Understanding Computers Today and Tomorrow

Chapter 13:

Programming Languages and Program Development

Learning Objectives

1. Understand the differences between structured programming, object-oriented programming (OOP), aspect-oriented programming (AOP), and adaptive software development.
2. Identify and describe the activities involved in the program development life cycle (PDLC).
3. Understand what constitutes good program design and list several tools that can be used by computer professionals when designing a program.

Learning Objectives

4. Explain the three basic control structures and how they can be used to control program flow during execution.
5. Discuss some of the activities involved with debugging a program and otherwise ensuring it is designed and written properly.
6. List some tools that can be used to speed up or otherwise facilitate the program development process.
7. Describe several programming languages in use today and explain their key features.

Overview

- This chapter covers:
  - The most common approaches to program design and development
  - The phases of the program development life cycle (PDLC)
  - Tools that can be used to design and develop a program
  - Good program design techniques and types of program errors
  - Common programming languages

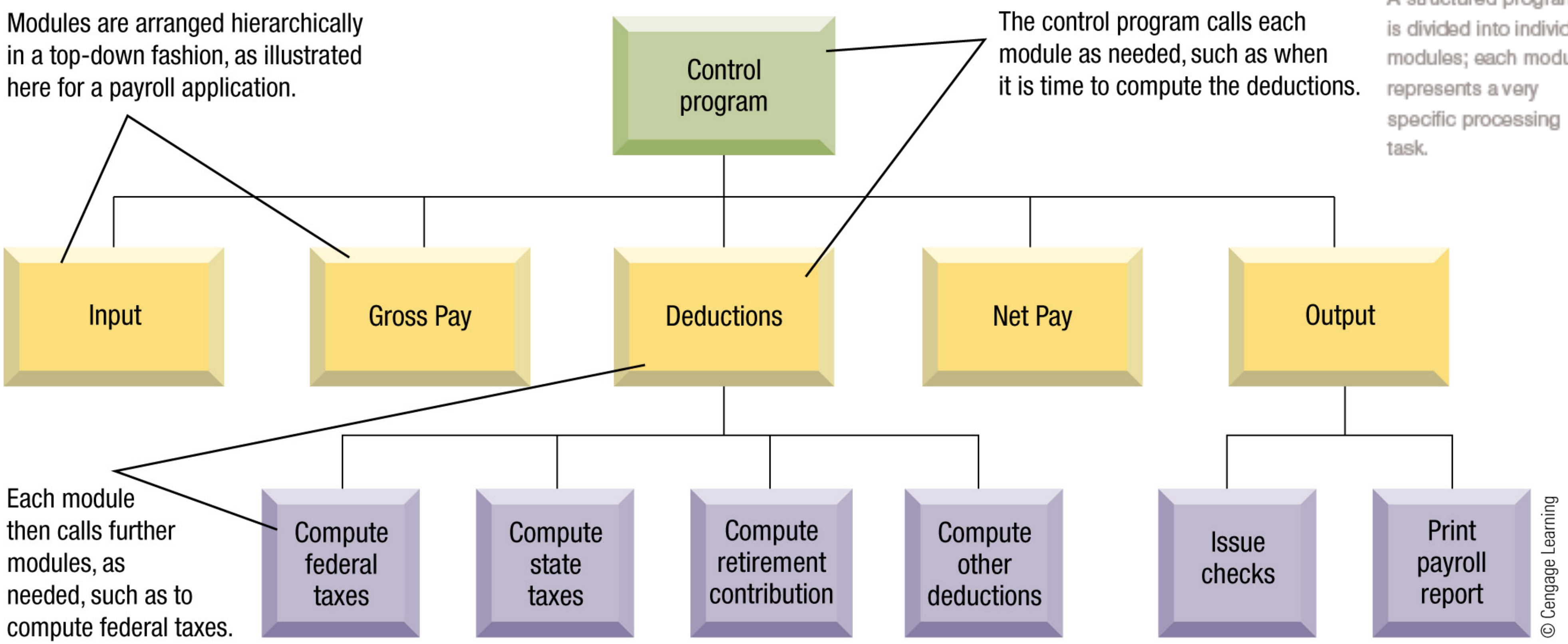
Approaches to Design and Development

- Procedural Programming
  - An approach to program design in which a program is separated into small modules that are called by the main program or another module when needed
    - Procedure call—locating specific tasks in procedures (modules or subprograms) that are called by the main program when needed
    - Allows each procedure to be performed as many times as needed; multiple copies of code not needed
    - Prior to procedural programming, programs were one large set of instructions (used GOTO statements)

Approaches to Design and Development

- Structured Programming
  - Goes even further, breaking the program into small modules (Top-down design)
- Variables
  - Named memory locations that are defined for a program
  - Used to store the current value of data items used in the program

Approaches to Design and Development



Modules are arranged hierarchically in a top-down fashion, as illustrated here for a payroll application.

The control program calls each module as needed, such as when it is time to compute the deductions.

Each module then calls further modules, as needed, such as to compute federal taxes.

FIGURE 13-1

Structured programming. A structured program is divided into individual modules; each module represents a very specific processing task.

Approaches to Design and Development

- Object-Oriented Programming (OOP)
  - Programs consist of a collection of objects that contain data and methods to be used with that data
    - Class
      - Group of objects that share some common properties
    - Instance
      - An individual object in a class
    - Attributes
      - Data about the state of an object

Approaches to Design and Development

- Methods
  - Perform actions on an object
- Objects can perform nontraditional actions and be easily used by more than one program

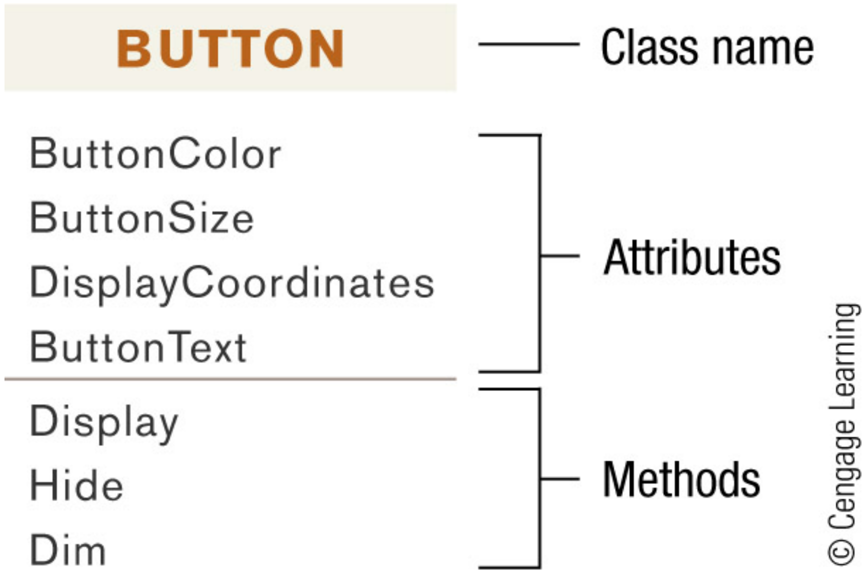


FIGURE 13-2

Button objects. This class diagram illustrates that each object in the Button class has four attributes to hold data about the current state of the button and three methods to react to messages the object receives.

Approaches to Design and Development

- Aspect-Oriented Programming (AOP)
  - Separates functions so program components can be developed and modified individually from one another
  - The components can be easily reused with separate nonrelated objects
- Adaptive Software Development
  - Designed to make program development faster and more efficient and focuses on adapting the program as it is being written
    - Iterative and/or incremental

Approaches to Design and Development

- Agile software development
  - Goal is to create software rapidly
  - Focuses on building small functional program pieces during the project
  - Includes earlier adaptive software approaches such as RAD (rapid application development) and extreme programming (XP)

The Program Development Life Cycle (PDLC)

- Program Development (application software development)
  - The process of creating application programs
- Program Development Life Cycle (PDLC)
  - The five phases of program development

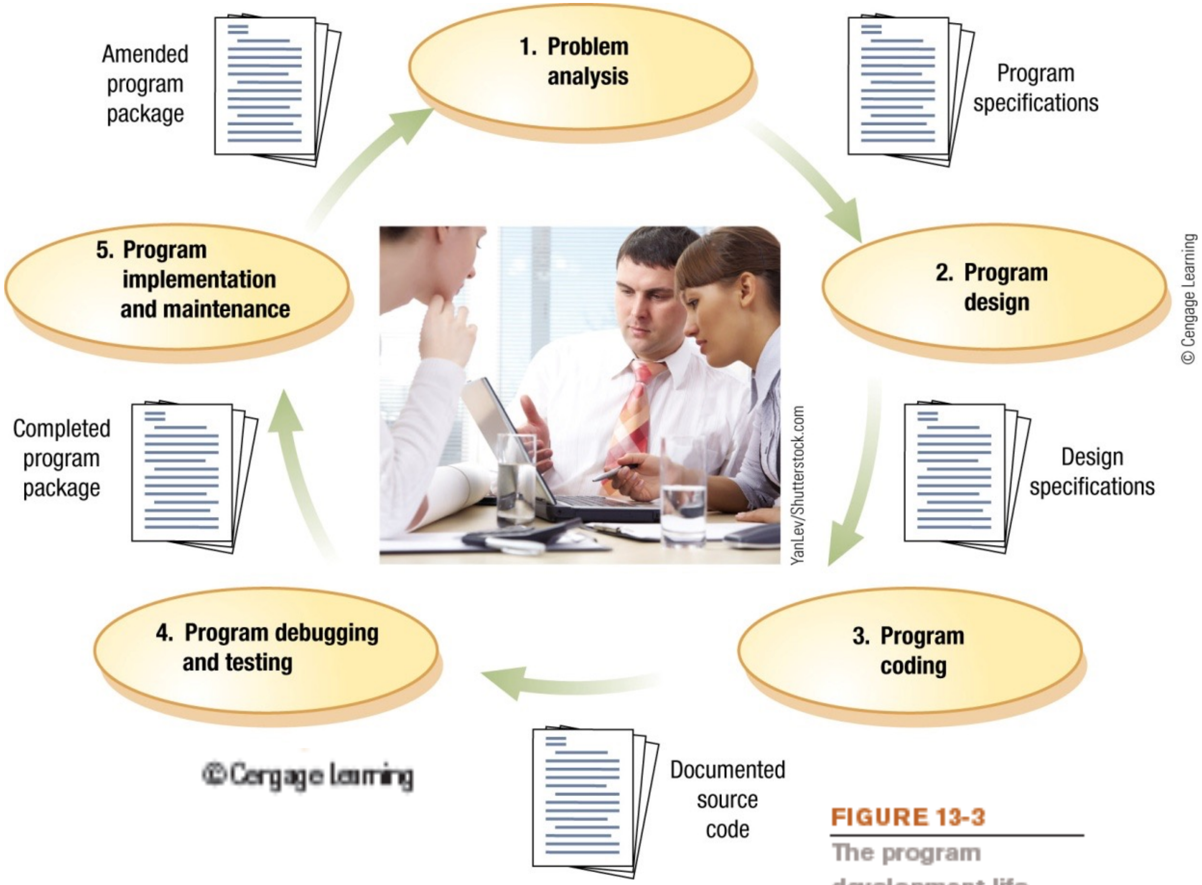


FIGURE 13-3

The program development life cycle (PDLC). Each phase of the program development life cycle produces some type of documentation to pass on to the next phase.

The Program Development Life Cycle (PDLC)

- Problem Analysis
  - The problem is considered and the program specifications are developed
    - Specifications developed during the PDLC are reviewed by the systems analyst and the programmer (the person who will code the program)
    - Goal is to understand the functions the software must perform
  - Documentation: Program Specifications
    - Result of the first phase of the PDLC outlining what the program must do

The Program Development Life Cycle (PDLC)

- Program Design
  - The program specifications are expanded into a complete design of the new program
    - Algorithm for the program is developed
    - Careful planning and design of a computer program are extremely important
  - Program Design Tools
    - Planning tools that include diagrams, charts, tables, and models
    - Structure Charts (hierarchy charts)
      - Depict the overall organization of a program

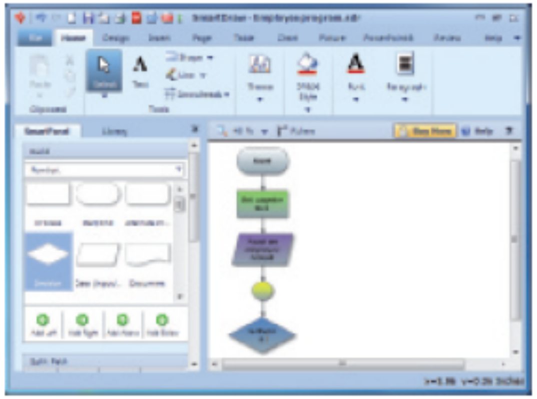
The Program Development Life Cycle (PDLC)

- Flowcharts
  - Show graphically, step-by-step, how a computer program will process data
  - Use special symbols and relational operators
  - Can be drawn by hand or with flowcharting software
- Pseudocode
  - Uses English-like statements to outline the logic of a program rather than the flowchart’s graphical symbols

The Program Development Life Cycle (PDLC)

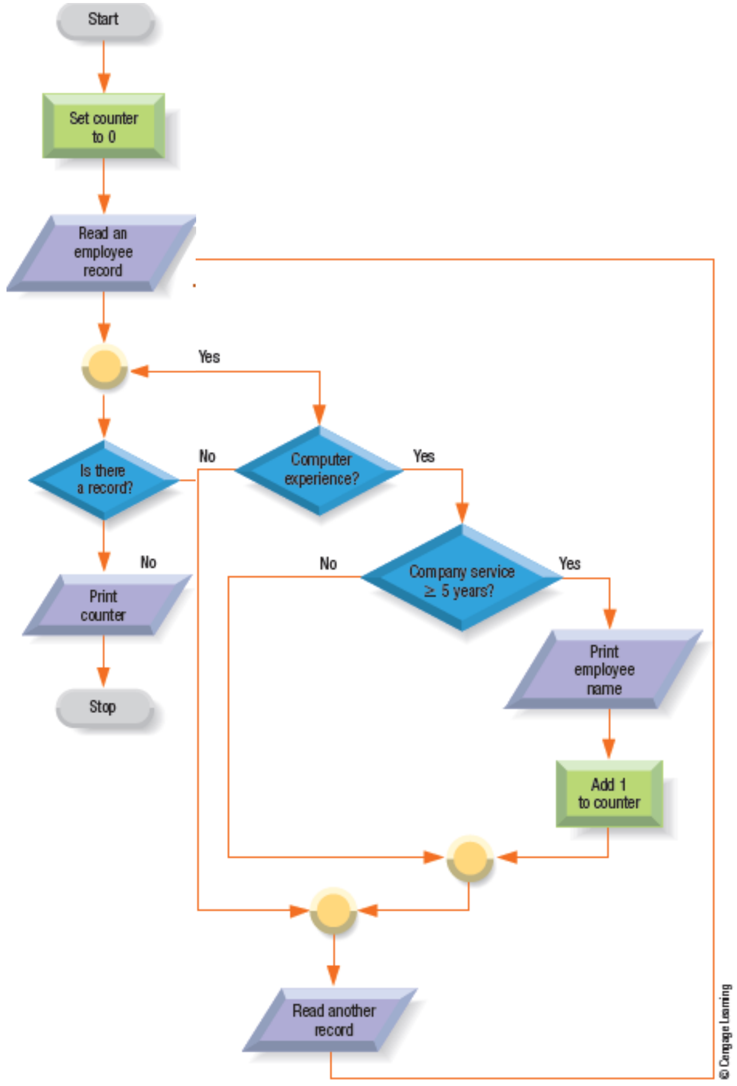
FLOWCHART OPERATORS

<		Less than
<=	≤	Less than or equal to
>		Greater than
>=	≥	Greater than or equal to
=	==	Equal to
≠	<>	Not equal to



Dan Hoffman: dhoffman@smartdraw.com

FLOWCHART SOFTWARE
Can be used to create and modify flowcharts.



