INCLUDES MICROSOFT®
VISUAL BASIC 6.0 WORKING MODEL

AN INTRODUCTION TO PROGRAMMING USING
Visual Basic® 6.0

FOURTH EDITION

DAVID I. SCHNEIDER

An Introduction to Programming

USING VISUAL BASIC® 6.0

Fourth Edition

David I. Schneider
University of Maryland

Prentice Hall
Upper Saddle River, New Jersey 07458

Library of Congress Cataloging-in-Publication Data

Schneider, David I.
An introduction to programming using Visual Basic 6.0
David I. Schneider—4th ed.
p. cm.
Includes index.
ISBN: 0-13-936428-5
1. BASIC (Computer program language). 2. Microsoft Quick BASIC.
2. Microsoft Visual BASIC. I. Title.
CIP data available.

Publisher: ***Alan Apt***
Acquisitions Editor: ***Laura Steele***
Production Editor: ***Rhodora Penaranda***
Editor-in-Chief: ***Marcia Horton***
Assistant Vice President of Production and Manufacturing: ***David W. Riccardi***
Managing Editor: ***Eileen Clark***
Manufacturing Manager/Buyer: ***Trudy Pisciotti***
Cover Director: ***Heather Scott***
Cover Designer: ***Tamara Newnam-Cavallo***
Editorial Assistant: ***Kate Kaibni***
Compositor: ***Rebecca Evans & Associates***

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author andpublisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentationcontained in this book. The author and publisher shall not be liable in any event for incidental or consequential damagesin connection with, or arising out of, the furnishing, performance, or use of these programs.

Printed in the United States of America.

20 19 18 17 16 15 14 13 12 11

ISBN 0-13-936428-5

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall Japan, Inc., *Tokyo*
Prentice-Hall Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

PREFACE

This text provides an introduction to programming using Microsoft® Visual Basic™ 6.0 on IBM PC and IBM PC compatible computers running Windows.

Due to its extraordinary combination of power and ease of use, Visual Basic has become the tool of choice for developing user-friendly Windows applications in the business world. In addition, Microsoft has made Visual Basic the language used to take full control of its best selling Windows applications such as Microsoft Word, Access, and Excel. Not only is Visual Basic the state of the art in Basic programming, but Visual Basic is fun! Learning Visual Basic was very exciting to me, and most students have similar reactions when they see how easy it is to build powerful visual interfaces using it.

My objectives when writing this text were as follows:

1. To develop focused chapters. Rather than covering many topics superficially, I concentrate on important subjects and cover them thoroughly.
2. To use examples and exercises with which students can relate, appreciate, and feel comfortable. I frequently use real data. Examples do not have so many embellishments that students are distracted from the programming techniques illustrated.
3. To produce compactly written text that students will find both readable and informative. The main points of each topic are discussed first and then the peripheral details are presented as comments.
4. To teach good programming practices that are in step with modern programming methodology. Problem-solving techniques and structured programming are discussed early and used throughout the book.
5. To provide insights into the major applications of computers.

Unique and Distinguishing Features

Exercises for Most Sections. Each section that teaches programming has an exercise set. The exercises both reinforce the understanding of the key ideas of the section and challenge the student to explore applications. Most of the exercise sets require the student to trace programs, find errors, and write programs. The answers to all the odd-numbered exercises in Chapters 2 through 6 and selected odd-numbered exercises from Chapters 7 through 14 are given at the end of the text.

Practice Problems. Practice problems are carefully selected exercises located at the end of a section, just before the exercise set. Complete solutions are given following the exercise set. The practice problems often focus on points that are potentially confusing or are best appreciated after the student has worked on them. The reader should seriously attempt the practice problems and study their solutions before moving on to the exercises.

Programming Projects. Beginning with Chapter 3, nearly every chapter contains programming projects. The programming projects not only reflect the variety of ways that computers are used in the business community, but also present some games and general-interest topics. The large number and range of difficulty of the programming projects allow adapting the course to the students' interests and abilities. Some programming projects in later chapters can be assigned as end-of-the-semester projects.

Comments. Extensions and fine points of new topics are defered the "Comments" portion at the end of each section so that they will not interfere with the flow of the presentation.

Case Studies. Each of the four case studies focuses on an important programming application. The problems are analyzed and the programs are developed with hierarchy charts and pseudocode. The programs are available to students on an accompanying CD.

Chapter Summaries. In Chapters 3 through 14 the key results are stated and the important terms are summarized at the end of the chapter.

