processing, it seems, has a lot in common with outlining, which we all remember from high school. Think Tank offers a multitude of ways to outl—oops: idea process.

Doesn't sound like a hot idea, does it? But users rave about Think Tank, and it receives consistently good reviews, so there must be something to it. Get a demo; see what you think.

Software Publishing Corporation
1901 Landing Drive
Mountain View, CA 94301

Software Publishing markets programs with titles such as PFS:Write, PFS:File, PFS:Report, and the like. For Macintosh, expect PFS:File and PFS:Report early in the 2nd quarter, with PFS:Chart and PFS:Write to follow.

PFS programs are typically easy to use and typically big sellers. they're also priced right: around $100 for each program. Try 'em, you'll probably like them.

Penguin Software
830 4th Avenue
Geneva, IL 60134

Well-known in Apple circles, Penguin markets adventure and arcade games, and some of the best graphics programs available. For Macintosh, the adventure programs Transylvania and Quest
FURTHER SOFTWARE AND HARDWARE

will be released early in '84. Then will come The Graphics Magician, an intriguing title, especially in light of the magical graphics abilities already possible with MacPaint.

Still, expect Graphics Magician to offer even more graphic wonders to Macintosh.

Penguin expects to have their complete line of software available for Macintosh by the end of '84. It's one of the best collections of software available for the Apple II and IIe; it should be even better for Macintosh.

Lotus Development Corporation
161 First Street
Cambridge, MA 02142

Lotus markets 1-2-3: a spreadsheet, charting, database, word processing program for the IBM PC. All this in one program.

1-2-3 is the monster program for the PC: its sales are incredible, far and away greater than any other program sold for the PC.

Why is it so popular? Well, it offers a spreadsheet of enormous size, speed, and functionality. It produces great graphs effortlessly. It comes with top drawer documentation, including the best tutorial disk on the market.

Lotus is an all-around class operation.

Will they repeat their success on the Macintosh?

Infocom, Inc.
P.O. Box 885
Garden City, NY 11530

Infocom markets text adventure games. No graphics, just marvelously descriptive text, incredibly involved scenarios, and a one-of-a-kind "game language" that seems vastly more intelligent than it really is.

All Infocom games are wonderful and truly addictive. Their price is generally under $40 each. By the time you win one of these games (if you ever do), you'll have received your money's worth and more.

Also, Infocom games come in some of the industry's most delightfully engaging packaging.
Teecmar is a big name in hardware add-on boards and peripherals, primarily in the IBM PC marketplace, where Tecmar offers a bewildering array of products.

Tecmar will also offer a bewildering array of products for Macintosh. Slated for release in early ’84 are:

- A five megabyte, removable cartridge, hard disk drive. Those in the know call these “toaster” hard disks. Each cartridge holds up to five megabytes of programs, data, documents, tools, or what-have-you. You probably won’t have enough software to fill even one of these cartridges, at least not right away. Some-day, though, you probably will. You’ll then know why toaster hard disks are such a great idea.

- A print buffer. Print buffers are filled with extra RAM memory, and are used to free up use of the computer during printing. Instead of waiting for a slow printer (and all printers are slow compared to a computer’s internal memory), text is “dumped” into the print buffer, which then doles out the characters to the printer, while you go on to other work. Handy if you do a lot of lengthy document printing.

- An IEEE48 interface card. Much loved in the scientific community, this card will allow you to attach laboratory instruments to your Macintosh. If you don’t know what this is, you probably don’t need one.

- An expansion chassis. Expansion! More memory! More memory! And a place to plug in your IEEE48 cards, and lot of other cards that are soon to come from Tecmar (and other places). The Tecmar expansion chassis is certain to be a big seller, unless some other manufacturer’s expansion chassis sells more.

Also, some sort of spectacular telephone/modem combination that, to hear Tecmar tell it, does everything conceivable to do with telephones and modems. And more. We’ll see.
Davong Systems
217 Humbolt Court
Sunnyvale, CA 94089

Davong made their name in the add-on hard disk market for the IBM PC. They hope to repeat that success with Macintosh.

Their first offerings will be (surprise!) hard disks, in sizes from five to thirty-two megabytes. Expect to spend from $1,500 to $4,000, depending on capacity.

By the 2nd quarter of '84, Davong hopes to have a tape back-up unit for their hard disks available. Following that, a full networking system is promised, possibly the first entry in the sure-to-be-crowded network market for Macintosh.
Text Samples

This is a sample of the New York font

This is a sample of 12 point Geneva

This is Monaco in 12 points

This is Chicago in 12 points

This is Venice 14 points, plain text.

London in 18 Points. Underlined.

Athens in 18 points, plain text.

Toronto in 14 points, bold.
New York font

9 point bold, *Italic, Underline, Outline, Shadow, the works.*

12 point bold, *Italic, Underline, Outline, Shadow, the works.*

14 point bold, *Italic, Underline, Outline, Shadow, the works.*

18 point bold, *Italic, Underline, Outline, Shadow, the works.*

24 point bold, *Italic, Underline, Outline, Shadow, the works.*
Geneva font

9 point bold, *italic*, *underline*, *outline*, *shadow*, *the works*.

12 point bold, *italic*, *underline*, *outline*, *shadow*, *the works*.

14 point bold, *italic*, *underline*, *outline*, *shadow*, *the works*.

18 point bold, *italic*, *underline*, *outline*, *shadow*, *the works*.

24 point bold, *italic*, *underline*, *outline*, *shadow*, *the works*.
Monaco font

9 point bold, *italic*, Underline, Outline, Shadow, the works.

12 point bold, *italic*, Underline, Outline, Shadow, the works.

14 point bold, *italic*, Underline, Outline, Shadow, the works.

18 point bold, *italic*, Underline, Outline, Shadow, the works.

24 point bold, *italic*, Underline, Outline, Shadow, the works.
Chicago font

9 point bold, *italic* Underline, Outline, Shadow, the works.