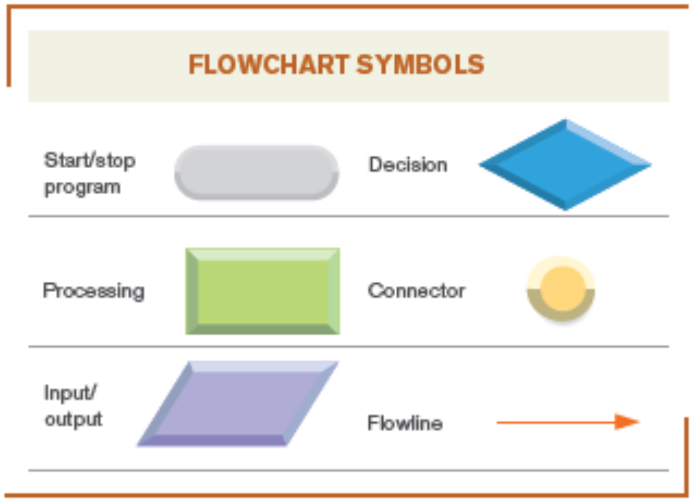
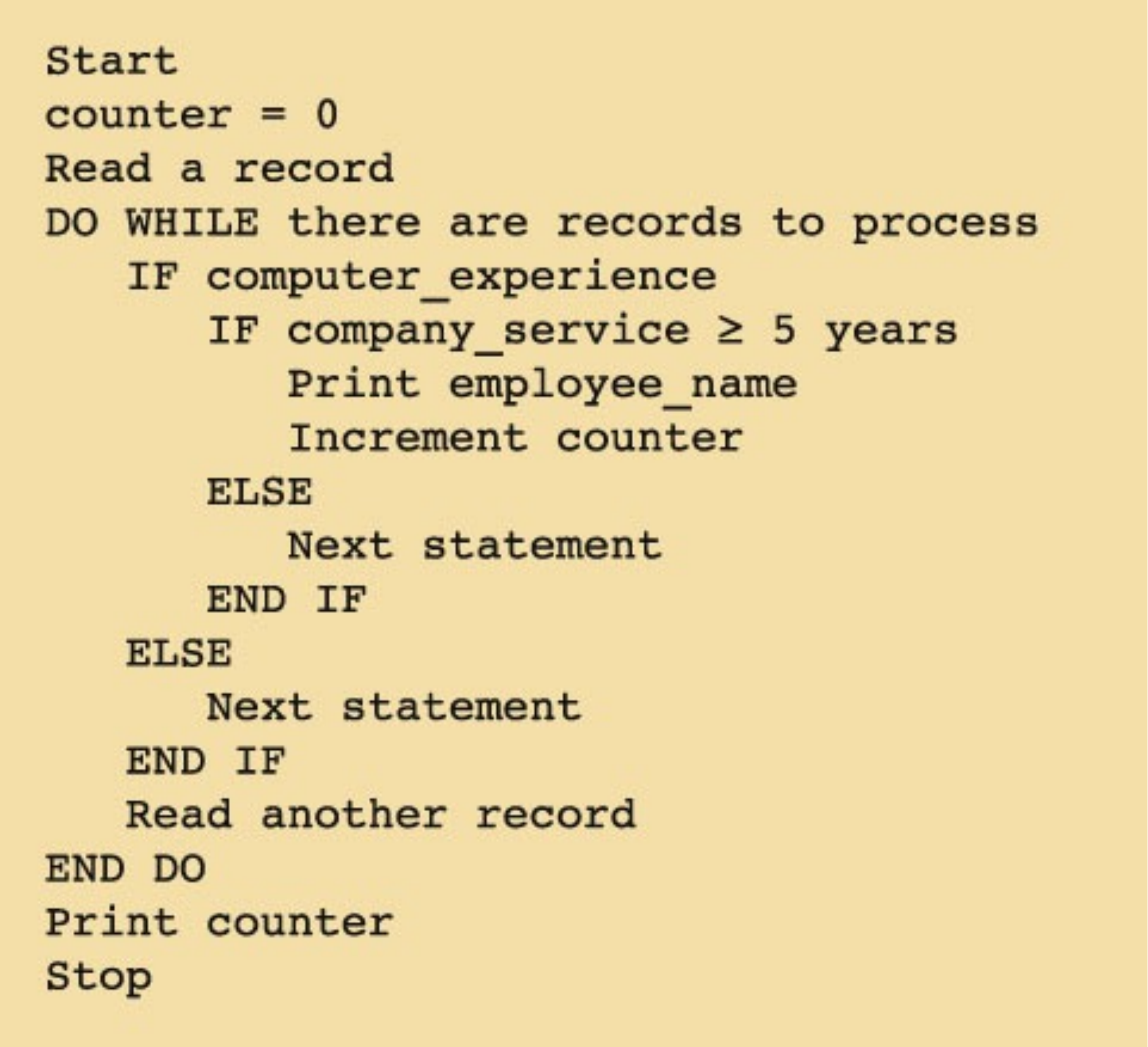


FIGURE 13-4
A flowchart example.

The Program Development Life Cycle (PDLC)



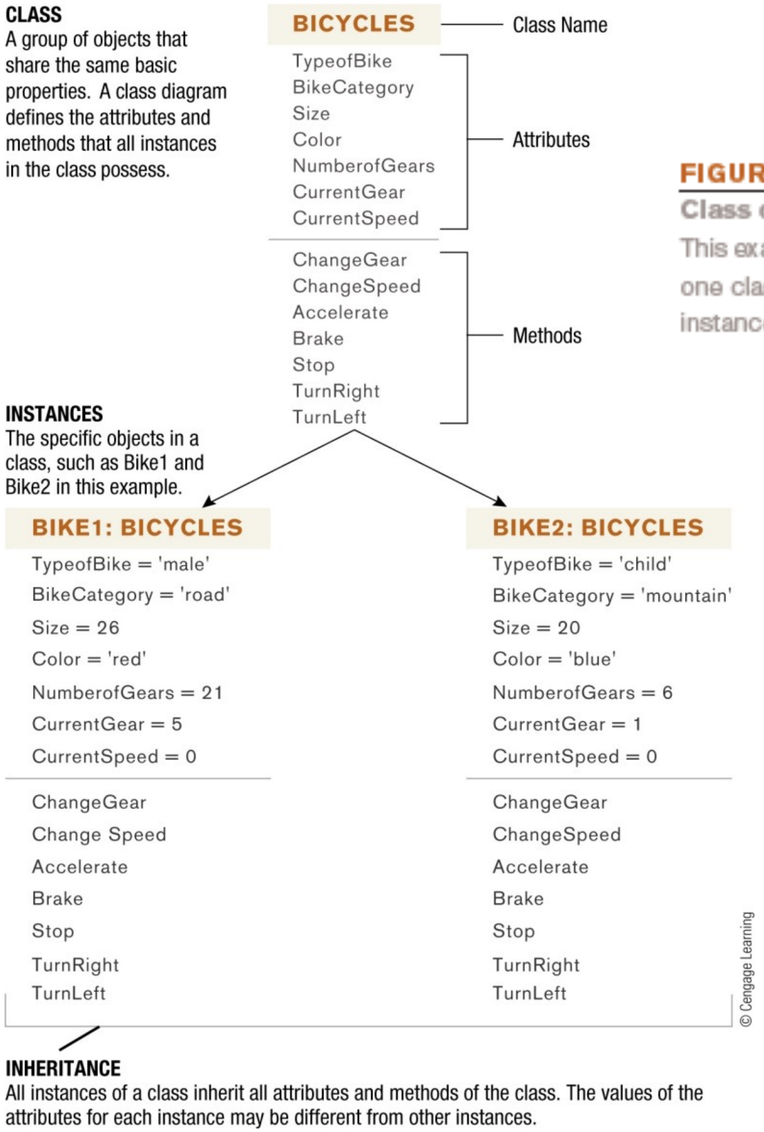
```
Start
counter = 0
Read a record
DO WHILE there are records to process
   IF computer_experience
      IF company_service ≥ 5 years
         Print employee_name
         Increment counter
      ELSE
         Next statement
      END IF
   ELSE
      Next statement
   END IF
   Read another record
END DO
Print counter
Stop
```

FIGURE 13-5

Pseudocode. The problem is the same as illustrated in the flowchart in Figure 13-4.

The Program Development Life Cycle (PDLC)

- Unified Modeling Language (UML) Models
  - Set of standard notations for creating business models
  - Widely used in object-oriented programs
  - Includes class diagrams and case diagrams



CLASS
A group of objects that share the same basic properties. A class diagram defines the attributes and methods that all instances in the class possess.

INSTANCES
The specific objects in a class, such as Bike1 and Bike2 in this example.

INHERITANCE
All instances of a class inherit all attributes and methods of the class. The values of the attributes for each instance may be different from other instances.

© Cengage Learning

FIGURE 13-6
Class diagrams.
This example shows one class and two instances of that class.

The Program Development Life Cycle (PDLC)

- Control Structures
  - A pattern for controlling the flow of logic in a computer program, module, or method
  - The Sequence Control Structure
    - Series of statements that follow one another

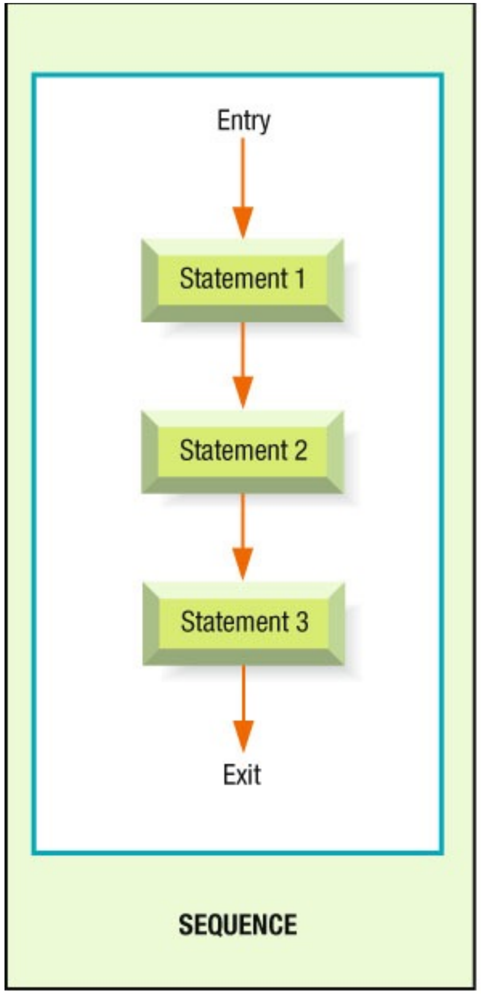


FIGURE 13-7

The Program Development Life Cycle (PDLC)

- The Selection Control Structure
  - Multiple paths, direction depends on result of a certain condition
    - » If-then-else
  - Case control structure
    - » Allows for as many possible results of the specified condition as needed
- Repetition Control Structure (iteration control structure)
  - Repeat series of steps
    - » Do-while
    - » Do-until

The Program Development Life Cycle (PDLC)