Procedures. The early introduction of general procedures in Chapter 4 allows structured programming to be used in simple situations before being applied to complex problems. However, the text is written so that the presentation of procedures easily can be postponed until decision and repetition structures have been presented. In Chapters 5 and 6 (and Sections 7.1 and 7.2), all programs using procedures appear at the ends of sections and can be deferred or omitted.

Arrays. Arrays are introduced gently in two sections. The first section presents the basic definitions and avoids procedures. The second section presents the techniques for manipulating arrays and shows how to pass arrays to procedures.

Appendix on Debugging. Placing the discussion of Visual Basic's sophisticated debugger in Appendix D allows the instructor flexibility in deciding when to cover this topic.

Reference Appendices. Appendices serve as a compact reference manual for Visual Basic's environments and statements.

Visual Basic 6.0 Included. A Working Model Edition of Visual Basic 6.0 is packaged with every copy of this book. The Working Model is identical to the Learning Edition with the exceptions that it cannot compile to an EXE file and it does not have on-line help. In particular, the Working Model contains the Data control and the FlexGrid control.

Examples and Case Studies Files. The programs from all examples and case studies from the text have been copied onto the accompanying CD. The CD also contains all databases and text files used in the examples, and many of the text files used in exercises.

Instructors Diskette. A diskette containing every program in the text, the solution to every exercise and programming project, and a test item file for each chapter is available to the instructor.

What's New in the Fourth Edition

1. Several functions that are new in Visual Basic 6.0 are introduced and used extensively. They include Round, FormatNumber, FormatCurrency, and FormatPercent.

2. In the database chapter, new material has been added on using data-bound controls. In particular, the use of the FlexGrid, data-bound list box, and data-bound combo box are illustrated with examples.

3. The keyword Let has been eliminated from assignment statements and the keyword Rem has been replaced by an apostrophe.

4. Boolean variables have been introduced and are used for flags.

5. The real-life data in the examples and exercises have been updated and revised.

6. The version of Visual Basic has been upgraded to Visual Basic 6.0. The VB6.0 software packaged with the book now contains the Data control and the FlexGrid control.

7. Engineering Supplement: Contains additional examples, case studies, and exercises from engineering, mathematics, and the sciences.

8. Companion Web site for instructors and students: (Available in Spring 1999) This Web site will provide an on-line study guide for students that includes additional exercises and Visual Basic learning resources. The instructors' portion will include, among other materials, PowerPoint

ACKNOWLEDGMENTS

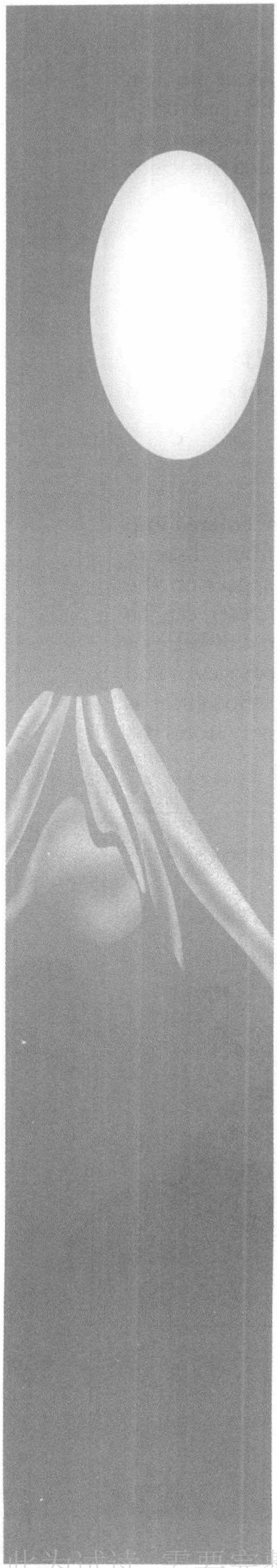
Many talented instructors, students, and programmers provided helpful comments and thoughtful suggestions at each stage in the preparation of this text. I extend my gratitude for their contributions to the quality of the book to A. Abonomah, University of Akron; Timothy Babbitt, Rochester Institute of Technology; William Barnett, Northwestern State University; Sherry Barriclow, Grand Valley State University; Robert Berman, Wayne State University; William Burrows, University of Washington; David Chao, San Francisco State University; Christopher Chisolm, University of Nebraska, Omaha; Robert Coil, Cincinnati State Technical and Community College; Gary Cornell, University of Connecticut; Ronit Dancis; John DaPonte, Southern Connecticut State Univ.; Ward Deutschman, Briarcliff; Ralph Duffy, North Seattle Community College; Charles Fairchild; Pat Fenton, West Valley College; David Fichbohm, Golden Gate University; Robert Fritz, American River College; Matthew Goddard, New Hampshire Technical College; Mickie Goodro, Casper College; Wade T. Graves, Grayson Community College; Christine Griffin; Gary Haw, MIPS Software Dev. Inc.; Shelly Hawkins, Microsoft; Tom Janicki, Kent State University; Dana Johnson, North Dakota State University; Dan Joseph, Rochester Institute of Technology; Del Kimber, Clemson University; Wanda Kunkle, Rowan College; Paul Lecoq, San Francisco Community College; David Leitch, Devry Institute; David Letcher, The College of New Jersey; Kieran Mathieson, Oakland University; Charlie Miri, Delaware Tech; George Nezlek, DePaul University; Ron Notes, Hebrew Academy of Greater Washington; Mike Paul, Berry University; T. S. Pennington, Maple Woods Community College; Arland Richmond, Computer Learning Center; David Rosser, Essex County College; Arturo Salazar, San Francisco State; Susanne Peterson, Microsoft; Janie Schwark, Microsoft; Mike Talber, Portland Community College; Steve Turek, Devry Institute of Technology, Kansas City; Jac Van Deventer, Washington State University; Randy Weinberg, St. Cloud State University; Laurie Werner, Miami University; Melinda White, Santa Fe Community College; Ronald Williams, Central Piedmont Community College.