12 point bold, *italic* Underline, Outline, Shadow, the works.

14 point bold, *italic* Underline, Outline, Shadow, the works.

18 point bold, *italic* Underline, Outline, Shadow, the works.

24 point bold, *italic* Underline, Outline, Shadow, the works.
Venice font

9 point bold, *Italic, Underline, Outline, Shadow*, the works.

12 point bold, *Italic, Underline, Outline, Shadow*, the works.

14 point bold, *Italic, Underline, Outline, Shadow*, the works.

18 point bold, *Italic, Underline, Outline, Shadow*, the works.

24 point bold, *Italic, Underline, Outline, Shadow, the works*.
London font

9 point bold, *Italic* Underline. Outline Shadow the works.

12 point bold, *Italic* Underline. Outline. Shadow, the works.

14 point bold, *Italic* Underline. Outline Shadow the works.

18 point bold, *Italic* Underline. Outline. Shadow, the works.

24 point bold, *Italic* Underline. Outline, Shadow. the works.
Athens font

9 point bold, *Italic* Underline Outline Shadow *the works*

12 point bold, *Italic* Underline Outline Shadow *the works*

14 point bold, *Italic* Underline Outline Shadow *the works*

16 point bold, *Italic* Underline Outline Shadow *the works*

24 point bold, *Italic* Underline Outline Shadow *the works*
Toronto font

9 point bold, *Italic*, Underline, Outline, Shadow, the works

12 point bold, *Italic*, Underline, Outline, Shadow, the works

14 point bold, *Italic*, Underline, Outline, Shadow, the works

18 point bold, *Italic*, Underline, Outline, Shadow, the works

24 point bold, *Italic*, Underline, Outline, Shadow, the works
San Francisco font

- 9 point bold, *ital*ic, underline, outline, shadow, *the works*
- 18 point bold, *ital*ic, underline, outline, shadow, *the works*
- 14 point bold, *ital*ic, underline, outline, shadow, *the works*
- 18 point bold, *ital*ic, underline, outline, shadow, *the works*
- 24 point bold, *ital*ic, underline, outline, shadow, *the works*
Macintosh Hardware Specifications

Microprocessor:

Motorola MC68000:
- 32-bit internal architecture
- 16-bit data bus
- 17 32-bit data and address registers
- Directly addresses 16 megabytes of memory
- 8 megahertz frequency

Disk drive:
- Sony/Apple single-sided, soft-sectored
- 437.5K unformatted
- Variable speed disk controller: 300–600rpm
- Auto-insert, software controlled disk eject
- Protective spring-loaded shutter
- Hard plastic disk casement

Main unit:
- ROM: 64K
- RAM: 128K
- Display: 9" black-and-white, 512 X 342 pixels, bit-mapped. No text mode; all is graphics.
Sound: 4-voice sound generator, internal 2" speaker.
Other: internal, battery-operated clock;
switching power supply.

Interfaces: Eight-port serial keyboard bus; quadrature mouse interface; two RS-422 high-speed serial ports; audio output jack; connector for security kit.

Keyboard:
Software mapped; 58 keys, 2-key rollover; foreign, alternative characters available by two-key combinations.

Options:
ImageWriter printer, modem, numeric keypad, external disk, carrying case, security kit.
### Standard ASCII Control Codes

<table>
<thead>
<tr>
<th>Hexcode</th>
<th>Abbreviation</th>
<th>Meaning</th>
<th>Macintosh key</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>SOH</td>
<td>Start of Heading</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>STX</td>
<td>Start of Text</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>ETX</td>
<td>End of Text</td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>EOT</td>
<td>End of Transmission</td>
<td>Enter (KB, KP)</td>
</tr>
<tr>
<td>05</td>
<td>ENQ</td>
<td>Enquiry</td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>ACK</td>
<td>Acknowledge</td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>BEL</td>
<td>Bell</td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>BS</td>
<td>BackspaceBackspace (KB)</td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>HT</td>
<td>Horizontal Tabulation</td>
<td>Tab (KB)</td>
</tr>
<tr>
<td>0A</td>
<td>LF</td>
<td>Line Feed</td>
<td></td>
</tr>
<tr>
<td>0B</td>
<td>VT</td>
<td>Vertical Tabulation</td>
<td></td>
</tr>
<tr>
<td>0C</td>
<td>FF</td>
<td>Form Feed</td>
<td></td>
</tr>
<tr>
<td>0D</td>
<td>CR</td>
<td>Carriage Return</td>
<td>Return (KB)</td>
</tr>
<tr>
<td>0E</td>
<td>SO</td>
<td>Shift Out</td>
<td></td>
</tr>
<tr>
<td>0F</td>
<td>SI</td>
<td>Shift In</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>DLE</td>
<td>Data Line Escape</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>DC1</td>
<td>Device Control 1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>DC2</td>
<td>Device Control 2</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>DC3</td>
<td>Device Control 3</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>DC4</td>
<td>Device Control 4</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>NAK</td>
<td>Negative Acknowledge</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>SYN</td>
<td>Synchronous Idle</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>ETB</td>
<td>End of Transmission Block</td>
<td></td>
</tr>
<tr>
<td>Hexcode</td>
<td>Abbreviation</td>
<td>Meaning</td>
<td>Macintosh key</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>18</td>
<td>CAN</td>
<td>Cancel</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>EM</td>
<td>End of Medium</td>
<td></td>
</tr>
<tr>
<td>1A</td>
<td>SUB</td>
<td>Substitute</td>
<td></td>
</tr>
<tr>
<td>1B</td>
<td>ESC</td>
<td>Escape</td>
<td>Clear [KP]</td>
</tr>
<tr>
<td>1C</td>
<td>FS</td>
<td>File Separator</td>
<td>Left arrow [KP]</td>
</tr>
<tr>
<td>1D</td>
<td>GS</td>
<td>Group Separator</td>
<td>Right arrow [KP]</td>
</tr>
<tr>
<td>1E</td>
<td>RS</td>
<td>Record Separator</td>
<td>Up arrow [KP]</td>
</tr>
<tr>
<td>1F</td>
<td>US</td>
<td>Unit Separator</td>
<td>Down arrow [KP]</td>
</tr>
<tr>
<td>20</td>
<td>SP</td>
<td>Space (Black)</td>
<td>Space bar [KB]</td>
</tr>
<tr>
<td>7F</td>
<td>DEL</td>
<td>Delete</td>
<td></td>
</tr>
</tbody>
</table>