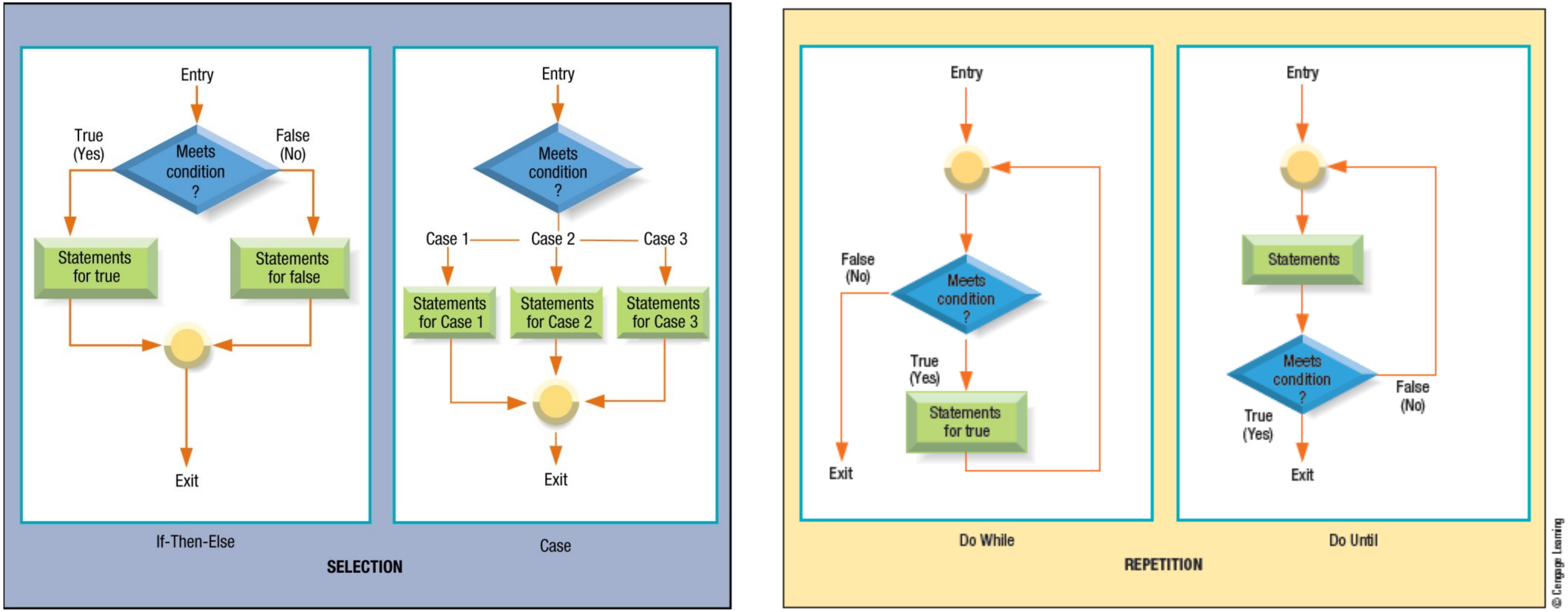


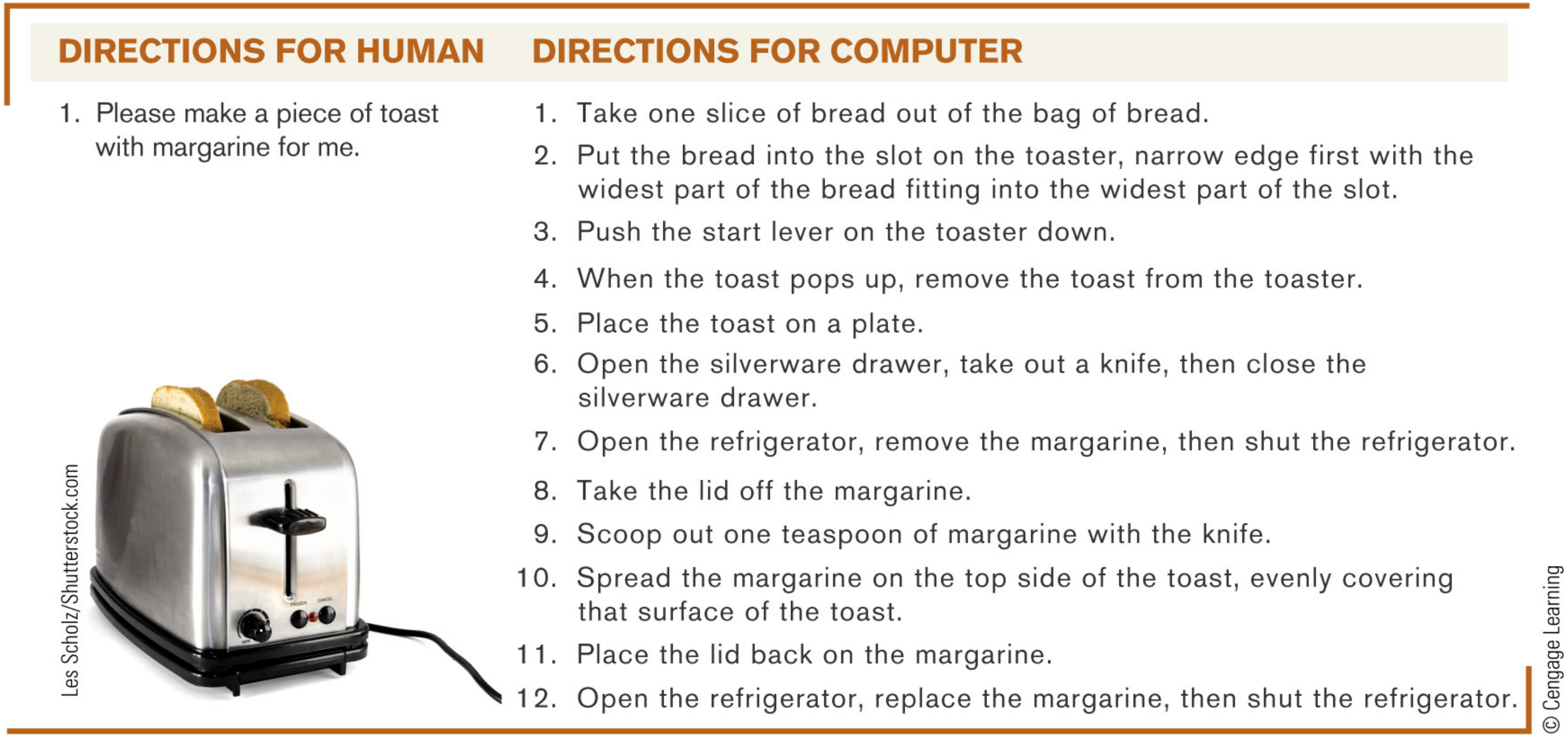
FIGURE 13-7

The three fundamental control structures. Note that each structure has only one entry point and only one exit point.

The Program Development Life Cycle (PDLC)

- Good Program Design
  - Is essential
  - Saves time
  - Good Program Design Principles
    - Be Specific
      - » All things the program must do or consider must be specified
    - Follow the One-Entry-Point/One-Exit-Point Rule
    - No Infinite Loops or Logic Errors
      - » Infinite loop is a series of steps that repeat forever

The Program Development Life Cycle (PDLC)



DIRECTIONS FOR HUMAN

1. Please make a piece of toast with margarine for me.

DIRECTIONS FOR COMPUTER

1. Take one slice of bread out of the bag of bread.
2. Put the bread into the slot on the toaster, narrow edge first with the widest part of the bread fitting into the widest part of the slot.
3. Push the start lever on the toaster down.
4. When the toast pops up, remove the toast from the toaster.
5. Place the toast on a plate.
6. Open the silverware drawer, take out a knife, then close the silverware drawer.
7. Open the refrigerator, remove the margarine, then shut the refrigerator.
8. Take the lid off the margarine.
9. Scoop out one teaspoon of margarine with the knife.
10. Spread the margarine on the top side of the toast, evenly covering that surface of the toast.
11. Place the lid back on the margarine.
12. Open the refrigerator, replace the margarine, then shut the refrigerator.

Les Scholz/Shutterstock.com

© Cengage Learning

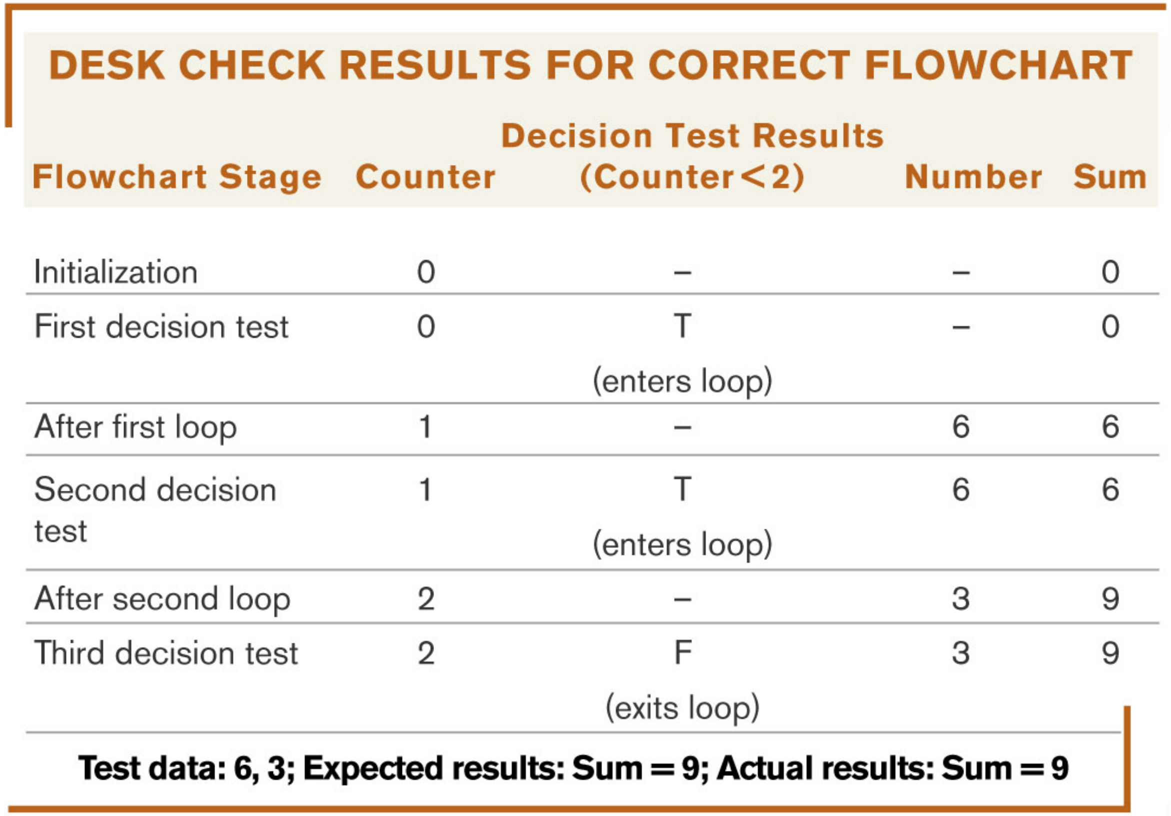
FIGURE 13-8

Writing instructions for a computer versus a person. A computer requires step-by-step instructions.

The Program Development Life Cycle (PDLC)

- Program Design Testing
  - Design should be tested to ensure logic is correct
    - Desk check
    - Tracing tables
- Documentation: Design Specifications
  - Illustrates the program needed to fulfill the program requirements
  - Expressed using structure charts, flowcharts, pseudocode, and UML models
  - Include any test data and results from desk checking

The Program Development Life Cycle (PDLC)

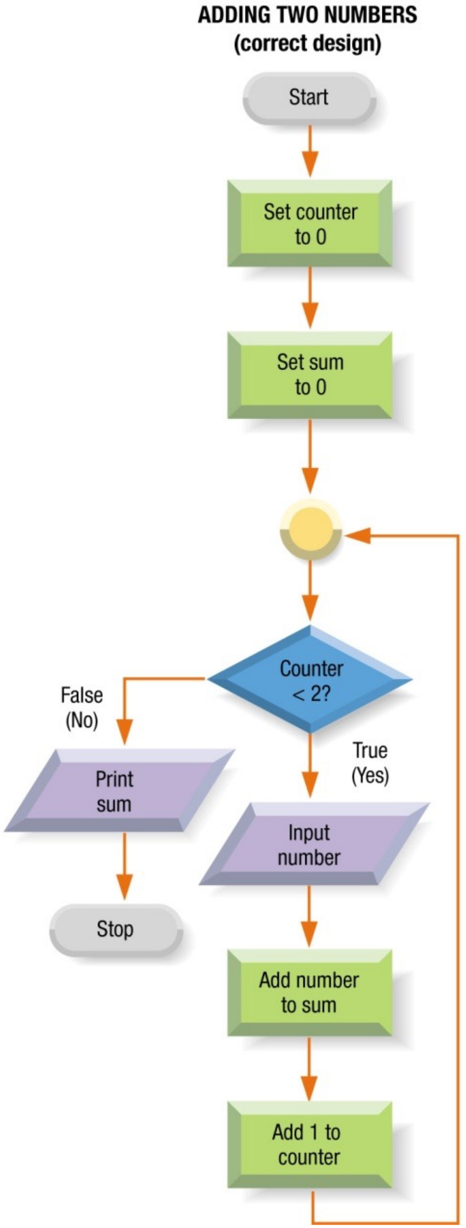


DESK CHECK RESULTS FOR CORRECT FLOWCHART

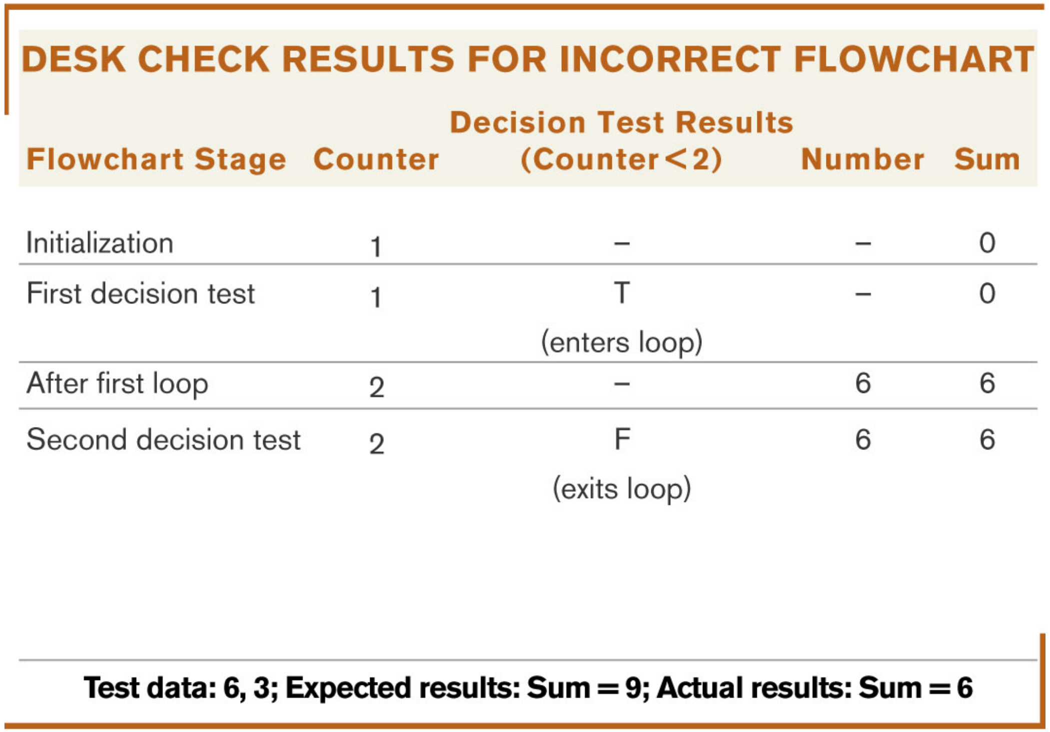
Flowchart Stage	Counter	Decision Test Results (Counter<2)	Number	Sum
Initialization	0	–	–	0
First decision test	0	T (enters loop)	–	0
After first loop	1	–	6	6
Second decision test	1	T (enters loop)	6	6
After second loop	2	–	3	9
Third decision test	2	F (exits loop)	3	9

Test data: 6, 3; Expected results: Sum = 9; Actual results: Sum = 9

FIGURE 13-9 Desk checking a flowchart.



The Program Development Life Cycle (PDLC)



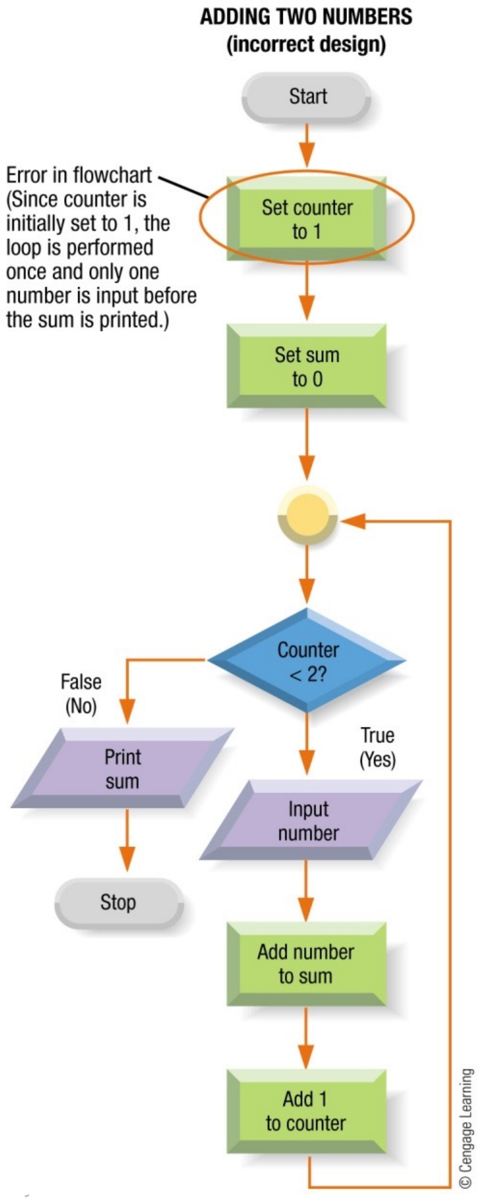
DESK CHECK RESULTS FOR INCORRECT FLOWCHART

Flowchart Stage	Counter	Decision Test Results (Counter<2)	Number	Sum
Initialization	1	–	–	0
First decision test	1	T (enters loop)	–	0
After first loop	2	–	6	6
Second decision test	2	F (exits loop)	6	6

Test data: 6, 3; Expected results: Sum = 9; Actual results: Sum = 6

FIGURE 13-9

Desk checking a flowchart.