Many people are involved in the successful publication of a book. I wish to thank the people at Prentice Hall whose support and diligence made this textbook possible. Trudy Pisciotti, Rhodora Penaranda, and Bayani DeLeon did a fantastic job producing the book and keeping it on schedule. Heather Scott and Tammy Cavallo created the innovative covers for my Visual Basic series. Alan Apt, Marcia Horton, and Kate Kaibni provided the needed editorial support and assistance.

I extend special thanks to my editor Laura Steele and my typesetter Rebecca Evans. Laura's ideas and enthusiasm helped immensely with the preparation of the book. Rebecca's considerable skills and congenial manner made for an uncomplicated and pleasant production process.

Last, but not least, I am grateful to the Microsoft Corporation for its commitment to producing outstanding programming languages and for its permission to include a copy of the Working Model Edition of Visual Basic 6.0 with each book.

CONTENTS



1 An Introduction to Computers and Visual Basic

1.1 AN INTRODUCTION TO COMPUTERS

An Introduction to Programming Using Visual Basic 6.0 is a book about problem solving with computers. The programming language used is Visual Basic, but the principles taught apply to many modern programming languages. The examples and exercises present a sampling of the ways that computers are used in society.

Computers are so common today that you certainly have seen them in use and heard some of the terminology applied to them. Here are some of the questions that you might have about computers and programming.

Question: What is meant by a personal computer?

Answer: The word "personal" does not mean that the computer is intended for personal, as opposed to business, purposes. Rather, it indicates that the machine is operated by one person at a time instead of by many people.

Question: What are the main components of a personal computer?

Answer: The visible components are shown in Figure 1.1. Instructions are entered into the computer by typing them on the keyboard or by reading them from a diskette in a diskette drive or from a hard disk. Characters normally appear on the monitor as they are typed. Information processed by the computer can be displayed on the monitor, printed on the printer, or recorded on a diskette or hard drive. Hidden from view inside the system unit are the microprocessor and the memory of the computer. The microprocessor, which can be thought of as the brain of the computer, carries out all computations. The memory stores the instructions and data that are processed by the computer.