This chart lists the first 34 characters of the ASCII character set. Many of the traditional control characters date back to the days when teletype printers were the typical computer display device; many of these characters are not implemented by the Macintosh, or used for different purposes. The rightmost column, Macintosh Key, lists the characters that can be generated from the Macintosh keyboard or from the optional numeric keypad. KP represents keypad. A number of other control codes have unique Macintosh uses, generally for characters [such as black and open apple symbols] that are only displayed on-screen.
Appendix D

Standard Key
Character Formats

Key Caps

Unshifted Key Characters
Key Caps

Shift Key Characters

~ ! @ # $ % ^ & * ( ) _ +
Q W E R T Y U I O P { } |
A S D F G H J K L : "
Z X C V B N M < > ?

Key Caps

Caps Lock Key Characters

' 1 2 3 4 5 6 7 8 9 0 - =
Q W E R T Y U I O P [ ] \|
A S D F G H J K L ; 
Z X C V B N M , . /
Key Caps

Option Key Characters

Key Caps

Shift-Option Key Characters
Macintosh BASIC keywords

Note: because Macintosh BASIC was incomplete when this appendix was prepared, this appendix may, itself, be incomplete and subject to change.

Numeric Functions

ABS  ASC  DEF FN  FRE  INT  PI  RANDOMIZE  SGN  SQR  TYP  TRUNC  WHEN—UPDATE #|numexpr|;|, label

String Functions

CHR$  LEFT$  LEN  MID$  PRINT  PRINT USING  RIGHT$  TAB
File Operations

OPEN #
CLOSE #
IN FROM #
OUT TO #
filename
STORAGE
PRINTER
SERIAL
RECORD PICTURE
WINDOW

Control Constructions

CALL name
DO—{EXIT} LOOP
FOR—TO—{STEP} NEXT
GOSUB label:—RETURN
GOTO label:
IF—THEN—{ELSE}
IF—THEN—{ELSE} ENDIF
POP

System Functions

MOUSEB
MOUSEX
SHOW DIAL
SHOW BUTTON
SHOW MENU
REMOVE DIAL
REMOVE BUTTON
REMOVE MENU

Other

WHEN—ENDWHEN
BTNWAIT
DATES
DIM
ERASE
ERR
FORMAT$
INPUT
KBD
LET (optional)
PRINT
READ—DATA—RESTORE
! [comment abbreviation]
REM
TIMES$

Set-Options

SET/ASK PENMODE
SET/ASK PENSIZE
SET/ASK PENPOS
SET/ASK VPOS
SET/ASK BOUNDS
SET/ASK PORT
SET/ASK VIEWPORT
SET/ASK WINDOW
SET/ASK PATTERN
SET/ASK DIAL
SET/ASK BUTTON
SET/ASK DIALSTATUS
SET/ASK BUTTONSTATUS
SET/ASK MENUSTATUS

Graphics

DRAW PICTURE
END PICTURE
ERASE
FRAME
INVERT
PAINT
PLOT
RECORD PICTURE
Macintosh BASIC Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Symbol</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double precision real</td>
<td>none</td>
<td>15 digits</td>
</tr>
<tr>
<td>Single precision real</td>
<td></td>
<td>(vertical bar)</td>
</tr>
<tr>
<td>Extended Precision real</td>
<td>\</td>
<td>18+ digits</td>
</tr>
<tr>
<td>Short integer</td>
<td>%</td>
<td>4+ digits</td>
</tr>
<tr>
<td>Long integer</td>
<td>#</td>
<td>18+ digits</td>
</tr>
<tr>
<td>Boolean</td>
<td>~</td>
<td>1 (1 or 0)</td>
</tr>
<tr>
<td>Strings</td>
<td>$</td>
<td>any length</td>
</tr>
</tbody>
</table>

Macintosh BASIC Operators

Arithmetic

<table>
<thead>
<tr>
<th>Operator</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition</td>
<td>+</td>
</tr>
<tr>
<td>Subraction</td>
<td>-</td>
</tr>
<tr>
<td>Multiplication</td>
<td>*</td>
</tr>
<tr>
<td>Real Division</td>
<td>/</td>
</tr>
<tr>
<td>Integer Division</td>
<td>DIV</td>
</tr>
</tbody>
</table>

Relational and Logical

(note: operators return 1 if true, 0 if false)

<table>
<thead>
<tr>
<th>Operator</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal to</td>
<td>=</td>
</tr>
<tr>
<td>Greater than</td>
<td>&gt;</td>
</tr>
<tr>
<td>Less than</td>
<td>&lt;</td>
</tr>
</tbody>
</table>
| Greater than or equal to | >= or ==>
| Less than or equal to | <= or =< |
| Greater or less than | <> or >>< |
| AND                 | True (1) if both expressions are true |
| OR                  | True (1) if either or both expressions are true |
| NOT                 | True (1) if original expression is false |
Macintosh BASIC Order of Precedence

