Chapter 2

Introduction to Programming

Objectives

- After studying Chapter 2, you should be able to:
  - Describe the steps involved in the programming process
  - Understand basic programming constructs
  - Understand how to use flowchart symbols and pseudocode statements

Understanding the steps in preparing a program

- A programmer's job involves writing instructions, and can be broken down into six programming steps:

1. Understand the problem
2. Plan the logic
3. Code the program
4. Translate the program into machine language
5. Test the program
6. Put the program into production

Understand the Problem

- Professional computer programmers write programs to satisfy the needs of others

- Programmers must first understand what it is the users want

Understand the Problem

- Problem:
  - Lecturer needs a printed list of students
  - Students need to know the results.
  - Billing department wants a list of clients who are 30 or more days overdue in their payments
  - Reads three integers and finds their average

Plan the Logic

The heart of the programming process lies in planning the program's logic

- The programmers plan
  - The steps to the program
  - Deciding what steps to include
  - How to order them
- Common tools
  - Flowcharts
  - Pseudocode
- Both involves writing steps of the program in English

Code the Program and Translate the program into machine language

- Programmers can write the program in programming languages such as VB, Java, c++ and etc
- Must use the correct syntax

Test the Program and Put the program into production

- A program that is free of *syntax errors* is not necessarily free of logical errors
- Once a program is free from *syntax errors*, the programmer can test it—that is, execute it with some sample data to see whether or not the results are logically correct
- Programs should be tested with many sets of data
- Selecting test data is somewhat of an art in itself, and it should be done carefully

Algorithm: Using Flowchart Symbols and Pseudocode Statements

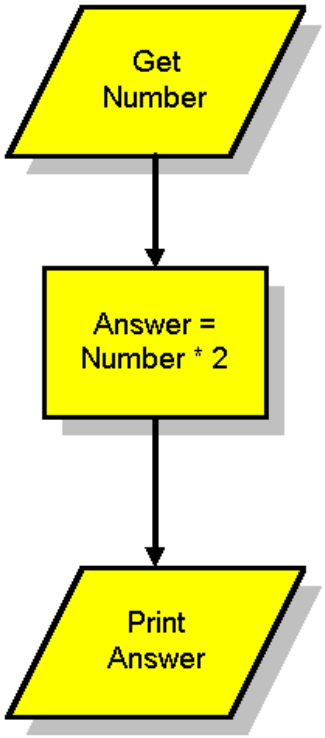
- When programmers plan the logic for a solution to a programming problem, they often use one of two tools, flowcharts or pseudocode
- A flowchart is a pictorial representation of the logical steps it takes to solve a problem
- Pseudocode is an English-like representation of the same thing
- Using pseudocode involves writing down all the steps you will use in a program

Algorithm: Using Flowchart Symbols and Pseudocode Statements

- Example:
  - Problem: input a number by user and calculate the number times 2.