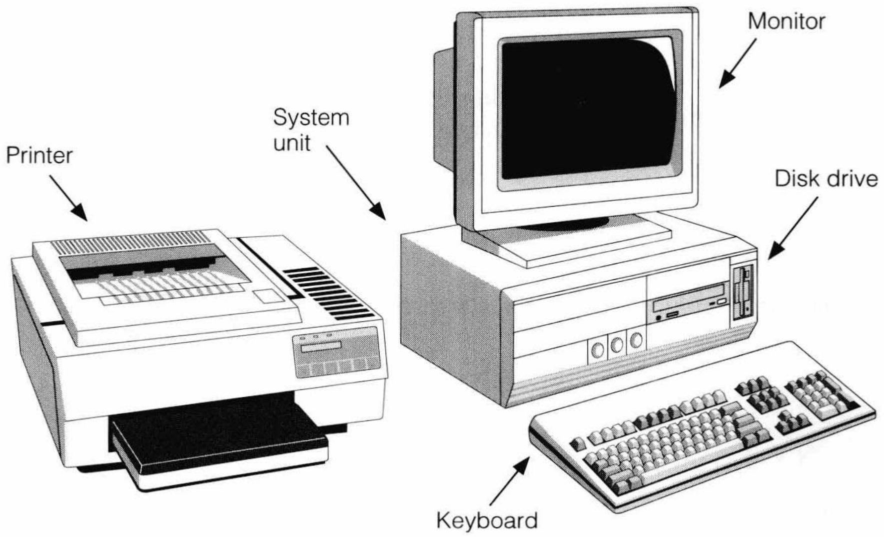


Figure 1.1 Components of a personal computer.

Question: What are some uses of computers in our society?

Answer: Whenever we make a phone call, a computer determines how to route the call and calculates the cost of the call. Banks store all customer transactions on computers and process these data to revise the balance for each customer. Airlines record all reservations into computers. This information, which is said to form a database, can be accessed to determine the status of any flight. NASA uses computers to calculate the trajectories of satellites. Business analysts use computers to create pie and bar charts that give visual impact to data.

Question: What are some topics covered in this text that students can use immediately?

Answer: Computer files can be created to hold lists of names, addresses, and phone numbers, which can be alphabetized and printed in entirety or selectively. Line graphs or attractive tables can be created to enhance the data in a term paper. Mathematical computations can be carried out for science, business, and engineering courses. Personal financial transactions, such as bank deposits and loans, can be recorded, organized, and analyzed.

Question: How do we communicate with the computer?

Answer: Many languages are used to communicate with the computer. At the lowest level, there is machine language, which is understood directly by the microprocessor but is awkward for humans. Visual Basic is an example of a higher-level language. It consists of instructions to which people can relate, such as Print, Input, and Do. The Visual Basic software translates Visual Basic programs into machine-language programs.

Question: How do we get computers to perform complicated tasks?

Answer: Tasks are broken down into a sequence of instructions that can be expressed in a computer language. (This text uses the language Visual Basic.) The sequence of instructions is called a program. Programs range in size from two or three instructions to tens of thousands of instructions. Instructions are typed on the keyboard and stored in the computer's memory. (They also can be stored permanently on a diskette or hard disk.) The process of executing the instructions is called running the program.

Question: Are there certain features that all programs have in common?

Answer: Most programs do three things: take in data, manipulate them, and give desired information. These operations are referred to as input, processing, and output. The input data might be held in a portion of the program, reside on a diskette or hard drive, or be provided by the computer operator in response to requests made by the computer while the program is running. The processing of the input data occurs inside the computer and can take from a fraction of a second to many hours. The output data are either displayed on the screen, printed on the printer, or recorded onto a disk. As a simple example, consider a program that computes sales tax. An item of input data is the cost of the thing purchased. The processing consists of multiplying the cost by a certain percentage. An item of output data is the resulting product, the amount of sales tax to be paid.

Question: What are the meanings of the terms *hardware* and *software*?

Answer: The term **hardware** refers to the physical components of the computer, including all peripherals, central processing unit, disk drives, and all mechanical and electrical devices. Programs are referred to as **software**.

Question: What are the meanings of the terms *programmer* and *user*?

Answer: A **programmer** is a person who solves problems by writing programs on a computer. After analyzing the problem and developing a plan for solving it, he or she writes and tests the program that instructs the computer how to carry out the plan. The program might be run many times, either by the programmer or by others. A **user** is any person who uses a program. While working through this text, you will function both as a programmer and as a user.

Question: What is meant by *problem solving*?

Answer: Problems are solved by carefully reading them to determine what data are given and what outputs are requested. Then a step-by-step procedure is devised to process the given data and produce the requested output. This procedure is called an **algorithm**. Finally, a computer program is written to carry out the algorithm. Algorithms are discussed in Section 2.2.

Question: What types of problems are solved in this text?

Answer: Carrying out business computations, creating and maintaining records, alphabetizing lists, and drawing line graphs are some of the types of problems we will solve.

Question: What is the difference between standard BASIC and Visual Basic?