Listed from highest precedence to lowest precedence

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( )</td>
<td>innermost parentheses evaluated first</td>
</tr>
<tr>
<td>+ -</td>
<td>unary operators</td>
</tr>
<tr>
<td>* /</td>
<td>multiplication and division</td>
</tr>
<tr>
<td>= &lt;&gt;</td>
<td>addition and subtraction</td>
</tr>
<tr>
<td>AND</td>
<td>relational operators</td>
</tr>
<tr>
<td>OR</td>
<td>logical AND</td>
</tr>
<tr>
<td></td>
<td>logical OR</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>ABCD</td>
<td>Add decimal with extend</td>
</tr>
<tr>
<td>ADD</td>
<td>Add</td>
</tr>
<tr>
<td>AND</td>
<td>Logical AND</td>
</tr>
<tr>
<td>ASL</td>
<td>Arithmetic shift left</td>
</tr>
<tr>
<td>ASR</td>
<td>Arithmetic shift right</td>
</tr>
<tr>
<td>Bcc</td>
<td>Branch conditionally</td>
</tr>
<tr>
<td>BCHG</td>
<td>Bit test and change</td>
</tr>
<tr>
<td>BCLR</td>
<td>Bit test and clear</td>
</tr>
<tr>
<td>BSET</td>
<td>Bit test and set</td>
</tr>
<tr>
<td>BRA</td>
<td>Branch always</td>
</tr>
<tr>
<td>BSR</td>
<td>Branch to subroutine</td>
</tr>
<tr>
<td>BTST</td>
<td>Bit test</td>
</tr>
<tr>
<td>CHK</td>
<td>Check register against bounds</td>
</tr>
<tr>
<td>CLR</td>
<td>Clear operand</td>
</tr>
<tr>
<td>CMP</td>
<td>Compare</td>
</tr>
<tr>
<td>DBcc</td>
<td>Test condition, decrement and branch</td>
</tr>
<tr>
<td>DIVS</td>
<td>Divide [signed]</td>
</tr>
<tr>
<td>DIVU</td>
<td>Divide [unsigned]</td>
</tr>
<tr>
<td>EOR</td>
<td>Logical Exclusive OR</td>
</tr>
<tr>
<td>EXG</td>
<td>Exchange registers</td>
</tr>
<tr>
<td>EXT</td>
<td>Sign extend</td>
</tr>
<tr>
<td>JMP</td>
<td>Jump</td>
</tr>
<tr>
<td>JSR</td>
<td>Jump to subroutine</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>LEA</td>
<td>Load effective address</td>
</tr>
<tr>
<td>LINK</td>
<td>Link stack</td>
</tr>
<tr>
<td>LSL</td>
<td>Logical shift left</td>
</tr>
<tr>
<td>LSR</td>
<td>Logical shift right</td>
</tr>
<tr>
<td>MOVE</td>
<td>Move</td>
</tr>
<tr>
<td>MOVEM</td>
<td>Move multiple registers</td>
</tr>
<tr>
<td>MOVEP</td>
<td>Move peripheral data</td>
</tr>
<tr>
<td>MULS</td>
<td>Multiply (signed)</td>
</tr>
<tr>
<td>MULU</td>
<td>Multiply (unsigned)</td>
</tr>
<tr>
<td>NBCD</td>
<td>Negate decimal with extend</td>
</tr>
<tr>
<td>NEG</td>
<td>Negate</td>
</tr>
<tr>
<td>NOP</td>
<td>No operation</td>
</tr>
<tr>
<td>NOT</td>
<td>Logical one's complement</td>
</tr>
<tr>
<td>OR</td>
<td>Logical OR</td>
</tr>
<tr>
<td>PEA</td>
<td>Push effective address</td>
</tr>
<tr>
<td>RESET</td>
<td>Reset external devices</td>
</tr>
<tr>
<td>ROL</td>
<td>Rotate left (no extend)</td>
</tr>
<tr>
<td>ROR</td>
<td>Rotate right (no extend)</td>
</tr>
<tr>
<td>ROXL</td>
<td>Rotate left (with extend)</td>
</tr>
<tr>
<td>ROXR</td>
<td>Rotate right (with extend)</td>
</tr>
<tr>
<td>RTE</td>
<td>Return from exception</td>
</tr>
<tr>
<td>RTR</td>
<td>Return and restore</td>
</tr>
<tr>
<td>RTS</td>
<td>Return from subroutine</td>
</tr>
<tr>
<td>SBCD</td>
<td>Subtract decimal with extend</td>
</tr>
<tr>
<td>Scc</td>
<td>Set conditional</td>
</tr>
<tr>
<td>STOP</td>
<td>Stop</td>
</tr>
<tr>
<td>SUB</td>
<td>Subtract</td>
</tr>
<tr>
<td>SWAP</td>
<td>Swap halves of data register</td>
</tr>
<tr>
<td>TAS</td>
<td>Test and set operand</td>
</tr>
<tr>
<td>TRAP</td>
<td>Trap</td>
</tr>
<tr>
<td>TRAPV</td>
<td>Trap on overflow</td>
</tr>
<tr>
<td>TST</td>
<td>Test</td>
</tr>
<tr>
<td>UNLK</td>
<td>Unlink</td>
</tr>
</tbody>
</table>
accessories, 239
active selection, 239
active windows, 63, 239
Ada, 166
advanced editing, 89
Alarm Clock, 55, 56, 196
  battery, 196
  Control Panel, 55
  internal battery, 55
  set, 56
alerts, 74
algorithms, 157, 158
Alert Boxes, 20, 63, 65, 74, 75, 240
  alerts, 74
  cancel, 75
  errors, 74

hierarchy of on-topness, 65
warnings, 74
analog board, 202, 206, 207
  internal speaker, 206
  power filters, 207
  power supply, 207
  video display circuits, 207
APL, 167
Apple menu, 53
Apple-Sony disk, 103, 106, 202
Arrange, 47
arrow, 29, 240
ASCII, 240, see also Appendix C
Assembler, 164
Assembly language, 241
ASKPENPOS, 177
Athens, 241, 302
Atkinson, Bill, 28, 134
attitude, 227
audio out jack, 197
auto-repeat, 39

