Chapter 2: *Problem Solving*

- Program Development Cycle (2.1)
- Programming Tools (2.2)

David I. Schneider, *An Introduction to Programming using Visual Basic.NET, 5th Edition*, Prentice Hall, 2002.

Terminology tip

- A computer program may also be called:
  - Project
  - Application
  - Solution

Program Development Cycle (2.1)

- Performing a Task on the Computer
- Program Planning

Program Development Cycle (2.1) (cont.)

- Program Development Cycle
  - Software refers to a collection of instructions for the computer
  - The computer only knows how to do what the programmer tells it to do
  - Therefore, the programmer has to know how to solve problems

Program Development Cycle (2.1) (cont.)

- Performing a Task on the Computer
  - Determine Output
  - Identify Input
  - Determine process necessary to turn given Input into desired Output

Program Development Cycle (2.1) (cont.)

- Problem-solving approach like algebra class
  - How fast is a car traveling if it goes 50 miles in 2 hours?
  - **Output**: a number giving the rate of speed in miles per hour
  - **Input**: the distance and time the car has traveled
  - **Process**: rate = distance/time

Program Development Cycle (2.1) (cont.)

- Pictorial representation of the problem solving process

Program Development Cycle (2.1) (cont.)

- Program Planning
  - A recipe is a good example of a plan
  - Ingredients and amounts are determined by what you want to bake
  - Ingredients are input
  - The way you combine them is the processing
  - What is baked is the output

Program Development Cycle (2.1) (cont.)

- Program Planning Tips
  - Always have a plan before trying to write a program
  - The more complicated the problem, the more complex the plan must be
  - Planning and testing before coding saves time coding

Program Development Cycle (2.1) (cont.)

- Program development cycle
  1. Analyze: Define the problem.
  2. Design: Plan the solution to the problem.
  3. Choose the interface: Select the objects (text boxes, buttons, etc.).

Program Development Cycle (2.1) (cont.)

- Program development cycle continued

4. Code: Translate the algorithm into a programming language.
5. Test and debug: Locate and remove any errors in the program.
6. Complete the documentation: Organize all the material that describes the program.

Programming Tools (2.2)

- Three tools used to convert algorithms into computer programs:
- Flowcharts - Graphically depict the logical steps to carry out a task and show how the steps relate to each other.
- Pseudocode - Uses English-like phrases with some VB.NET terms to outline the program.
- Hierarchy charts - Show how the different parts of a program relate to each other.

Programming Tools (2.2) (cont.)

- Algorithms
  - A step by step series of instructions for solving a problem (a recipe is an example of an algorithm)

Programming Tools (2.2) (cont.)

- Problem solving example
  - How many stamps do you use when mailing a letter?
  - One rule of thumb is to use one stamp for every five sheets of paper or fraction thereof.

Programming Tools (2.2) (cont.)

- Algorithm

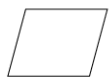
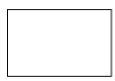
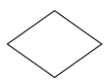
1. Request the number of sheets of paper; call it Sheets. (input)
2. Divide Sheets by 5. (processing)
3. Round the quotient up to the next highest whole number; call it Stamps. (processing)
4. Reply with the number Stamps. (output)

Programming Tools (2.2) (cont.)

- Flowcharts
  - Graphically depict the logical steps to carry out a task and show how the steps relate to each other.

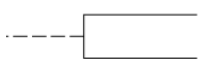
Programming Tools (2.2) (cont.)

◆Flowchart symbols

Symbol	Name	Meaning
	Flowline	Used to connect symbols and indicate the flow of logic.
	*Terminal*	Used to represent the beginning (Start) or the end (End) of a task.
	*Input/Output*	Used for input and output operations, such as reading and displaying. The data to be read or displayed are described inside.
	*Processing*	Used for arithmetic and data-manipulation operations. The instructions are listed inside the symbol.
	*Decision*	Used for any logic or comparison operations. Unlike the input/ouput and processing symbols, which have one entry and one exit flowline, the decision symbol has one entry and two exit paths. The path chosen depends on whether the answer to a question is "yes" or "no."