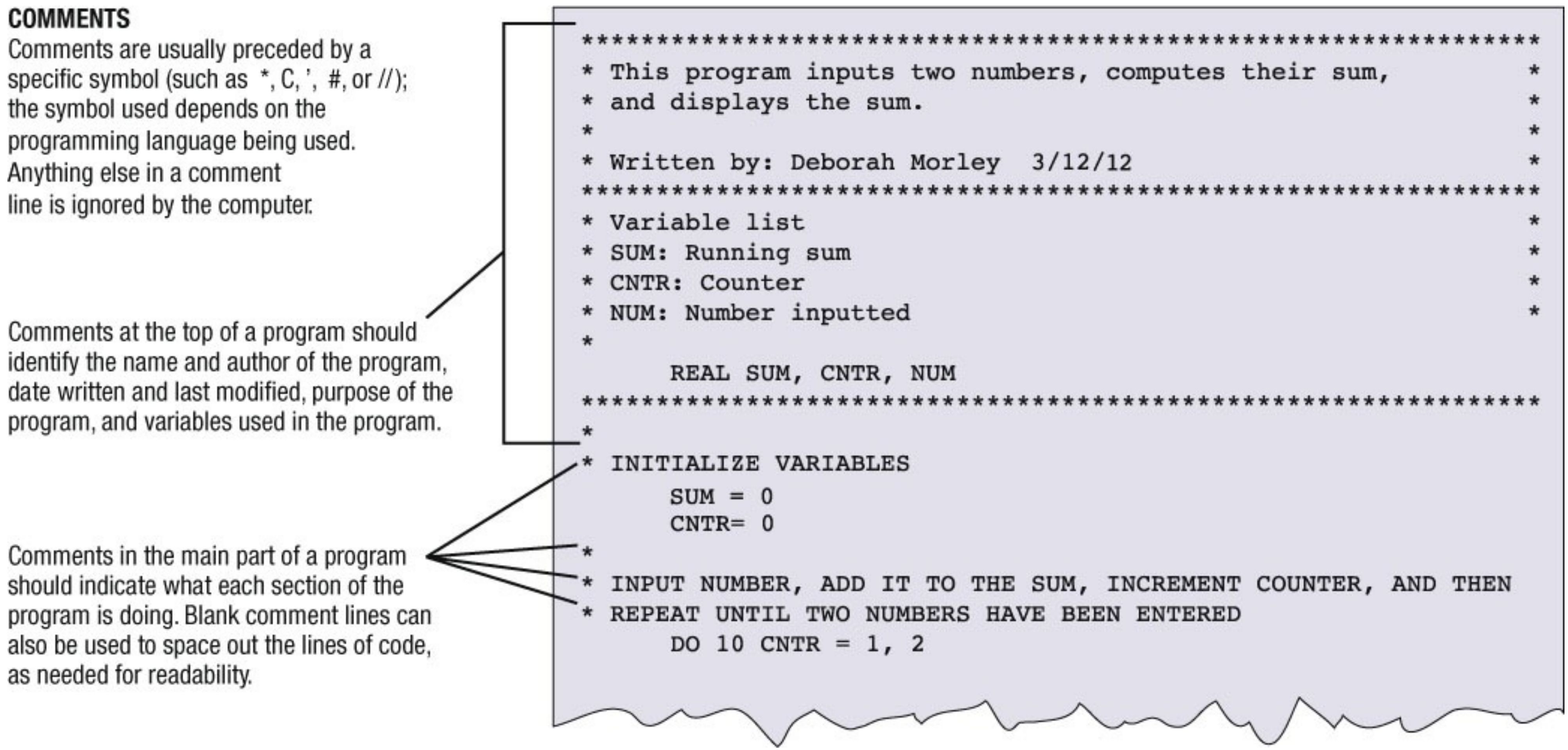
The Program Development Life Cycle (PDLC)

- Program Coding
  - The program code is written using a programming language
  - Choosing a Programming Language
    - Suitability to the application
    - Integration with other programs
    - Standards for the company
    - Programmer availability
    - Portability if being run on multiple platforms
    - Development speed

The Program Development Life Cycle (PDLC)

- The Coding Process
  - Coding Standards
    - Rules designed to standardize programming
    - Makes programs more readable and easier to maintain
    - Includes the proper use of comments to:
      - Identify the programmer and last modification date
      - Explain variables used in the program
      - Identify the main parts of the program

The Program Development Life Cycle (PDLC)



COMMENTS

Comments are usually preceded by a specific symbol (such as *, C, ', #, or //); the symbol used depends on the programming language being used. Anything else in a comment line is ignored by the computer.

Comments at the top of a program should identify the name and author of the program, date written and last modified, purpose of the program, and variables used in the program.

Comments in the main part of a program should indicate what each section of the program is doing. Blank comment lines can also be used to space out the lines of code, as needed for readability.

```
*****************************************************************
* This program inputs two numbers, computes their sum,          *
* and displays the sum.                                         *
*                                                               *
* Written by: Deborah Morley  3/12/12                           *
*****************************************************************
* Variable list                                                 *
* SUM: Running sum                                              *
* CNTR: Counter                                                 *
* NUM: Number inputted                                          *
*
     REAL SUM, CNTR, NUM
*****************************************************************
*
* INITIALIZE VARIABLES
     SUM = 0
     CNTR= 0
*
* INPUT NUMBER, ADD IT TO THE SUM, INCREMENT COUNTER, AND THEN
* REPEAT UNTIL TWO NUMBERS HAVE BEEN ENTERED
     DO 10 CNTR = 1, 2
```

The Program Development Life Cycle (PDLC)

- Reusable code
  - Pretested, error-free code segments that can be used over and over again with minor modifications
  - Can greatly reduce development time
- Documentation: Documented Source Code
  - Program coding phase results in the program written in the desired programming language
  - Should include enough comments (internal documentation) so that the source code is easy to understand and update

The Program Development Life Cycle (PDLC)

- Program Debugging and Testing
  - The process of ensuring a program is free of errors (bugs) and works as it is supposed to
  - Translating Coded Programs into Executable Code
    - Coded programs need to be translated from source code written by the programmer to object code the computer can execute
    - Converted using a language translator
      - Program that converts source code to machine language

The Program Development Life Cycle (PDLC)

- Compilers
  - Language translator that converts an entire program into machine language before executing it
  - Designed for specific programming languages such as Java or Python
- Interpreters
  - Translates one line of code at one time
- Assemblers
  - Convert assembly language programs into machine language

The Program Development Life Cycle (PDLC)

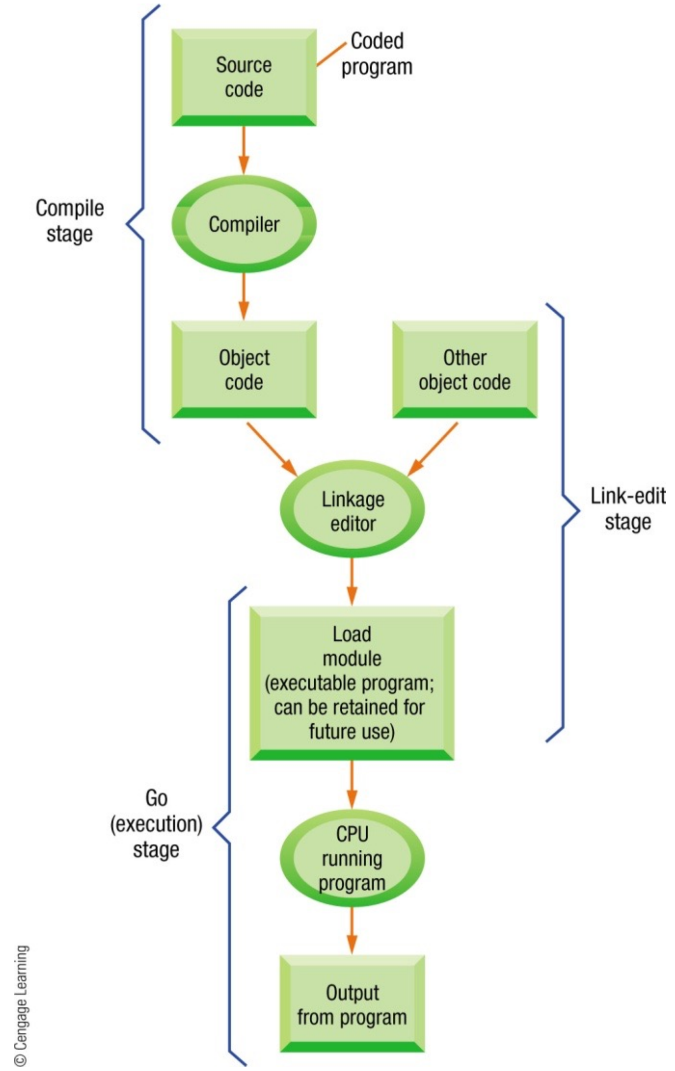


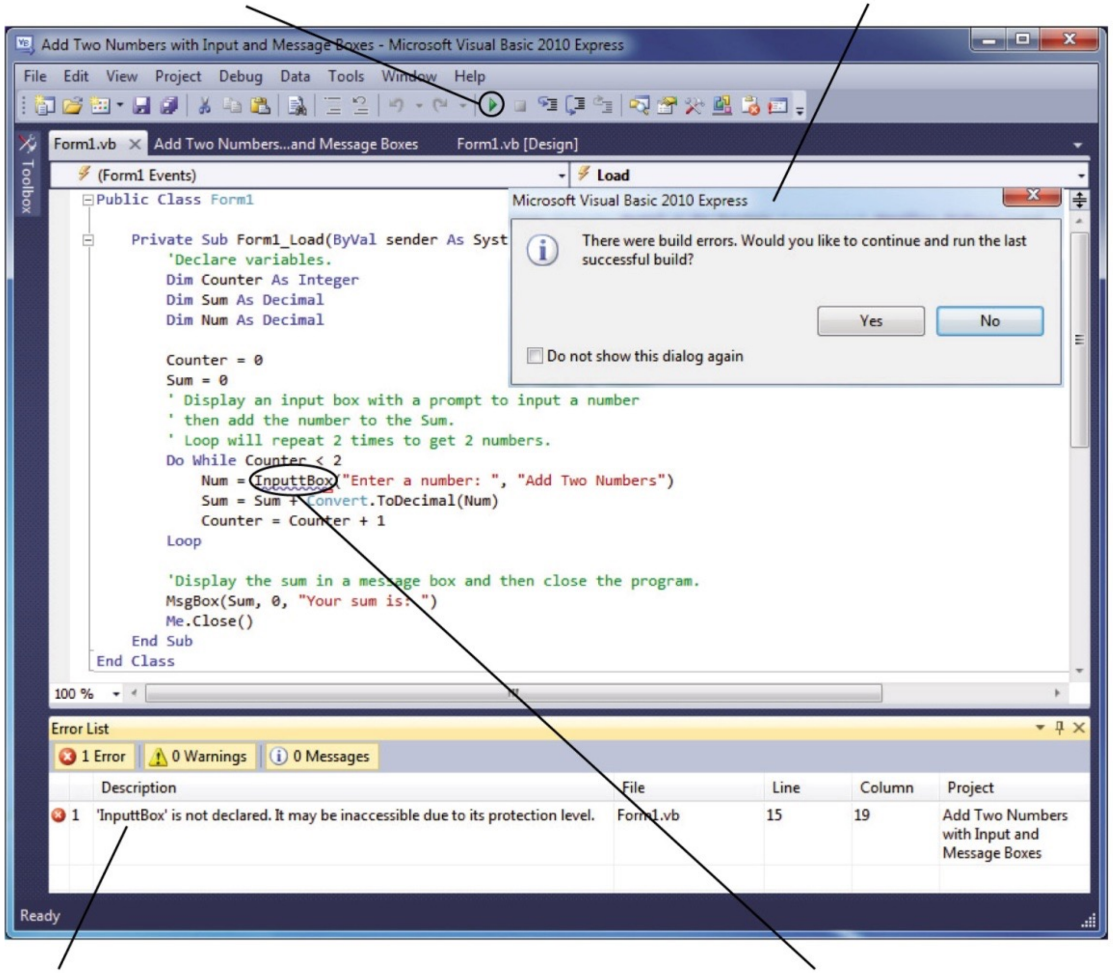
FIGURE 13-11

Compiler and linkage editor. A compiler and a linkage editor convert source code into executable code.

The Program Development Life Cycle (PDLC)

1. Clicking the Debug button starts the compilation and debugging process.

2. If a compiler error is encountered, the application typically displays an error message.



Add Two Numbers with Input and Message Boxes - Microsoft Visual Basic 2010 Express
File Edit View Project Debug Data Tools Window Help
Toolbox
Form1.vb
Add Two Numbers...and Message Boxes
Form1.vb [Design]
(Form1 Events)
Load
100 %
Ready

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As Syst
        'Declare variables.
        Dim Counter As Integer
        Dim Sum As Decimal
        Dim Num As Decimal

        Counter = 0
        Sum = 0
        ' Display an input box with a prompt to input a number
        ' then add the number to the Sum.
        ' Loop will repeat 2 times to get 2 numbers.
        Do While Counter < 2
            Num = InputtBox("Enter a number: ", "Add Two Numbers")
            Sum = Sum + Convert.ToDecimal(Num)
            Counter = Counter + 1
        Loop