B
back panel, see hardware
Backspace key, 34, 39, 55, 82, 241
  destructive backspace, 39
  in Key Caps, 55
  select/Backspace combination, 39
backslash, 174
BASIC, 241, see also MacBASIC
baud, 242
beep, 73, 75
  Alert Box, 75
  Modal Dialog Box, 73
bit, 95, 242
black Apple, 53
Borders, 137
  patterns, 137
breaking the computer, 187
Brush Mirrors, 135, 138
  brush shapes, 135
bulletin board, 242
Business, 116, 120
  VisiCalc, 116, 120
button, 243
by icon, 47
byte, 95, 243

C
C, 166, 243
C (clear), 55
Calculator, 55, 243
Call, 243
camera-ready copy, 128, 129
cancel, 75, 243
  in Alert Box, 75
Cancel Box, 72
Caps Lock, 36, 243
Cell documents, 21, 23, 244
  electronic spreadsheet, 21
  grid-like structure, 23
Character keys, 36
  Caps Lock, 36
  Command, 36
  Option key, 36
Chart, 143, 148, 149
  compatibility, 148
  graphing, 149
  manual, 149
Check Box, 244
Chicago, 244, 299
Clean up, 47, 244
Clicking, 29, 31, 32, 68, 83, 138, 245
  a pattern, 32
  the Eraser, 32
  the Hand, 138
  the Insertion Bar, 83
  the mouse, 29, 31, 32
  to scroll, 68
  to select an icon, 32
Clipboard, 86, 89, 90, 100, 101, 245
  clippings, 89
  Heap, 100
  Hide Clipboard, 90
  Scrap, 101
  Show Clipboard, 90
Clippings, 245, see also Clipboard
Clock, see Alarm Clock
Close All, 93
  discontinuous icons, 93
Close Box, 51, 55, 245
COBOL, 167
colon, 107, 108
  filename, 107, 108
Command key, 36, 37, 245
  and a single letter, 37
  command mode, see Dialog Box
communications, 117, 118
  CompuServe, 118, 245
  modem, 118
  The Source, 118
compilers, 158, 159
CompuServe, 118
computer games, 119
computerphobia, 187
continuous scrolling, 246
Control Panel, 55, 58, 246
Desktop Pattern Editor, 56
set clock, 56
set key speed and repeat, 56, 57
set menu item twinkle, 56
set mouse in motion, 56, 57
set speed and repeat, 56, 57
set speed of Insertion Bar blinking, 56
volume of the internal speaker, 56
Copy, 89, 90C
copy of Empty Folder, 52
CoreEdit, 81
cursor, 81
Cut, 89, 90, 246

D

data, 97, 100, 246
management, 114
structures, 98
transfer, 100
decimal tab, 247
Defender, 202
design, 202, 204
analog board, 202
Apple/Sony disk drive, 202
Burrell Smith, 202
digital motherboard, 202, 204
sound generation, 204
video tube, 202
Desk Accessories, 53-56, 58, 63, 247
Alarm Clock, 55
Calculator, 53
Control Panel, 55, 56
Key Caps, 55
more accessories, 58
Note Pad, 55
Puzzle, 54
Scrapbook, 56
desktop, 19, 20, 65, 247
display, 19, 20
hierarchy of on-topness, 65
object, 20
Desktop Pattern Editor, 56, 57
Fat Bits, 57
destructive backspace, 139
dial, 247
Dialog Box, 20, 63, 65, 66, 72, 73, 248
cancel box, 72
comments, information, instructions, 72
controls, 72
graphics, 72
hierarchy of on-topness, 65
Modal Dialog Box 73
Modeless Dialog Box, 73
Okay, 66, 72
rectangles for text entry, 72
directory of files, 47
disk, 4, 103-107, 109, 248
capacity, 104
eject, 107
hard, 109
initializing, 103, 105
insertion, 105
re-initializing, 104
sectors, 104
tracks, 103
variable-speed disk controller, 106
write-protecting, 106
disk drive, 103, 108, 197
one and two, 108
socket, 197
display, 248
document files, 25
Document Window, 62
close, 62
Quit from File menu, 62
Title Bar, 62
documents, 21-24
cell, 21, 23
graphic, 21, 22
icons, 24
text, 21, 22
DO-LOOP, 173
double clicking, 29, 31, 32, 45, 47, 61, 84, 249
to open a window, 61
to select an entire word, 84
to select and perform an action, 32
dragging, 29, 31, 33-35, 68, 84, 87, 92, 249
in MacPaint, 34
in MacWrite, 34
in spreadsheets, 34
Point-Press-Drag-Release, 33
scanning the menus, 35
Selection Triangle, 33
through text, 92
to edit, 87
to scroll, 68
to select portions of text, 84
drawing tools, see tools, for drawing
Duplicate, 51, 64
on the File menu, 64
Dynabook, 11

E

editing, 85-87
Backspace key, 85
Clipboard, 86
Copy, 86
Cut, 85
dragging, 87
mouse, 87
restore by Undo, 85
typing, 85
Edit menu, 90, 249
education, 118
electrical ground, 208
electronic interface, 41
general-purpose clocked bi-directional serial port, 41
electronic spreadsheet, 21, 68, 144
grid, 21
horizontal scroll bars, 68
Elevators, 67, 68, 249
gray bar, 67
horizontal scroll bars, 68
scroll arrows, 67
Scroll Box or thumb, 68
a line at a time, 68
by page, 68
continuously, 68
display a new portion, 68
Empty folder, 44, 49, 50, 63
enter, 38
Enter key, 249
ERASE, 175, 176
eraser, 135
errors, 74
Event Manager, 212, 213
Event queue, 39
events, 29, 31
clicking, 29, 31
double-clicking, 29, 31
dragging, 29, 31
press, 29
shift-clicking, 29, 31
EXIT, 173
external speaker, 199
extracurricular activities, 230
books, 230
dealers, 230
magazines, 230