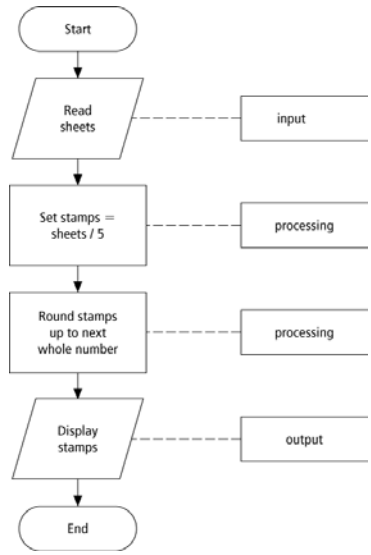
Programming Tools (2.2) (cont.)

◆Flowchart symbols continued

	*Connector*	Used to join different flowlines.
	*Offpage Connector*	Used to indicate that the flowchart continues to a second page.
	*Predefined Process*	Used to represent a group of statements that perform one processing task.
	*Annotation*	Used to provide additional information about another flowchart symbol.

Programming Tools (2.2) (cont.)

◆Flowchart example



Programming Tools (2.2) (cont.)

◆Pseudocode

– Uses English-like phrases with some VB.NET terms to outline the task.

Programming Tools (2.2) (cont.)

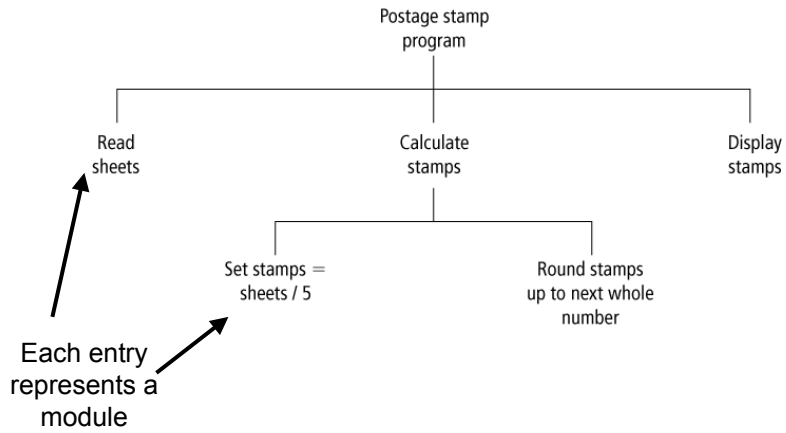
- Pseudocode example
  - Determine the proper number of stamps for a letter
  - Read Sheets (input)
  - Set the number of stamps to Sheets / 5 (processing)
  - Round the number of stamps up to the next whole number (processing)
  - Display the number of stamps (output)

Programming Tools (2.2) (cont.)

- Hierarchy charts
  - Show how the different parts of a program relate to each other
  - Hierarchy charts may also be called
    - structure charts
    - HIPO (Hierarchy plus Input-Process-Output) charts
    - top-down charts
    - VTOC (Visual Table of Contents) charts

Programming Tools (2.2) (cont.)

- Hierarchy charts example



Programming Tools (2.2) (cont.)

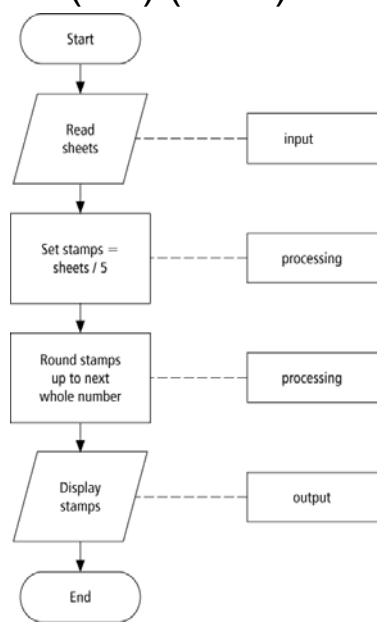
- Divide-and-conquer method
  - Used in problem solving – take a large problem and break it into smaller problems solving the small ones first
  - Breaks a problem down into modules

Programming Tools (2.2) (cont.)

- Statement structure
  - Sequence – follow instructions from one line to the next without skipping over any lines
  - Decision - if the answer to a question is “Yes” then one group of instructions is executed. If the answer is “No,” then another is executed
  - Looping – a series of instructions are executed over and over

Programming Tools (2.2) (cont.)