        'Display the sum in a message box and then close the program.
        MsgBox(Sum, 0, "Your sum is: ")
        Me.Close()
    End Sub
End Class
```

Microsoft Visual Basic 2010 Express

There were build errors. Would you like to continue and run the last successful build?

Yes No

□ Do not show this dialog again

Error List

1 Error | 0 Warnings | 0 Messages

	Description	File	Line	Column	Project
1	'InputtBox' is not declared. It may be inaccessible due to its protection level.	Form1.vb	15	19	Add Two Numbers with Input and Message Boxes

© Cengage Learning

4. The debugger displays an error list containing all compiler errors.

3. This misspelled command is marked by a blue wavy underline.

FIGURE 13-12

Syntax errors. Syntax errors occur when the syntax (grammar rules) for a program is not followed precisely; they become obvious when compiling a program.

The Program Development Life Cycle (PDLC)

- Preliminary Debugging
  - Compiler and Syntax Errors
    - As programs are compiled or interpreted, errors occur which prevent the program from running properly
    - Syntax errors occur when the programmer has not followed the rules of the programming language
  - Run Time and Logic Errors
    - Run time errors occur when the program is running

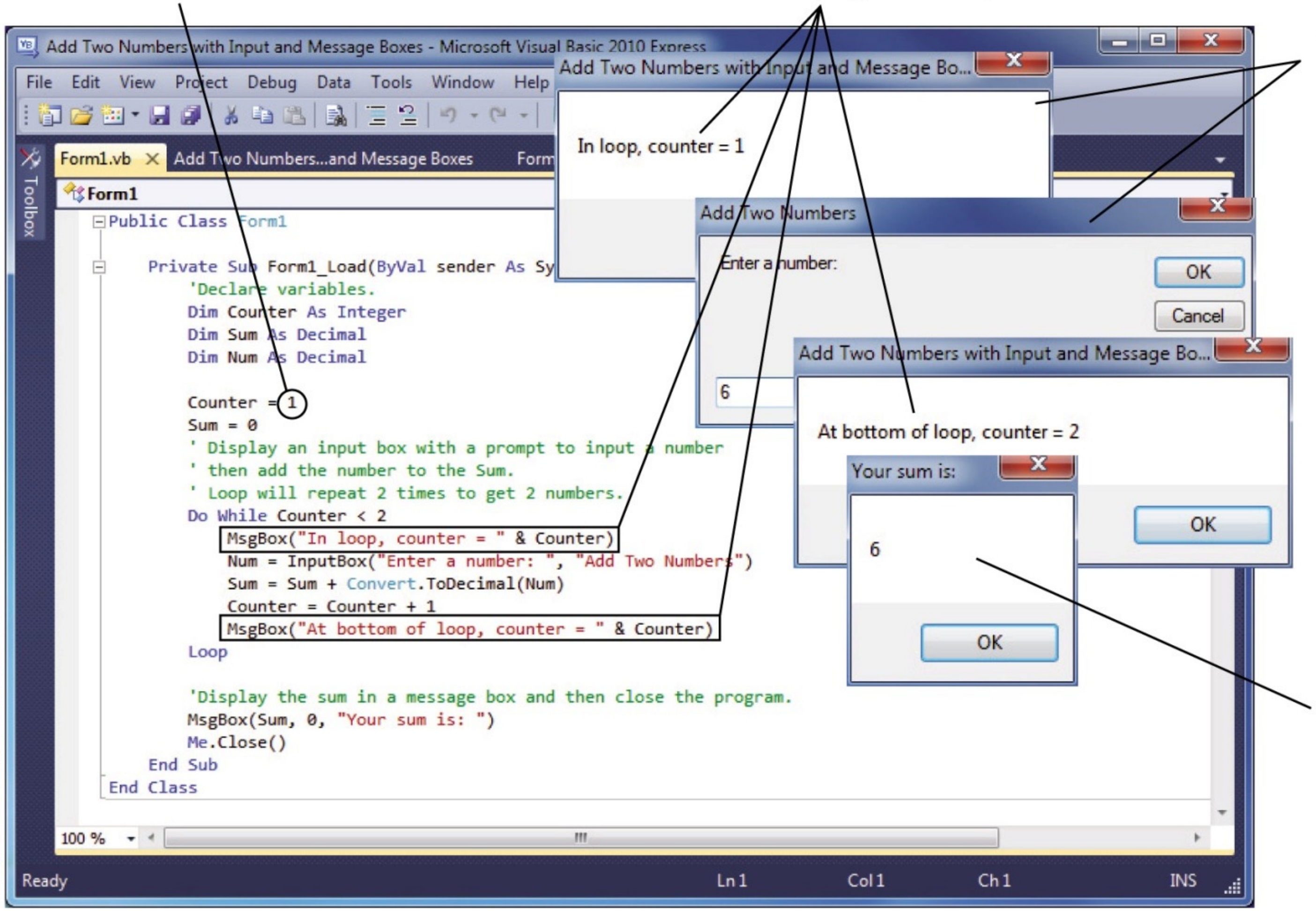
The Program Development Life Cycle (PDLC)

- Logic errors are errors in the logic of the program
  - Program will run but produces incorrect results
  - Dummy print statements can help locate logic errors and other run time errors

The Program Development Life Cycle (PDLC)

1. With logic errors, such as initializing a counter to the wrong number as shown here, the program will run but the output will be wrong.

2. Adding dummy print statements to display the values of key variables and key locations in the program can help to determine the error.


Add Two Numbers with Input and Message Boxes - Microsoft Visual Basic 2010 Express
File
Edit
View
Project
Debug
Data
Tools
Window
Help
Toolbox
Form1.vb
Add Two Numbers...and Message Boxes
Form1
100 %
Ready
Ln 1
Col 1
Ch 1
INS
Add Two Numbers with Input and Message Bo...
Add Two Numbers
OK
Cancel
Add Two Numbers with Input and Message Bo...
OK
Your sum is:
OK

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As Sy
        'Declare variables.
        Dim Counter As Integer
        Dim Sum As Decimal
        Dim Num As Decimal

        Counter = 1
        Sum = 0
        ' Display an input box with a prompt to input a number
        ' then add the number to the Sum.
        ' Loop will repeat 2 times to get 2 numbers.
        Do While Counter < 2
            MsgBox("In loop, counter = " & Counter)
            Num = InputBox("Enter a number: ", "Add Two Numbers")
            Sum = Sum + Convert.ToDecimal(Num)
            Counter = Counter + 1
            MsgBox("At bottom of loop, counter = " & Counter)
        Loop

        'Display the sum in a message box and then close the program.
        MsgBox(Sum, 0, "Your sum is: ")
        Me.Close()
    End Sub
End Class
```

```
In loop, counter = 1
```

```
Enter a number:
6
```

```
At bottom of loop, counter = 2
```

```
Your sum is:
6
```

© Cengage Learning; Courtesy of Microsoft

3. The dummy print statements, as well as the regular input and output messages belonging to the program, are displayed at the appropriate times when the program is executed.

4. The dummy print statements reveal that the loop is performed only once before the sum is displayed and help the programmer locate the counter initialization error.

FIGURE 13-13

Logic errors. Logic errors are more difficult to identify; dummy print statements can help determine the error.

The Program Development Life Cycle (PDLC)

- Testing
  - Occurs after the program appears to be correct to find any additional errors
  - Uses good test data—data that is very similar to the actual data that will be used in the program
  - Tests conditions that will occur when the program is implemented
  - Checks for coding omissions (i.e., product quantity allowed to be < 0)

The Program Development Life Cycle (PDLC)

  - Two stages
    - Alpha test—internal on-site test
    - Beta test—outside test
- Documentation: Completed Program Package
  - Copy of the test data, test results, finished program code, and other documentation generated during the testing phase should be added to the program package
    - Developer documentation
    - User documentation

The Program Development Life Cycle (PDLC)

- Program Implementation and Maintenance
  - Once the system containing the program is up and running, the implementation process is complete
  - Program maintenance
    - Process of updating software so it continues to be useful
    - Very costly
  - Documentation: Amended program package
    - Program package should be updated to reflect new problems or issues that occur and what changes to the program were necessary

Quick Quiz

1. Which approach to programming uses the concept of inheritance?
    a. Procedural
    b. Object-oriented
    c. Aspect-oriented
2. True or False: An infinite loop is an example of a logic error.
3. A(n)__________________________ is a program design tool that shows graphically step-by-step the actions a computer program will take.

Answers:

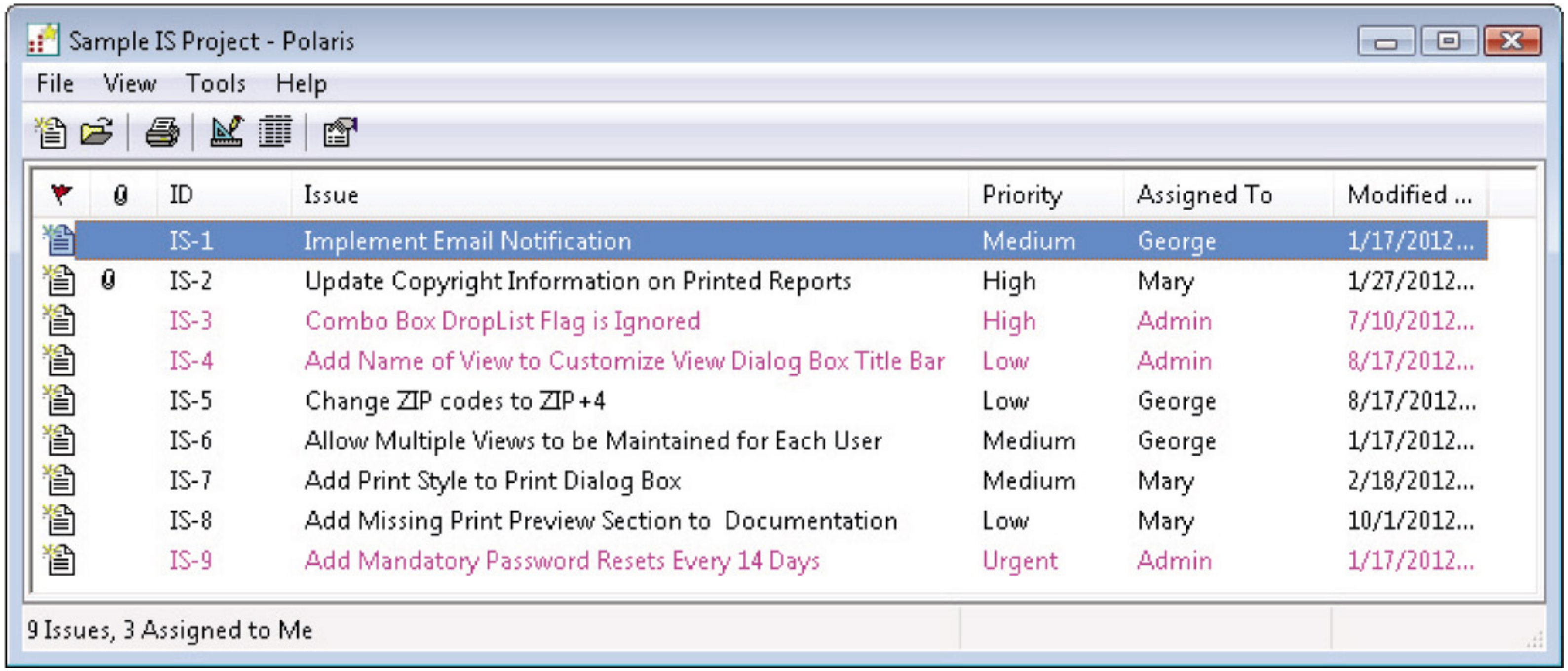
1) b; 2) True; 3) flowchart

Tools for Facilitating Program Development

- Application Lifecycle Management (ALM) Tools
  - Creating and managing an application during its entire lifecycle, from design through retirement
  - Tools include:
    - Requirements management
      - Keeping track of and managing the program requirements as they are defined and then modified
    - Configuration management
      - Keeping track of the progress of a program development project

Tools for Facilitating Program Development

- Issue tracking
  - Recording issues such as bugs or other problems that arise during development or after the system is in place



		ID	Issue	Priority	Assigned To	Modified ...
		IS-1	Implement Email Notification	Medium	George	1/17/2012...
	0	IS-2	Update Copyright Information on Printed Reports	High	Mary	1/27/2012...
		IS-3	Combo Box DropList Flag is Ignored	High	Admin	7/10/2012...
		IS-4	Add Name of View to Customize View Dialog Box Title Bar	Low	Admin	8/17/2012...
		IS-5	Change ZIP codes to ZIP+4	Low	George	8/17/2012...
		IS-6	Allow Multiple Views to be Maintained for Each User	Medium	George	1/17/2012...
		IS-7	Add Print Style to Print Dialog Box	Medium	Mary	2/18/2012...
		IS-8	Add Missing Print Preview Section to Documentation	Low	Mary	10/1/2012...
		IS-9	Add Mandatory Password Resets Every 14 Days	Urgent	Admin	1/17/2012...

9 Issues, 3 Assigned to Me

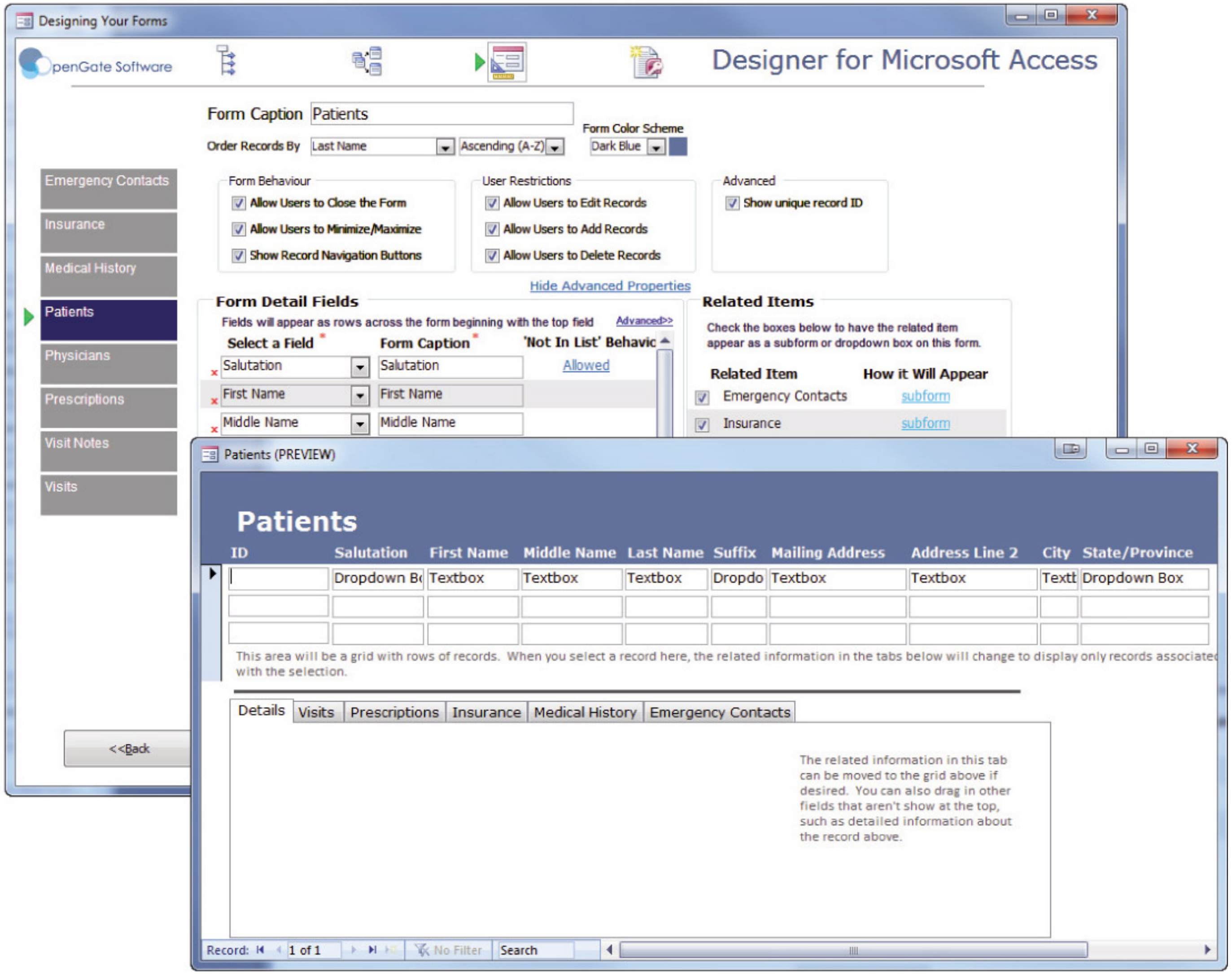
Courtesy Visible Systems

FIGURE 13-14
Issue tracking software. Allows you to track issues during the development and life of an application.

Tools for Facilitating Program Development

- Application Generators
  - Software program that helps programmers develop software
  - Macros
    - Record and play back a series of keystrokes
    - Programmers write them in a macro programming language such as Visual Basic for Applications
  - Report and Form Generators
    - Tools that enable individuals to prepare reports and
      forms quickly

Tools for Facilitating Program Development



Designing Your Forms

OpenGate Software — Designer for Microsoft Access

Form Caption: Patients

Order Records By: Last Name, Ascending (A-Z)

Form Color Scheme: Dark Blue

Form Behaviour
- ☑ Allow Users to Close the Form
- ☑ Allow Users to Minimize/Maximize
- ☑ Show Record Navigation Buttons

User Restrictions
- ☑ Allow Users to Edit Records
- ☑ Allow Users to Add Records
- ☑ Allow Users to Delete Records

Advanced
- ☑ Show unique record ID

Hide Advanced Properties

Form Detail Fields

Fields will appear as rows across the form beginning with the top field

Select a Field	Form Caption	'Not In List' Behavic
Salutation	Salutation	Allowed
First Name	First Name	
Middle Name	Middle Name	

Related Items

Check the boxes below to have the related item appear as a subform or dropdown box on this form.

Related Item	How it Will Appear
☑ Emergency Contacts	subform
☑ Insurance	subform

Patients (PREVIEW)

Patients

ID	Salutation	First Name	Middle Name	Last Name	Suffix	Mailing Address	Address Line 2	City	State/Province
	Dropdown B	Textbox	Textbox	Textbox	Dropdo	Textbox	Textbox	Textt	Dropdown Box

This area will be a grid with rows of records. When you select a record here, the related information in the tabs below will change to display only records associated with the selection.

The related information in this tab can be moved to the grid above if desired. You can also drag in other fields that aren't show at the top, such as detailed information about the record above.

Courtesy of OpenGate Software

FIGURE 13-15

Form generators. This program is used to create input screens for a database application.