F

Fat Bits, 57, 138, 250
Hand, 138
in Desktop Pattern Editor, 57
field-motion keys, 250
File, 143
File menu, 32, 2, 34, 45
Get Info, 45
Open, 32, 34
Quit, 45
filenames, 107
Files, 25, 26
document, 25
resource, 26
tool, 26
Find, 250
Finder, 44, 56, 50
desktop, 50
icons, 50
Menu Bar, 50
more icons, 50
operating system, 43, 44, 212
asynchronous I/O, 212
INDEX

hardware, 196, 197, 205, 208, see also Appendix B
Apple Reference Guide, 208
audio out jack, 197
back panel, 196
battery, 196
design, see design
disk drive socket, 197
modem socket, 197
mouse socket, 197
on/off switch, 197
PALs, 205
printer socket, 197
security chain, 197
Heap, 100
hexadecimal, 96
Hid e Clipboard, see Clipboard
hierarchy of on-topness, 65
Alert Boxes, 65
desktop, 65
Dialog Boxes, 65
pointer, 65
pull-down menus, 65
windows, 65
hobbies, 232
hollow cross, 30, 252
hollow or filled,
freehand lines, 137
irregular polygons, 137
ovals, 137
rectangles, 137
rounded-corner rectangles, 137
home insurance, see insurance
home-publishing, 130
hoopless design, 2

icon, 24, 44, 46, 49, 69, 252
actions, 44
documents, 44
entire disks, 44
files, 44
invert, 46
split bar, 69
tools, 44
Trash, 49

I

G

games, see computer games
Geneva, 251, 297
Get Info, 45, 82, 93
discont iguous icons, 93
to type in comment box, 82
GOSUB, 172
GOTO, 172
graphic commands, see Mac-
BASIC, graphics
graphic documents, 21, 22, 251
free-form documents, 22
graphics, see MacPaint
graphics routines, see MacPaint;
see also QuickDraw
Grid aid, 138
grow region, 251

H

hacking, 161
hand, 136, 182
Pascal, 182
hard disk, 252
IF-ENDIF, 174
IF-THEN--ELSE-ENDIF, 173
ImageWriter, 129, 198
In Four Words, 26
inactive window, 252
initialize, 253
Insertion Point or Bar, 30, 56, 82, 83, 92, 253
Insertion Bar,
Backspace key to edit, 82
large and extended text, 92
set speed of blinking, 56
insurance, 192, 193
home, 192
work, 193
internal speaker, 56, 199, 206,
also see sound
set volume, 56
international Macs, 207
interpreters, 158, 159
INVERT, 175, 176
invert, 32, 34, 46, 253
Tab key, 36
two-key rollover, 39
typeahead, Event queue, 39

L
label, 172
MacBASIC, 172
Lisa, 13
LISP, 167, 254
LOGO, 165, 254
London, 254, 301
loop, 173

M
MacBASIC, 161, 164, 170-178,
214, see also Appendix E
ASKPENPOS, 177
BOUNDS, 178
BUTTON, 178
concurrency, 170
data types, 174
boolean data types, tilde (t), 174
double precision real-default, 174
extended precision real, backslash, 174
flow of control structures, 173
DO-LOOP, 173
EXIT, 173
FOR-TO-NEXT loop, 173
IF-ENDIF, 174
IF-THEN-(ELSE)-ENDIF, 173
graphics, 175
BUTTONSTATUS, 178
coordinates, 175
DIALSTATUS, 178
ERASE, 175, 176
FRAME, 175, 176
graphics commands, 175
ASK PENPOS, 177
colon, 172

J
Jobs, Steve, 7

K
Kare, Susan, 134
Kay, Alan, 11
Key Caps, 55, see also Appendix D
Option key, 55
shift key, 55
shifted characters, 55
typing with the mouse, 55
keyboard, 35, 36, 39, 40, 56, 57, 197
American and European, 36
auto-repeat, 39, 56, 57
layout, 36
Return key, 36
security chain, 197
software-mapped, 40
GOSUB, 172
GOTO, 172
INVERT, 175, 176
label, 172
manipulation of shapes, 175
MENUSTATUS, 178
optional line numbers, 171
OVAL, 175
PATTERN, 178
pen, 177
PENMODE, 177
PENPOS, 177
PLOT, 177
PORT, 178
PRINT, 172
QuickDraw routines, 175
RECT, 175
ROUNDRECT, 175
SET PENPOS, 177
SET/ASK, 177
SHOW/REMOVE, 177
VIEWPORT, 178
user interface, 177
VPOS, 178
WINDOW, 178
syntax, 160
Macintosh Pascal, 181, 214
Check, 181
GO, 181
output window, 181
program window, 181
syntax errors, 181
Macintosh System Font, 62
MacPaint, 94, 133, 254
drawing tools, see tools, for
drawing
graphic routines,
insert text, 136
Print, 139
QuickDraw, 14, 134
Scrapbook, 94
screen, 139
MacWrite, 24, 25, 90, 123, 254
Cut, Copy, Paste, 89
icons, 24
features, 124
Clipboard, 124
decimal tabs, 124
headers and footers, 124
Insertion Bar, 82
printing features,
multiple copies, 125
paper size, 125
print quality/printing
rulers, 124
speeds, 125
range of pages, 125
search and replace, 124
spacing options, 124
style options, 124
text formats, 124
type fonts, 124
type sizes, 124
Undo typing, 124
user-set tabs, 124
MC68000, 4, 205, 210, 217-222,
see also Appendix F
design, 4, 217-221
address,
32-bit program counter, 221
32-bit stack pointers, 221
addressable memory space,
217
architecture, 217
arithmetic and logic opera-
tions, 218
data, 218, 219
data sizes, 217
data types, 218
debugging aids [trace mode],
221
fast instructions, 218
flat memory, 219
instruction set, 217
instructions, 218
memory management units,
218
microprogramming, 218
multiprocessing, 218
multitasking, 218
number, type of registers, 217
motherboard, 202, 204
digital processing, 205
electrical ground, 208
MC68000, 205
PALs, 205
RAM, 205
RF emissions, 208
ROM, 205
Serial Communication Controller (SCC), 205
Versatile Interface Adapter (VIA), 205
Motorola, 210
MC68000 microprocessor, see MC68000
mouse, 28, 29, 31, 55-58, 256
clicking, 29, 31
double-clicking, 29, 31
dragging, 29, 31, 33-35
press, 29
selecting an accessory, 56, 57
set mouse in motion, 56, 57
set time of double-click, 56, 58
shift-clicking, 29, 31
socket, 197
to edit, 89
typing in Key Caps, 55
Multiplan, 143, 144
compatibility, 148
electronic worksheet, 144
formulas, 146
functions, 145
list of functions, 146
Microsoft manuals, 148
simple applications, 148
templates, 148
trouble, 148