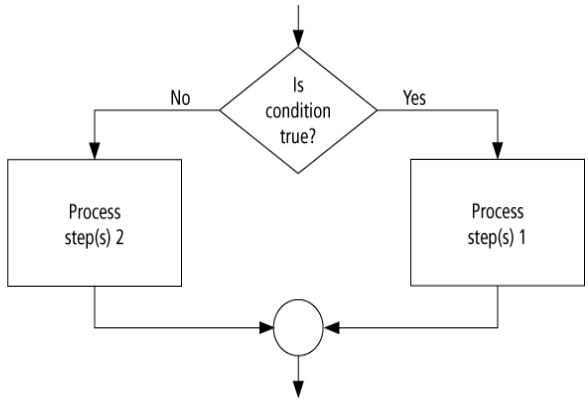
- Sequence flow chart



Programming Tools (2.2) (cont.)

◆Decision flow chart

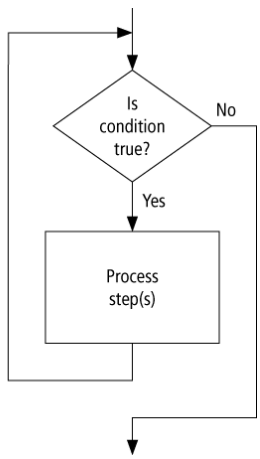
```
If condition is true Then
      Process step(s) 1
  Else
      Process step(s) 2
End If
```



Programming Tools (2.2) (cont.)

◆Looping flow chart

```
Do While condition is true
    Process step(s)
Loop
```

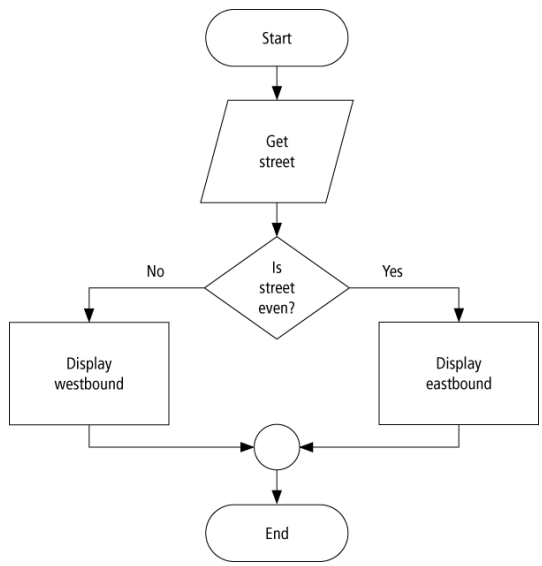


Programming Tools (2.2) (cont.)

- Direction of Numbered NYC Streets Algorithm
  - *Problem:* Given a street number of a one-way street in New York, decide the direction of the street, either eastbound or westbound
  - *Discussion:* in New York even numbered streets are Eastbound, odd numbered streets are Westbound

Programming Tools (2.2) (cont.)

- Flowchart



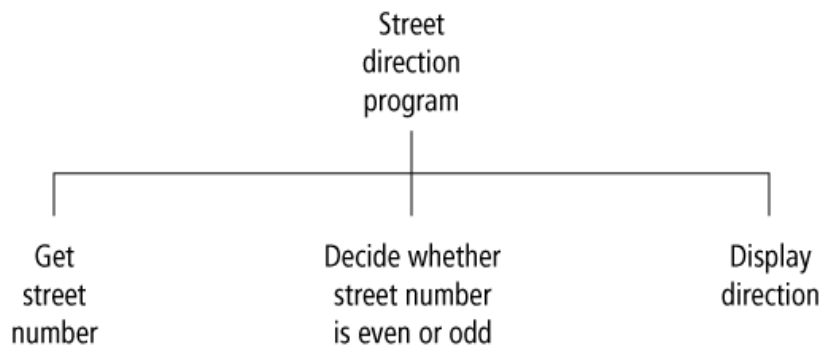
Programming Tools (2.2) (cont.)