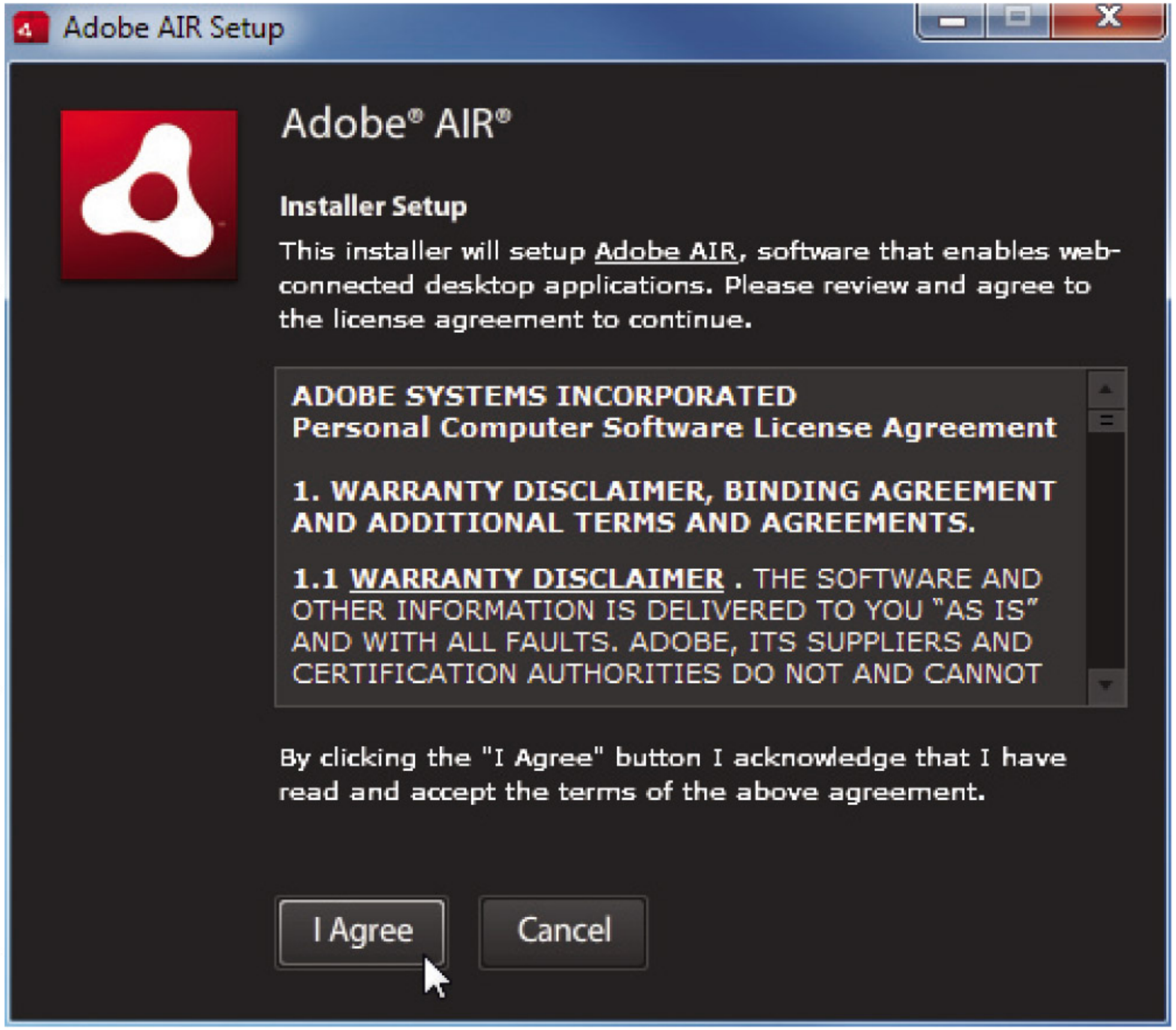
Tools for Facilitating Program Development

- Device Software Development Tools
  - Assist with developing embedded software to be used on devices, such as cars, ATM machines, and consumer devices
- Software Development Kits (SDKs) and Application Program Interfaces (APIs)
  - Designed for a particular platform
  - Enables programmers to develop applications more quickly and easily
    - Often released by hardware or software companies
      - iOS SDK—allows third party developers to create new applications for iPhone, iPad, iPod Touch

Tools for Facilitating Program Development

    - Application Program Interfaces (APIs)
        - Help applications interface with a particular operating system
    - Often used in conjunction with Web sites
- Rich Internet Application (RIA) Tools
  - Web-based applications that work like installed software programs
  - Desktop RIA can access local files and used without an Internet connection
  - Web-based RIAs are common
  - Tools to develop RIAs
    - Adobe AIR

Tools for Facilitating Program Development



Adobe® AIR®

Installer Setup

This installer will setup Adobe AIR, software that enables web-connected desktop applications. Please review and agree to the license agreement to continue.

ADOBE SYSTEMS INCORPORATED
Personal Computer Software License Agreement

1. WARRANTY DISCLAIMER, BINDING AGREEMENT AND ADDITIONAL TERMS AND AGREEMENTS.

1.1 WARRANTY DISCLAIMER . THE SOFTWARE AND OTHER INFORMATION IS DELIVERED TO YOU "AS IS" AND WITH ALL FAULTS. ADOBE, ITS SUPPLIERS AND CERTIFICATION AUTHORITIES DO NOT AND CANNOT

By clicking the "I Agree" button I acknowledge that I have read and accept the terms of the above agreement.

Adobe product screen shot(s) reprinted with permission from Adobe Systems Incorporated.

FIGURE 13-16

Adobe AIR must be installed on a computer in order to run AIR applications.

Quick Quiz

1. Which of the following is not an Application Lifecycle Management (ALM) tool?

a. Requirements definition software

b. Code generator

c. Application program interface (API)

2. True or False: A software development kit (SDK) is designed for a particular platform and allows programmers to develop applications quickly for that platform.

3. A(n) _______________________ is often used to create the forms or input screens used to input data into a program or database.

Answers:

1) c; 2) True; 3) form generator

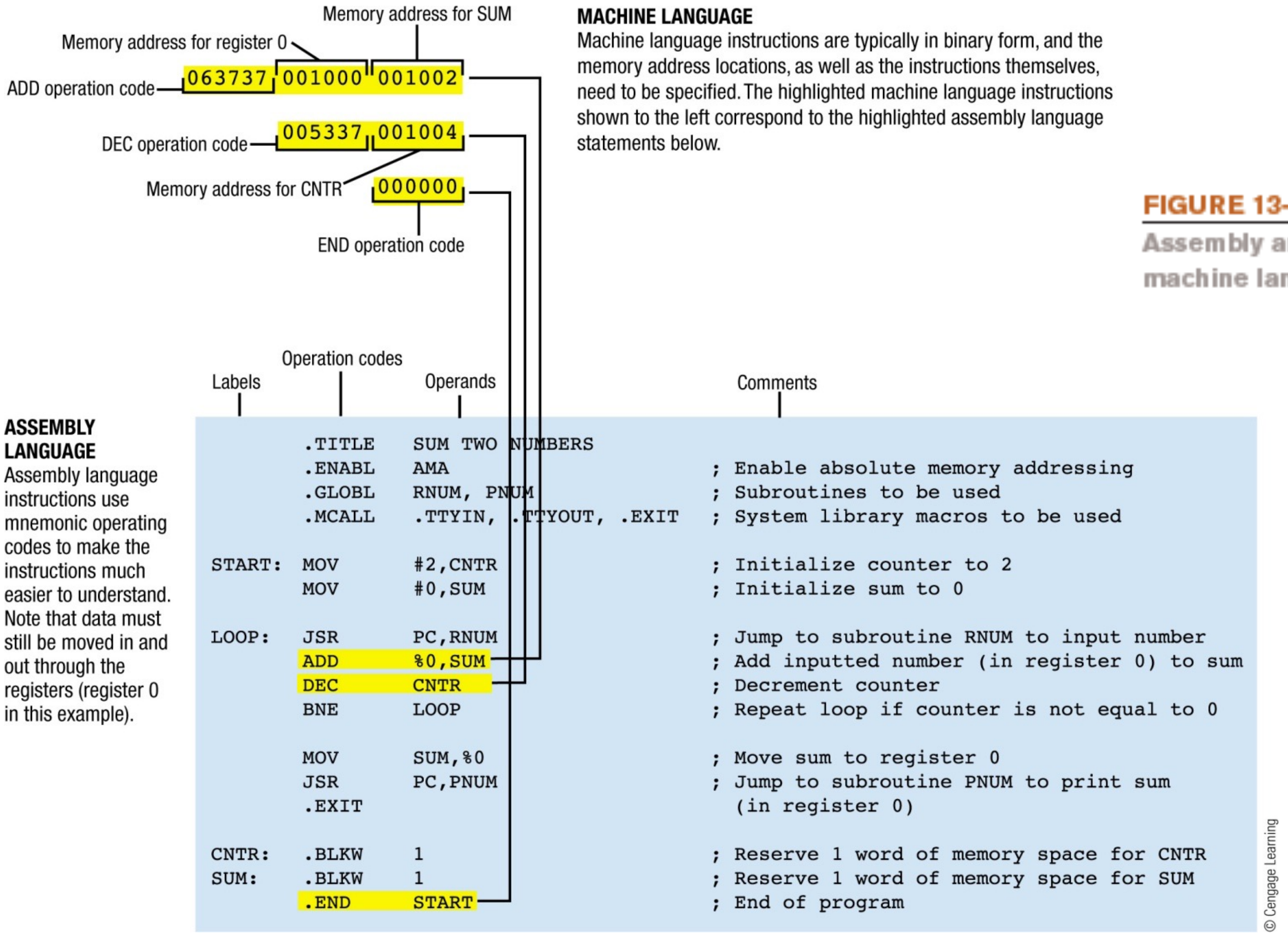
Programming Languages

- What is a Programming Language?
  - A set of rules, words, symbols, and codes used to write computer programs
  - To write a program, appropriate software for the programming language being used is needed
- Categories of Programming Languages
  - Types of programs they are designed to create
    - Procedural languages or object-oriented languages
  - How evolved the programming language is
    - Levels or generations

Programming Languages

- Low-Level Languages (earliest programming languages)
  - Machine language
    - Written at a very low level, just using 1s and 0s
    - First generation of programming languages
  - Assembly language
    - Includes some names and other symbols to replace some of the 1s and 0s in machine language
    - Second generation of programming languages
    - Machine dependent
      - » Written for one specific type of computer

Programming Languages



MACHINE LANGUAGE
Machine language instructions are typically in binary form, and the memory address locations, as well as the instructions themselves, need to be specified. The highlighted machine language instructions shown to the left correspond to the highlighted assembly language statements below.

ASSEMBLY LANGUAGE
Assembly language instructions use mnemonic operating codes to make the instructions much easier to understand. Note that data must still be moved in and out through the registers (register 0 in this example).

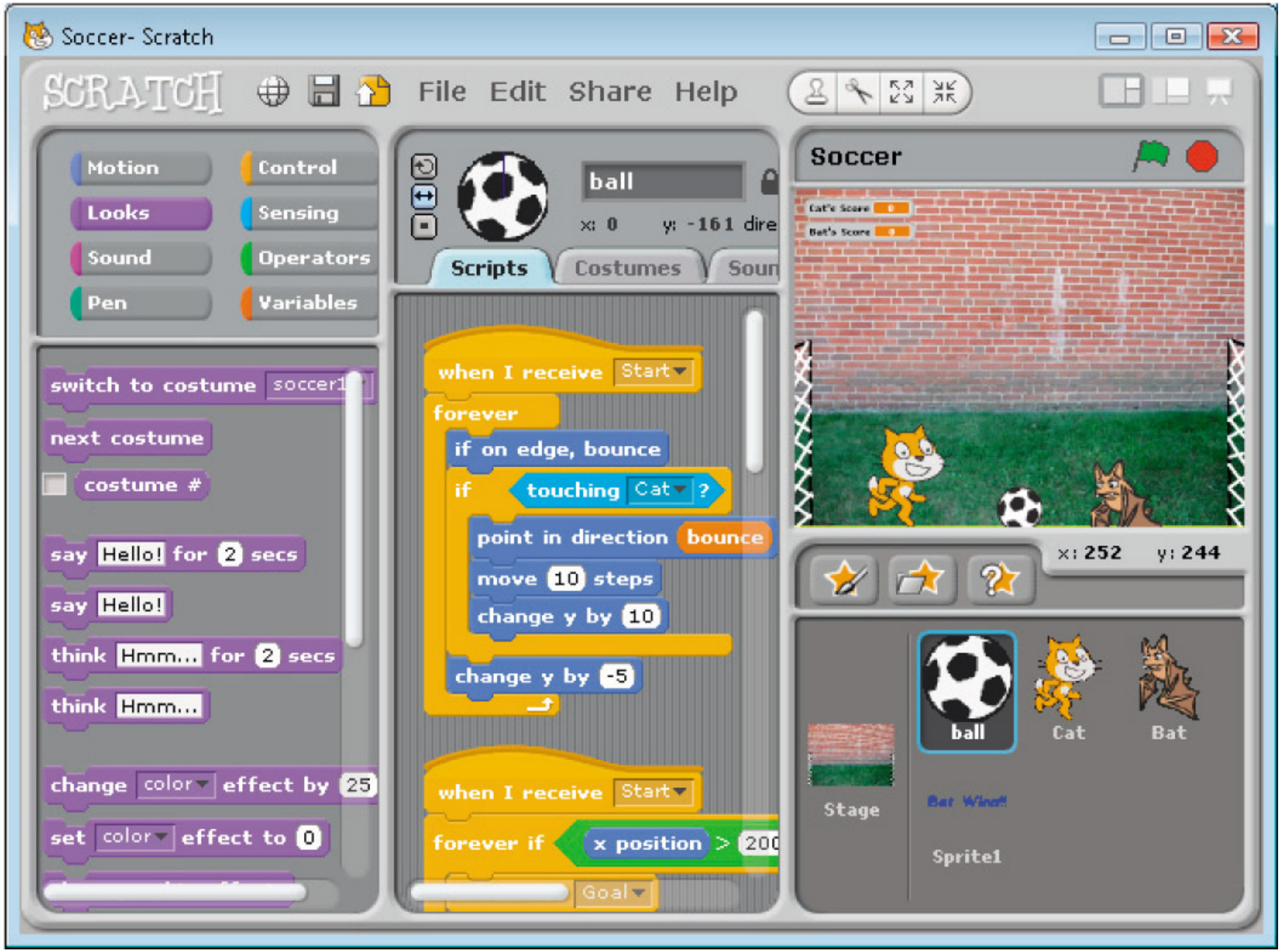
```
        .TITLE   SUM TWO NUMBERS
        .ENABL   AMA                        ; Enable absolute memory addressing
        .GLOBL   RNUM, PNUM                 ; Subroutines to be used
        .MCALL   .TTYIN, .TTYOUT, .EXIT     ; System library macros to be used

START:  MOV      #2,CNTR                    ; Initialize counter to 2
        MOV      #0,SUM                     ; Initialize sum to 0

LOOP:   JSR      PC,RNUM                    ; Jump to subroutine RNUM to input number
        ADD      %0,SUM                     ; Add inputted number (in register 0) to sum
        DEC      CNTR                       ; Decrement counter
        BNE      LOOP                       ; Repeat loop if counter is not equal to 0

        MOV      SUM,%0                     ; Move sum to register 0
        JSR      PC,PNUM                    ; Jump to subroutine PNUM to print sum
        .EXIT                                 (in register 0)

CNTR:   .BLKW    1                          ; Reserve 1 word of memory space for CNTR
SUM:    .BLKW    1                          ; Reserve 1 word of memory space for SUM
        .END     START                      ; End of program
```

FIGURE 13-17
Assembly and machine language.