N

networking, 198
New York, 257, 296
non-hollow cross, 30
Note Pad, 55
numbers, 53, 96
adding, 53
dividing, 53

opcodes, 220
registers, 218, 219
address, 218, 219
speed, 217, 221, 222
16-bit wide data bus, 222
benchmarks, 222
instruction speed, 222
pre-fetch queue, 222
status register, 220
system byte, 220, 221
interrupt mask, 220
supervisory mode, 221
trace mode
system dispatcher, 210
memory, 4, 95-100
address, 96
ASCII, 97
bit, 95
byte, 95
data, 97
disk, 4
Heap, 100
programs, 97
RAM, 4, 97, 99, 205
ROM, 4, 95, 97, 99, 205
Menu, 255
set twinkle, 56, 58
Menu Bar, 33, 35, 53, 255
black Apple, 53
Menus, 20
microprocessor, 255, see also
MC68000 history, 216
Microsoft,
Chart, 143, 149, also see Chart
File, 143
manuals, 148
Multiplan, 143, 144
Word, 143
Modal Dialog Box, 73, 255
mode, 256
Modeless Dialog Box, 73, 255
modeless operation, 12
modem, 118, 197, 231, 256
socket, 197
modes, 71
Modula 2, 166, 256
Monaco, 256, 298
monitor, 256
hexadecimal, 96
multiplying, 53
subtracting, 53
numeric keypad, 40, 41, 257
Clear and Enter, 40
comma, 40
field motion keys, 41
numbers, 400

O

object, 20
Okay, 72
on/off switch, 197
Open,
a window, 61
Open, 257
operating system, 43, 44, 212, 257
asynchronous I/O, 212
copying files, 43
displaying files, 43
file system, 212
Event Manager, 212
files and volumes, 212
formatting disks, 43
input/output, 212
loading files, 43
Memory Manager, 212
renaming and erasing files, 43
subdirectories, 43
Option key, 36, 55, 257
Key Caps, 55
OVAL, 175

P

PAINT, 175, 176
paint brush, 135
Brush Mirrors, 138
brush shapes, 135
paint bucket,
patterns, 137
paint sprayer, 135
patterns, 137
PALs (Programmable Array Logic), 205
panes, 69
paradigm, 258
Pascal, 165, 258
debugging, 182
editing, 181, 182
interpreted, 165, 181
p-code, 166
pre-written functions and procedures, 182
MoveWindow, 182
DragWindow, 182
GrowWindow, 182
SizeWindow, 182
reserved words, 182
Step, 182
UCSD Pascal, 166
Paste, 89, 90, 258
Pasting Madness, 90
patterns, 137
pen,
MacBASIC, 177
pencil, 135
black or white, 135
PENMODE, 177
PENPOS, 177
PENSIZE, 177
PILOT, 167
PL/I, 166
PLOT, 177
pointer, 65, 259
hierarchy of on-topness, 65
I-bar, 83
pointer shapes, 29, 259
arrow, 29
hollow cross, 30
Insertion Bar, 30
wristwatch, 31
Point-Press-Drag-Release, 33
preamplifier, 199
precautions, 192
dangerous voltages, 192
pre-defined shape and line templates, 136
Preemption Dilemma, 13
preferences, 56
pressing, 29, 33, 68, 259
scrolling, 33
to scroll, 68
principal tool, 25, 260
PRINT, 172
Print, 71
MacPaint, 139
socket, 197
programmers, 217
MC68000, 217, see also
MC68000, design
size, 217
speed, 217
programming, 157-161
algorithms, 157, 158
coding, 161
compilers, 158, 159
hacking, 161
interpreters, 158, 159
MC68000, 158, 159
machine language, 157
problem definition, 160
speed, 159
stepwise refinement, 161
syntax, 160
programming languages, 157
Ada, 166
APL, 167
Assembler, 164
C, 166
coding, 161
COBOL, 167
compilers, 158, 159
FORTH, 167
FORTRAN, 167
hacking, 161
interpreters, 158, 159
LISP, 167
MacBASIC, 161, 164
Modula 2, 166
PL/1, 166
Pascal, 165
PILOT, 167
problem definition, 160
PROLOG, 167
syntax, 160
programs, 97, 99, 100
large, 99
overlays, 100
virtual memory, 99
PROLOG, 167
pull-down menus, 65, 261
hierarchy of on-topness, 65
Put Back, 51
Puzzle, 54
QuickDraw, 14, 134, 175, 212,
213, 261
Quit, 45
radio button, 261
RAM, 4, 95, 99, 205, 210, 261
new routines, 210
reboot the system, 77
RECT, 175
reference guide, 208
resource files, 26
Resource Manager, 261
Return, 38, 86
to mark end of paragraphs, 86
Return key, 36, 262
RF emissions, 208
ROM, 4, 62, 95, 97, 99, 205, 210-
214, 262, see also system
dispatcher
call, 211
information, 211
name of routine, 211
value returned, 211
location, 211
operating system, 212
QuickDraw routines, 212
User Interface Toolbox, 212
ROUNDRECT, 175
routine care, 187-191
dust, 190
keyboard, 189
mouse, 189
screen care, 188
ventilation, 191
RS-232, 262
RS-422, serial interfaces, 197, 198,
262
hard disk drive, 198
modem port, 198
printer port, 198
ruler, 262