- Pseudocode

```
Program: Determine the direction of a numbered
   NYC street
Get street
If street is even Then
  Display Eastbound
Else
  Display Westbound
End If
```

Programming Tools (2.2) (cont.)

- Hierarchy Chart



Programming Tools (2.2) (cont.)

- Class Average Algorithm
  - *Problem:* Calculate and report the grade-point average for a class
  - *Discussion:* The average grade equals the sum of all grades divided by the number of students

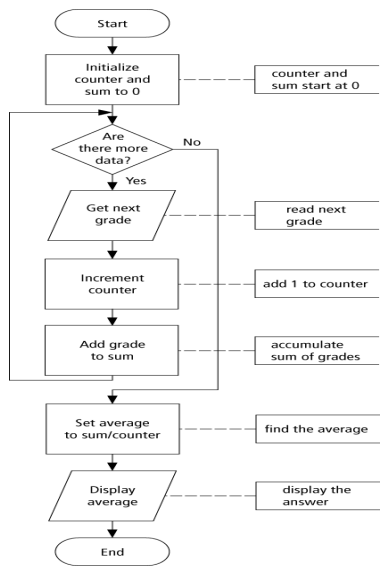
Output: Average grade

Input: Student grades

Processing: Find the sum of the grades; count the number of students; calculate average

Programming Tools (2.2) (cont.)

- Flowchart



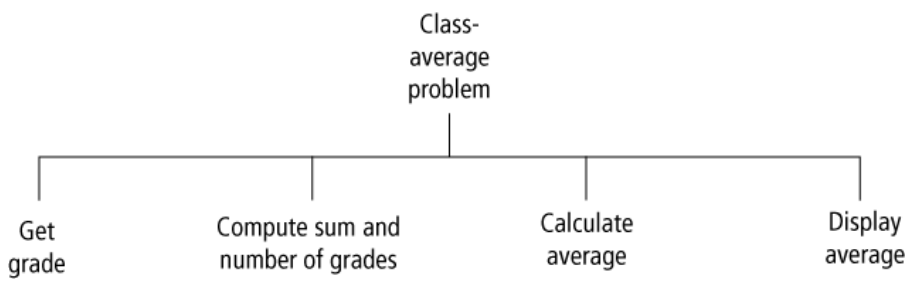
Programming Tools (2.2) (cont.)

◆Pseudocode

Program: Determine the average grade of a class
Initialize Counter and Sum to 0
Do While there are more data
  Get the next Grade
  Add the Grade to the Sum
  Increment the Counter
Loop
Computer Average = Sum/Counter
Display Average

Programming Tools (2.2) (cont.)

◆Hierarchy Chart



Programming Tools (2.2) (cont.)

- Comments
  - When tracing a flow chart, start at the start symbol and follow the flow lines to the end symbol
  - Testing an algorithm at the flow chart stage is known as desk checking
  - Flowcharts, pseudocode, and hierarchy charts are program planning tools that are not dependent on the programming language being used

Programming Tools (2.2) (cont.)

- Comments continued
  - There are four primary logical programming constructs
    - sequence
    - decision
    - loop
    - unconditional branch

Programming Tools (2.2) (cont.)

- Unconditional branch
  - Appear in some languages as Goto statements
  - Involves jumping from one place in a program to another
  - Structured programming uses the sequence, decision, and loop but forbids unconditional branch

Programming Tools (2.2) (cont.)

- Tips and tricks of flowcharts
  - Flowcharts are time-consuming to write and difficult to update
  - For this reason, professional programmers are more likely to favor pseudocode and hierarchy charts
  - Because flowcharts so clearly illustrate the logical flow of programming techniques, they are a valuable tool in the education of programmers

Programming Tools (2.2) (cont.)

- Tips and tricks of pseudocode
  - There are many styles of pseudocode
  - Some programmers use an outline form
  - Some use a form that looks almost like a programming language
  - The case studies of this text focuses on the primary tasks to be performed by the program and leaves many of the routine details to be completed during the coding process

Programming Tools (2.2) (cont.)

- Tips and tricks of hierarchy charts
  - Many people draw rectangles around each item in a hierarchy chart
  - In the text, rectangles are omitted to encourage the use of hierarchy charts by making them easier to draw