Programming Languages

- High-Level Languages
  - Closer to natural languages
  - Machine independent
  - Includes 3GLs (FORTRAN, BASIC, COBOL,C, etc.) and object-oriented languages (Visual Basic, C#, Python, Java, etc.)
  - Visual or graphical languages
    - Use graphical interface to create programs
    - Designed for educational purposes

Programming Languages



```
when I receive Start
forever
  if on edge, bounce
  if touching Cat ?
    point in direction bounce
    move 10 steps
    change y by 10
  change y by -5

when I receive Start
forever if x position > 200
  Goal
```

Scratch is developed by the Lifelong Kindergarten Group at the MIT Media Lab.
See http://scratch.mit.edu

FIGURE 13-18
The Scratch graphical programming language.

Understanding Computers: Today and Tomorrow, 14th Edition

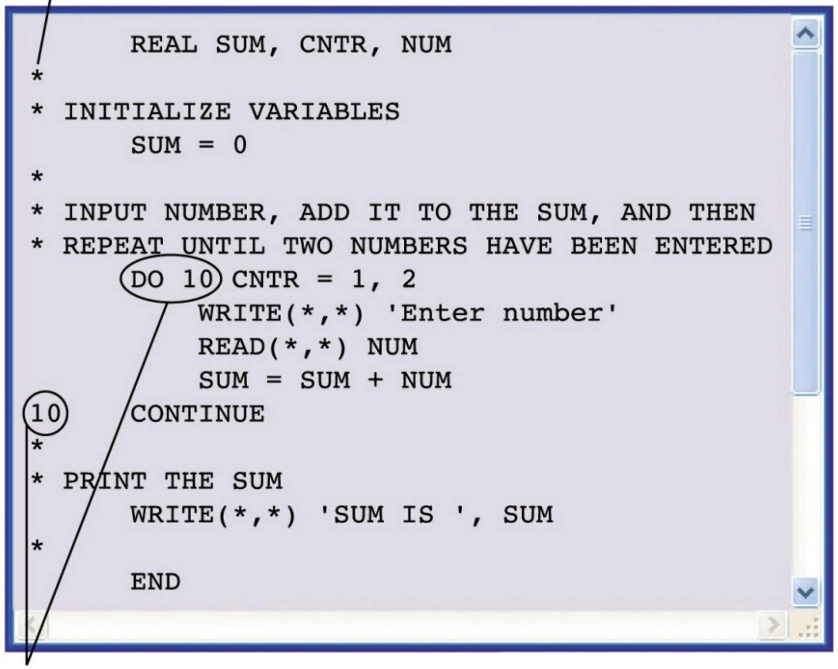
Programming Languages

- Fourth-Generation Languages (4GLs)
  - Even closer to natural languages and easier to work with than high-level
  - Declarative rather than procedural
  - Includes structured query language (SQL) used with databases
- Common Programming Languages
  - Languages not widely used today
    - Logo (teach children how to program)
    - PL/1 (business and scientific applications
    - Prolog and LISP (artificial intelligence)
    - SmallTalk (one of the first object-oriented languages)

Programming Languages

– FORTRAN
  - High-level programming language used for mathematical, scientific, and engineering applications
  - Efficient for math, engineering and scientific applications
  - Still used today for high-performance computing tasks (weather forecasting)
  - Fortress may eventually replace FORTRAN

Comments are preceded by an asterisk or a C.



```
      REAL SUM, CNTR, NUM
*
* INITIALIZE VARIABLES
      SUM = 0
*
* INPUT NUMBER, ADD IT TO THE SUM, AND THEN
* REPEAT UNTIL TWO NUMBERS HAVE BEEN ENTERED
      DO 10 CNTR = 1, 2
        WRITE(*,*) 'Enter number'
        READ(*,*) NUM
        SUM = SUM + NUM
10    CONTINUE
*
* PRINT THE SUM
      WRITE(*,*) 'SUM IS ', SUM
*
      END
```

© Cengage Learning

Program statements can be numbered in order to control loops and other types of branching.

FIGURE 13-19 The adding-two-numbers program written in FORTRAN.

Programming Languages

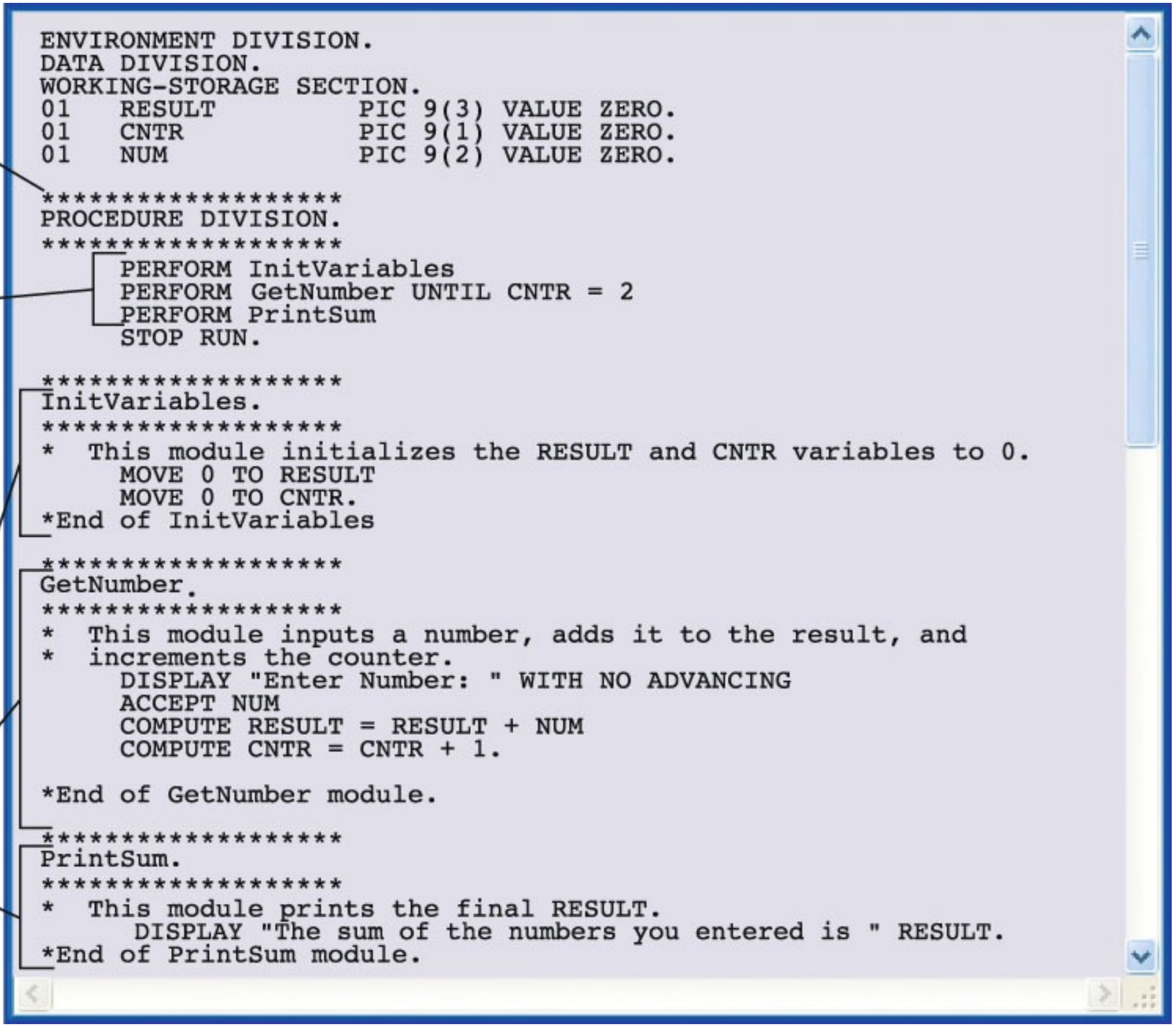
- COBOL
  - Designed for business transaction processing
  - Makes extensive use of modules
  - Strength lies in batch processing and its stability
  - Programs are lengthy and take a long time to write
  - Considered to be outdated by some
  - New versions are evolving
    - COBOL.NET

Programming Languages

Comments are preceded by an asterisk.

Most COBOL programs use a number of modules to break the program into manageable pieces. These submodules are called from the main control module using these statements.

Three submodules are used in this program.



```
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01    RESULT          PIC 9(3) VALUE ZERO.
01    CNTR            PIC 9(1) VALUE ZERO.
01    NUM             PIC 9(2) VALUE ZERO.

*******************
PROCEDURE DIVISION.
*******************
     PERFORM InitVariables
     PERFORM GetNumber UNTIL CNTR = 2
     PERFORM PrintSum
     STOP RUN.

*******************
InitVariables.
*******************
*   This module initializes the RESULT and CNTR variables to 0.
     MOVE 0 TO RESULT
     MOVE 0 TO CNTR.
*End of InitVariables

*******************
GetNumber.
*******************
*   This module inputs a number, adds it to the result, and
*   increments the counter.
     DISPLAY "Enter Number: " WITH NO ADVANCING
     ACCEPT NUM
     COMPUTE RESULT = RESULT + NUM
     COMPUTE CNTR = CNTR + 1.

*End of GetNumber module.

*******************
PrintSum.
*******************
*   This module prints the final RESULT.
      DISPLAY "The sum of the numbers you entered is " RESULT.
*End of PrintSum module.
```

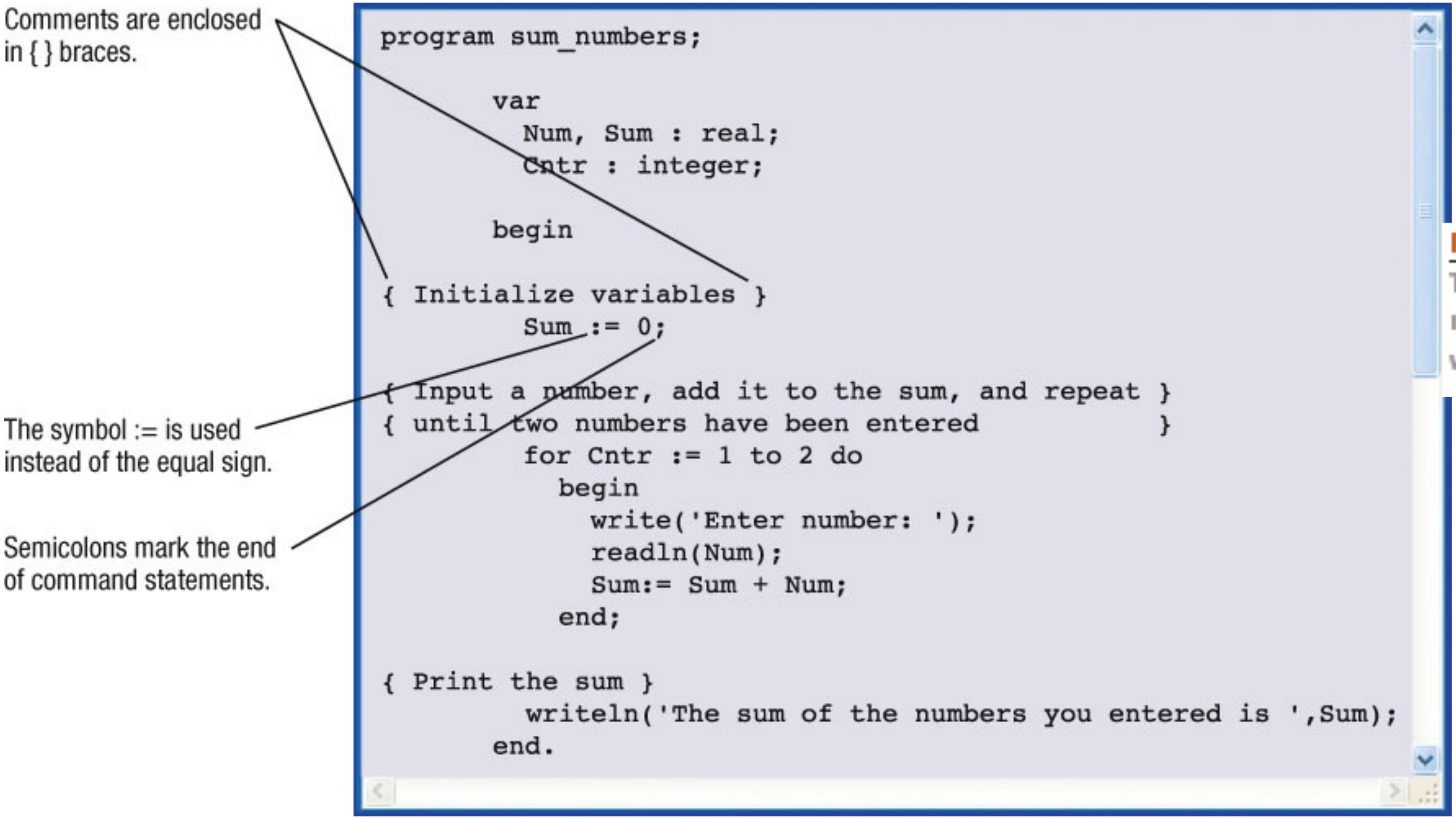
FIGURE 13-20

The adding-two-numbers program written in COBOL

Programming Languages

- Pascal
  - Named after mathematician Blaise Pascal
  - Created as a teaching tool to encourage structured programming
  - Contains a variety of control structures used to manipulate modules systematically
- BASIC and Visual Basic
  - Easy-to-learn, high-level programming language that was developed to be used by beginning programmers
  - Visual Basic
    - Object-oriented version of BASIC; uses a visual environment

Programming Languages



```
program sum_numbers;

       var
         Num, Sum : real;
         Cntr : integer;

       begin

{ Initialize variables }
         Sum := 0;

{ Input a number, add it to the sum, and repeat }
{ until two numbers have been entered           }
         for Cntr := 1 to 2 do
           begin
             write('Enter number: ');
             readln(Num);
             Sum:= Sum + Num;
           end;

{ Print the sum }
         writeln('The sum of the numbers you entered is ',Sum);
       end.
```

Comments are enclosed in { } braces.

The symbol := is used instead of the equal sign.