Answer: In the early 1960s, two mathematics professors at Dartmouth College developed BASIC to provide their students with an easily learned language that could tackle complicated programming projects. As the popularity of BASIC grew, refinements were introduced that permitted structured programming, which increases the reliability of programs. Visual Basic is a version of BASIC that was written by the Microsoft Corporation to incorporate object-oriented programming into BASIC and to allow easy development of Windows applications.

1.2 USING WINDOWS

Programs such as Visual Basic, which are designed for Microsoft Windows, are supposed to be easy to use—and they are once you learn a little jargon and a few basic techniques. This section explains the jargon, giving you enough of an understanding of Windows to get you started in Visual Basic. Although Windows may seem intimidating if you've never used it before, you need to learn only a few basic techniques, which are covered right here.

Mouse Pointers

When you use Windows, think of yourself as the conductor and Windows as the orchestra. The conductor in an orchestra points to various members, does something with his or her baton, and then the orchestra members respond in certain ways. For a Windows user, the baton is called the **pointing device**; most often it is a **mouse**. The idea is that as you move the mouse across your desk, a pointer moves along the screen in sync with your movements. Two basic types of mouse pointers you will see in Windows are an arrow and an hourglass.

The **arrow** is the ordinary mouse pointer you use to point at various Windows objects before activating them. You will usually be instructed to "Move the pointer to . . . ." This really means "Move the mouse around your desk until the mouse pointer is at . . . ."

The **hourglass** mouse pointer pops up whenever Windows is saying: "Wait a minute; I'm thinking." This pointer still moves around when you move the mouse, but you can't tell Windows to do anything until it finishes what it's doing and the mouse pointer no longer resembles an hourglass. (Sometimes you can press the Esc key to tell Windows to stop what it is doing.)

Note: The mouse pointer can take on many other shapes, depending on which document you are using and what task you are performing. For instance, when entering text in a word processor or Visual Basic, the mouse pointer appears as a thin, large, uppercase I (referred to as an I-Beam).

Mouse Actions

After you move the (arrow) pointer to a place where you want something to happen, you need to do something with the mouse. There are four basic things you can do with a mouse—point, click, double-click, and drag.

Pointing means moving your mouse across your desk until the mouse pointer is over the desired object on the screen.

Clicking (sometimes people say single-clicking) means pressing and releasing the left mouse button once. Whenever a sentence begins "Click on . . . ," you need to

1. Move the mouse pointer until it is at the object you are supposed to click on.
2. Press and release the left mouse button.

An example of a sentence using this jargon might be "Click on the button marked Yes." You also will see sentences that begin "Click inside the. . . ." This means to move the mouse pointer until it is inside the boundaries of the object, and then click.

Double-clicking means clicking the left mouse button twice in quick succession (that is, pressing it, releasing it, pressing it, and releasing it again *quickly* so that Windows doesn't think you single-clicked twice). Whenever a sentence begins "Double-click on . . . ", you need to

1. Move the mouse pointer until it is at the object you are supposed to double-click on.
2. Press and release the left mouse button twice in quick succession.

For example, you might be instructed to "Double-click on the little box at the far left side of your screen."

Note: An important Windows convention is that clicking selects an object so you can give Windows or the document further directions about it, but double-clicking tells Windows (or the document) to do something.

Dragging usually moves a Windows object. If you see a sentence that begins "Drag the . . . ", you need to

1. Move the mouse pointer until it is at the object.
2. Press the left mouse button, and hold it down.
3. Now move the mouse pointer until the object moves to where you want it to be.
4. Finally, release the mouse button.

Sometimes this whole activity is called *drag and drop*.

Starting Windows

Windows starts automatically when you turn on your computer. After a little delay, you will first see the Windows logo and finally a screen looking something like Figure 1.2. The little pictures (with labels) are called **icons**. You double-click on the My Computer icon to see your computer's contents and manage your files. You click on the **Start button** (at the bottom left corner of the screen) to run programs such as Visual Basic, end Windows, and carry out several other tasks. (The Start menu also can be accessed with Ctrl+Esc.)