San Francisco, 305
Scrap, 101
Scrapbook, 56, 94
MacPaint, 94
screen coordinates, 175, 176
HorizontalPoint, 176
semi-colon, 176
VerticalPoint, 176
scroll arrows, 263, see also Scroll Box
Scroll Bars, 67, 263, see also elevators
Scroll Box, 68, 92, 263
scroll arrows, 68, 92
scrolling, 92, 263
backwards, 92
search menu, 263
Secondary tools, 25
security chain, 197
selection, 263
selection lasso, 136
selection rectangle, 33, 136
semi-colon,
screen coordinates, 176
Serial Communication controller (SCC), 205
Serial interface, 264
SET/ASK, 177
SET PENPOS, 177
shift-clicking, 29, 31, 32, 92
discontiguous icons, 92
scrolling, 92
selecting discontiguous icons, 32
shift keys, 36, 55, 264
Key Caps, 55
Show Clipboard, see Clipboard
Show Page, 138
SHOW/REMOVE, 177

68000, see MC68000
Size Box, 66, 264
Smith, Burrell, 202
Socrates, 121
sound, 199, 204, 206
audio out jack, 199
external speaker, 199
internal speaker, 199, 206
preamplifier, 199
special keys, 38
Enter, 38
"Okay" with Return and Enter, 38
Return, 38
Tab, 38
split bar, 69, 265
Stanford Homebrew Computer Club, 7
STEP, 173
Stepwise Refinement, 161
strange symbol, 37
straight lines, 136
style menu, 136
subdirectory, 52
system crashes, 76, 77
reboot the system, 77
return to Finder, 77
system dispatcher, 210
look-up table, 210
undefined instructions, 210
System Font, 62
System windows, 62

Tab, 38
Tab key, 36, 266
telecommunications, 231
modem, 231, also see communications
text, large and extended
selections, 92
text documents, 21, 22
strings of characters, 22
Think Technologies, 183
thumb, 266
tilde (t), 174
title bar, 62, 66, 266
   in a document window, 62, 66
Tool files, 26
Toolbox, 213
   ASCII text, 214
   Control Manager, 213
custom, 214
data types, 214
   Desk Manager, 213
Dialog Manager, 213
Font Manager, 213
Menu Manager, 213
QuickDraw, 213
Resource Manager, 213
System Error Handler 213
TextEdit, 213
Toolbox Event Manager, 213
tools, 23, 24, 267
   application program, 24
   business, 116
   communications, 117, 118
data management, 114
   education, 118
for drawing, 135-138
   borders, 137
Brush Mirrors, 138
   eraser, 135
Fat Bits, 138
freehand lines, 137
Grid aid, 138
Hand, 136
hollow or filled rectangles, 136
hollow or filled rounded-corner rectangles, 137
insert text, 136
   paint brush, 135
   paint bucket, 135
   paint sprayer, 135
   patterns, 137
   pencil, 135
   rectangles, hollow or filled, 136
   pre-defined shape and line templates, 136
   rounded-corner rectangles, hollow or filled, 137
   selection rectangle and lasso, 136
   Show page, 138
   straight lines, 136
icons, 24
Toronto, 266
traveling, 229
Trash, 45, 49, 63
two-key rollover, 39
typeahead, 39, 267

U
Undo, 268
Paste, 90
typing, 34, 85
user groups, 168
user interface, 29, 177, 212, see also Mac BASIC
events, 29
   Toolbox, 212, 268

V
variable names, 174
Venice, 268, 300
Versatile Interface Adapter (VIA), 205
video display, 19
video tube, 202
View menu, 47
VisiCalc, 116, 120
Volumes, 107

W
window, 13, 20, 51, 61-65, 69, 269
   Alert Box, 20
Close Box, 51, 66
Dialog Box, 20, 66
dismiss, 61
Document, 62
dragging, 61
duplicate, 64
hierarchy of on-topness, 65
icon, 61
Menus, 20
open, 61
overlapping, 63
resize, 61
splitting, 69
panes, 69
split bar icon, 69
Size Box, 66
to close, 51
window frame, 66, 67
Close Box, 66
go-away region, 66
Scroll Bar, 67
Size Box, 66
grow region, 66
Title Bar, 66
drag region, 66
Window Manager, 62, 66
content region, 66
document windows, 62
structure region, 66
system windows, 62
word processor, 113, 123, also see
MacWrite, MC68000
Microsoft Word, 143
other, 126
word wrap, 82
work insurance, see insurance
workstyles, 228
Wozniak, Steve, 7, 203
wristwatch, 31
write-protect, 269

X

Xerox Alto, 11
Xerox Star, 11
Enthusiasm. Acuity. Expertise. Wit. Enthusiasm. This is Doug Clapp, microcomputer aficionado turned Macintosh zealot, InfoWorld opinion maker, Softalk Mac guru.

Taste a sampling of the wisdom of the really, really great Mr. Clapp:

"It's easier to merely want something than actually to do it. Often, some of the best ideas are impractical—the technology isn't perfected, or affordable, or even yet invented."

"Macintosh can do all the things you'd expect a computer to do. It also does things you might not have believed any machine could do."

"If Socrates were alive today, he might conclude that 'The unexamined use of computers is unforgivable.'"

"The folly of waiting to buy a computer is that while you're waiting, you don't have a computer. Not having a computer means that life, in many ways, is more difficult than it needs to be."