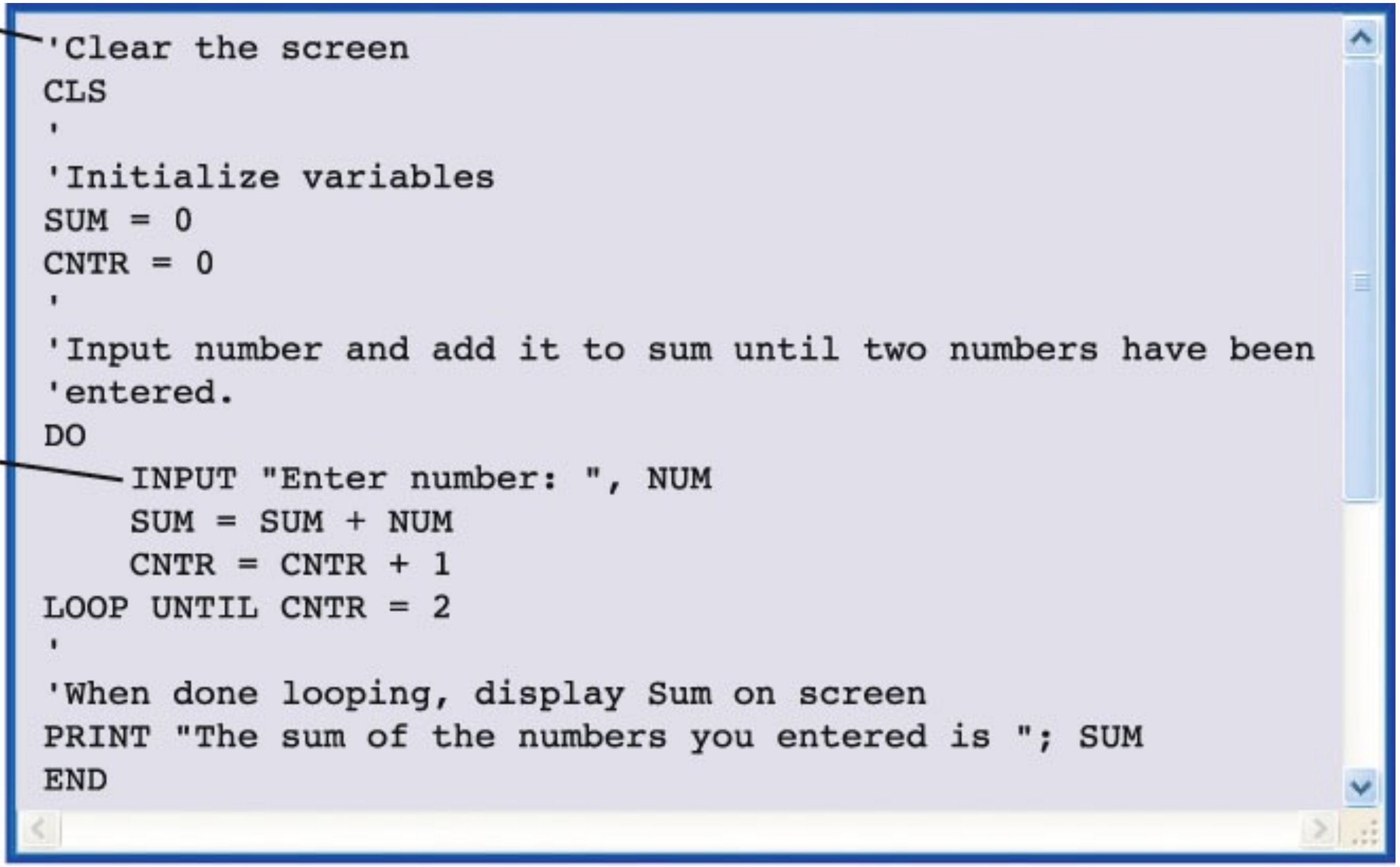
Semicolons mark the end of command statements.

FIGURE 13-21
The adding-two-numbers program written in Pascal.

Programming Languages

Comments are preceded by a single quotation mark.

Programs typically include input statements that pause the program until the user supplies the appropriate data.



```
'Clear the screen
CLS
'
'Initialize variables
SUM = 0
CNTR = 0
'
'Input number and add it to sum until two numbers have been
'entered.
DO
    INPUT "Enter number: ", NUM
    SUM = SUM + NUM
    CNTR = CNTR + 1
LOOP UNTIL CNTR = 2
'
'When done looping, display Sum on screen
PRINT "The sum of the numbers you entered is "; SUM
END
```

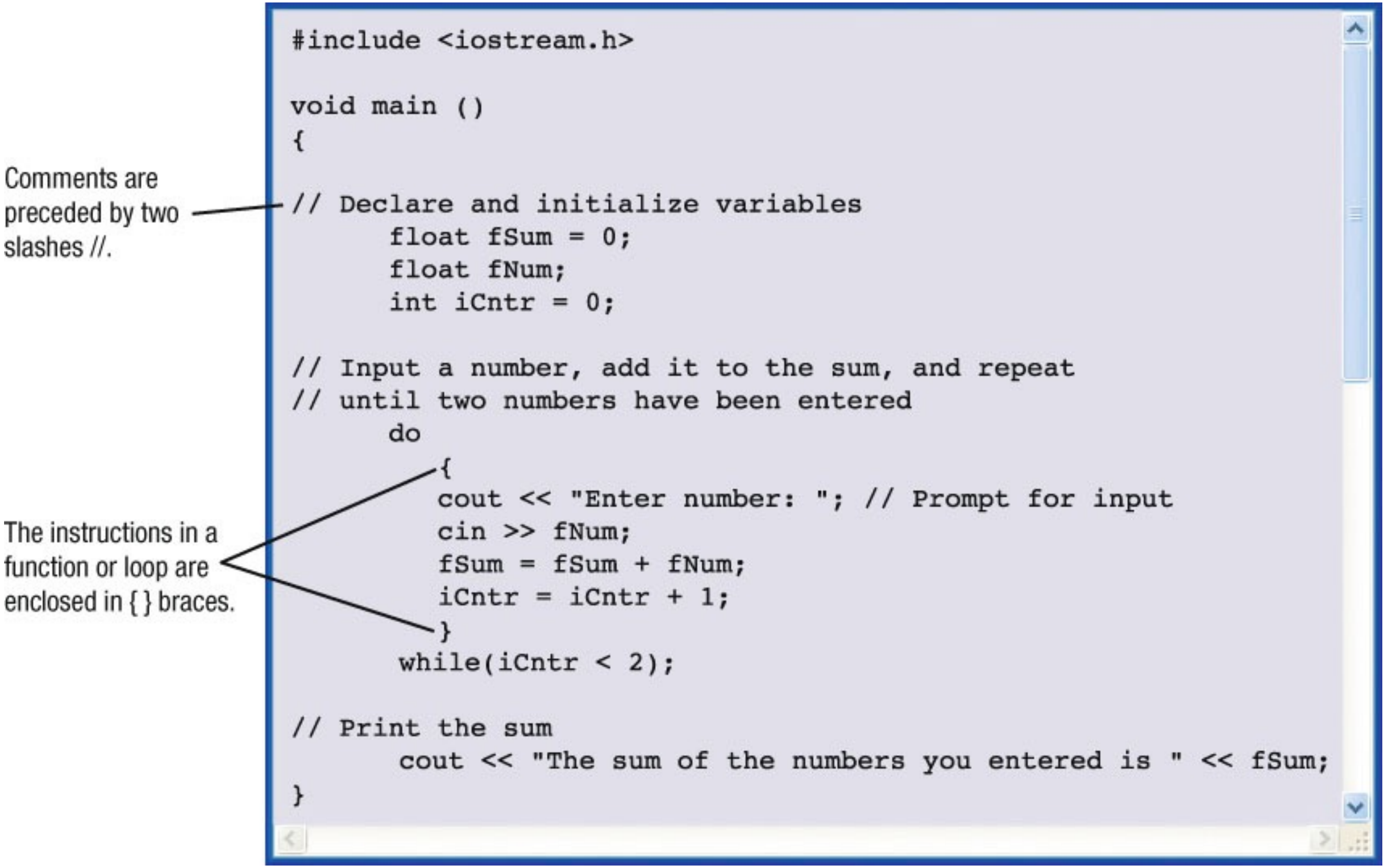
FIGURE 13-22

The adding-two-numbers program written in BASIC.

Programming Languages

- C, C++, and C#
  - C
    - Much closer to assembly language than other high-level languages
    - Designed for system programming
  - C++
    - Object-oriented versions of C
    - Very popular for graphical applications
  - C# (C sharp)
    - Used to create Web applications and XML-based Web services
  - Objective-C:
    - For iPhone and other Apple applications

Programming Languages



```
#include <iostream.h>

void main ()
{

// Declare and initialize variables
      float fSum = 0;
      float fNum;
      int iCntr = 0;

// Input a number, add it to the sum, and repeat
// until two numbers have been entered
      do
         {
         cout << "Enter number: "; // Prompt for input
         cin >> fNum;
         fSum = fSum + fNum;
         iCntr = iCntr + 1;
         }
       while(iCntr < 2);

// Print the sum
       cout << "The sum of the numbers you entered is " << fSum;
}
```

Comments are preceded by two slashes //.

The instructions in a function or loop are enclosed in { } braces.

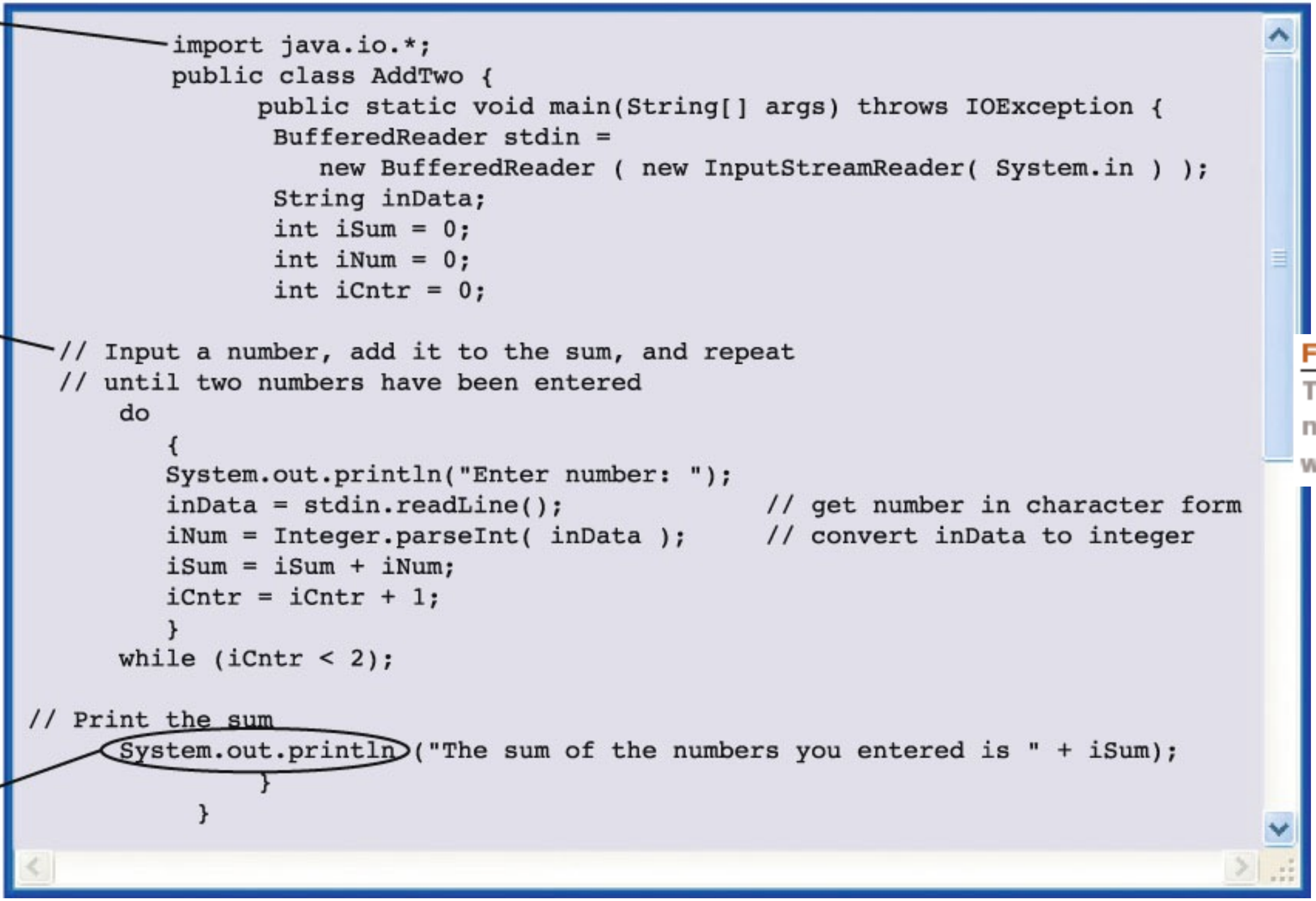
FIGURE 13-23

The adding-two-numbers program written in C++.

Programming Languages

- Java
  - High-level, object-oriented programming language frequently used for Web-based applications
  - Java programs are compiled into bytecode
  - Can run on any computer that includes Java Virtual Machine (Java VM)
  - Can be used to write Java applets
    - Scroll text on Web page, games, calculators, etc
  - Is one of the most popular programming languages today

Programming Languages



```
         import java.io.*;
         public class AddTwo {
               public static void main(String[] args) throws IOException {
                BufferedReader stdin =
                   new BufferedReader ( new InputStreamReader( System.in ) );
                String inData;
                int iSum = 0;
                int iNum = 0;
                int iCntr = 0;

  // Input a number, add it to the sum, and repeat
  // until two numbers have been entered
      do
         {
         System.out.println("Enter number: ");
         inData = stdin.readLine();                      // get number in character form
         iNum = Integer.parseInt( inData );              // convert inData to integer
         iSum = iSum + iNum;
         iCntr = iCntr + 1;
         }
      while (iCntr < 2);

// Print the sum
      System.out.println("The sum of the numbers you entered is " + iSum);
               }
           }
```

The java.io package will handle the user input; * indicates all classes will be available.

Comments within the code are preceded by two slashes //.

The *out* attribute and *println* method in the System class of the java.io package are used to output the results.

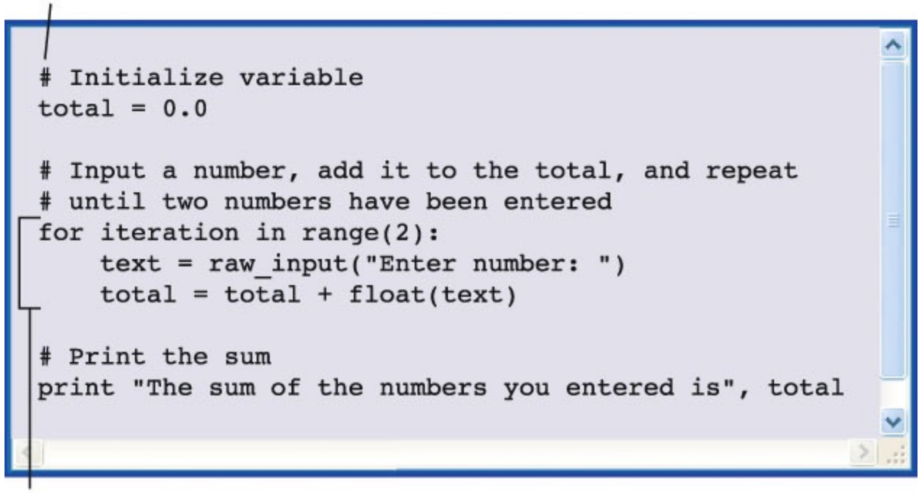
FIGURE 13-24

The adding-two-numbers program written in Java.

Programming Languages

- Python
  - Open-source, dynamic, object-oriented language that can be used to develop a variety of applications
  - Gaming, scientific, database, and Web applications
  - Used by large organizations such as NASA, Google, Honeywell, New York Stock Exchange, and some colleges such as MIT

Comments are preceded by a pound symbol #.



```
# Initialize variable
total = 0.0

# Input a number, add it to the total, and repeat
# until two numbers have been entered
for iteration in range(2):
    text = raw_input("Enter number: ")
    total = total + float(text)

# Print the sum
print "The sum of the numbers you entered is", total
```

The indented statements in this For statement will be executed two times.

Programming Languages

- Ruby
  - Open-source, object-oriented language that can be used to create Web applications and general-purpose programming for Linux, Mac OS X, and Microsoft Windows computers
  - Uses a syntax that is fairly easy to read and write, allowing programmers to create database-driven Web applications easily and quickly

Quick Quiz

1. An example of a high-level programming language is ________________________.
   a. Pascal
   b. Assembly language
   c. Machine language
2. True or False: Visual Basic is an object-oriented version of COBOL.
3. Java applets are small programs written in the ________________________ programming language.

Answers:

1) a; 2) False; 3) Java

Summary

- Approaches to Program Design and Development
- The Program Development Life Cycle (PDLC)
- Tools for Facilitating Program Development
- Programming Languages