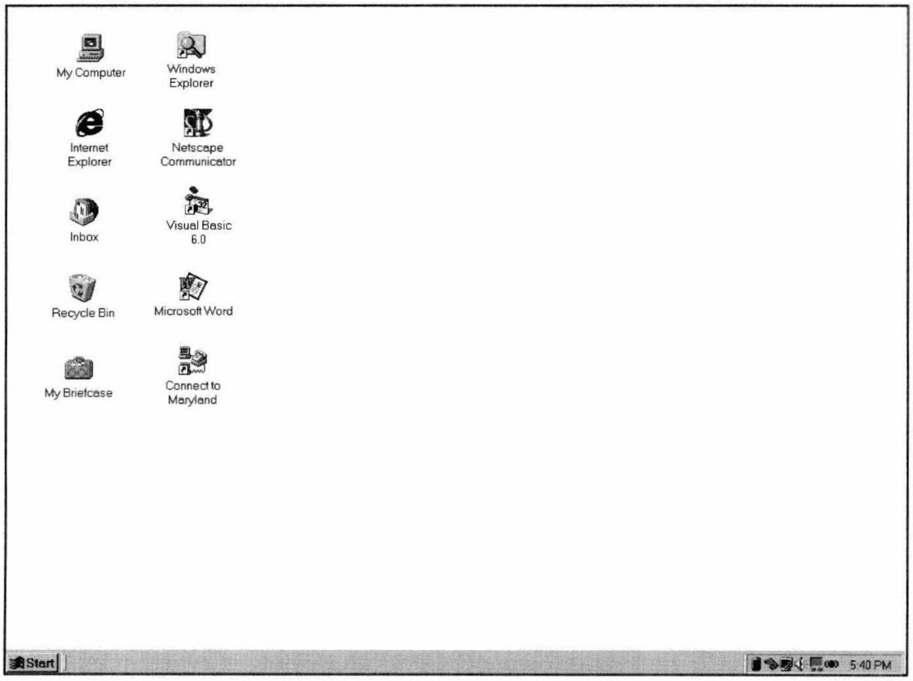


Figure 1.2 Windows desktop.

Windows and Its Little Windows

Windows gets its name from the way it organizes your screen into rectangular regions. When you run a program, the program runs inside a bordered rectangular box. Unfortunately Windows jargon calls all of these windows, so there's only a lowercase "w" to distinguish them from the program called Windows.

When Windows' attentions are focused on a specific window, the bar at the top of the window is highlighted and the window is said to be **active**. The active window is the only one that can be affected by your actions. An example of a sentence you might see is "Make the window active." This means that if the title bar of the window is not already highlighted, click inside the window. At this point, the (new) window will be responsive to your actions.

Using the Notepad

We will explore the Windows application Notepad to illustrate the Windows environment. The Notepad is used extensively in this text to create data files for programs. Most of the concepts learned here carry over to Visual Basic and other Windows applications.

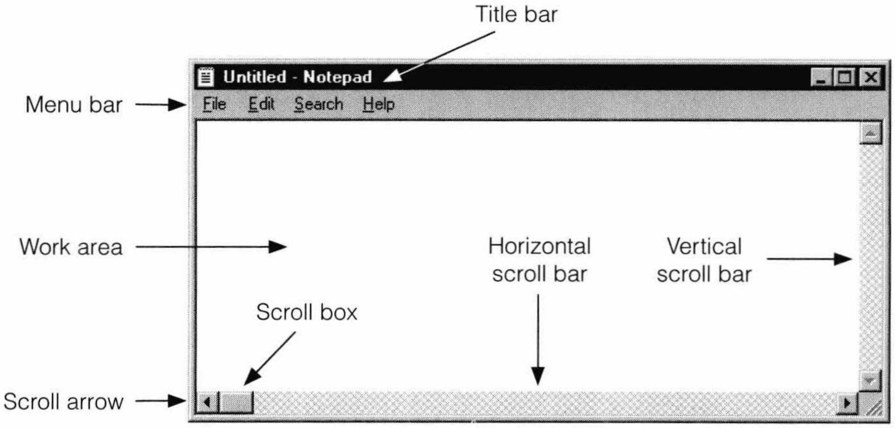


Figure 1.3 The Notepad window.

To invoke Notepad from Windows, click the Start button, point to Programs, point to Accessories, and click Notepad. As its name suggests, Notepad is an elementary word processor. You can type text into the Notepad window, edit the text, print the text on the printer, and save the text for later recall.

The blinking vertical line is called the **cursor**. Each letter you type will appear at the cursor. The Notepad window is divided into four parts. The part containing the cursor is called the **Work area**. It is the largest and most important part of the window because documents are typed into this window.

The **Title bar** at the top of the screen holds the name of the document currently being written. Until the document is given a name, the document is called "Untitled."