Flowcharting vs. Pseudocode

Flowchart - pictorial representation of logical steps



Pseudocode **- English-like representation of logical steps**

Get inputNumber
*calculatedAnswer = inputNumber * 2*
Print calculatedAnswer

Input Statement

Get inputNumber
calculatedAnswer =
*inputNumber * 2*
Print calculatedAnswer

Input statement in pseudocode

Get inputNumber

In a flowchart, an input statement is represented by a parallelogram

Processing Statement

Get inputNumber

*calculatedAnswer = inputNumber * 2*

Print calculatedAnswer

Processing statement in pseudocode

Compute calculatedAnswer as inputNumber times 2

In a flowchart, a processing statement is represented by a rectangle

Output Statement

Get inputNumber
*calculatedAnswer = inputNumber * 2*
Print calculatedAnswer

Output statement in pseudocode

Print calculatedAnswer

In a flowchart, an output statement is also represented by a parallelogram

Using Flowchart Symbols and Pseudocode Statements

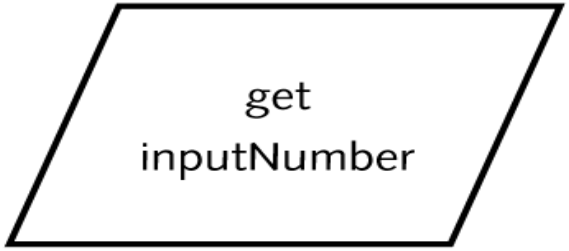


Figure 1-3 Input symbol

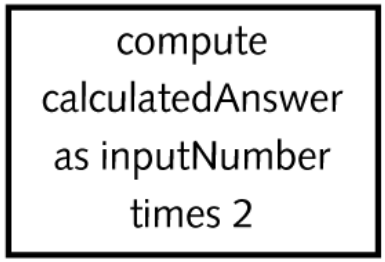


Figure 1-4 Processing symbol

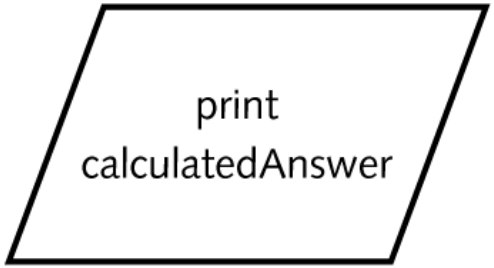


Figure 1-5 Output symbol

Using Flowchart Symbols and Pseudocode Statements

- Arrows (flowlines) are used to connect steps
  - Top to bottom
  - Left to right

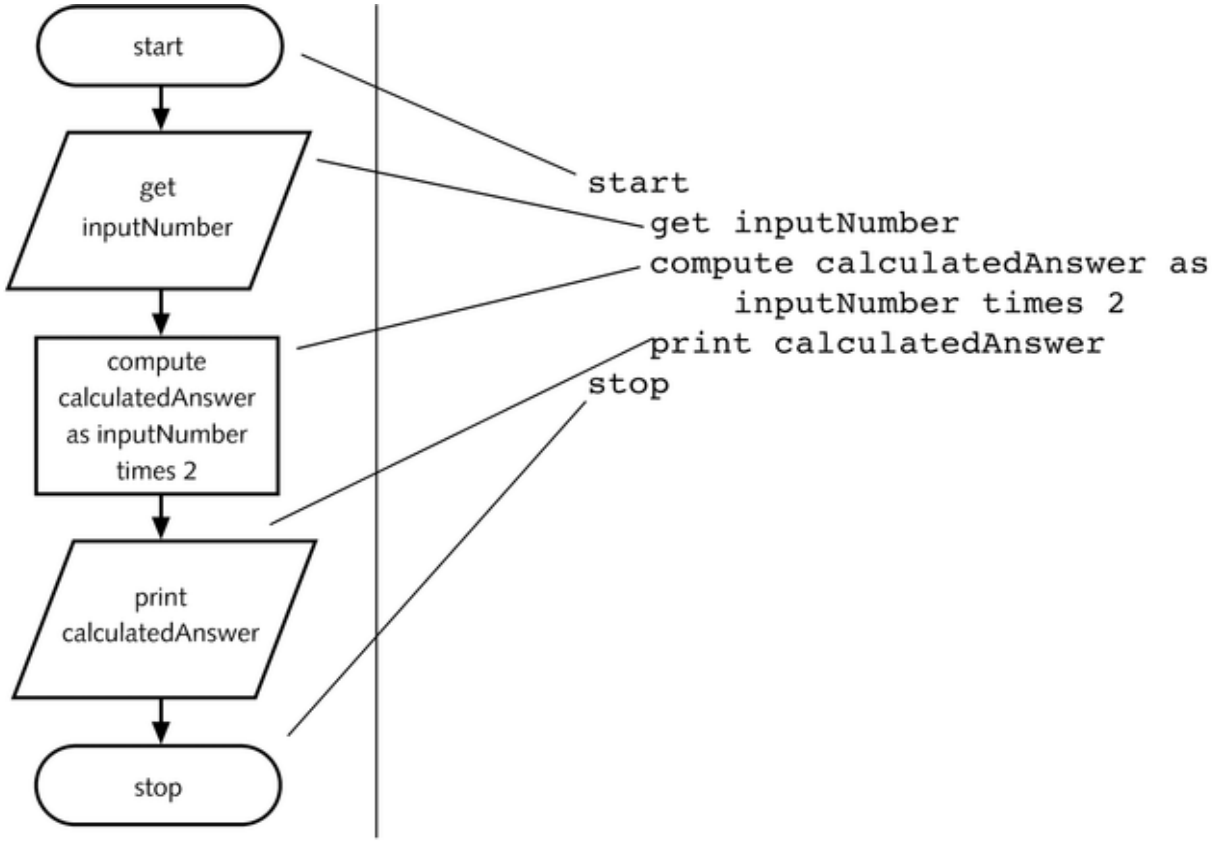


Figure 1-6 Flowchart and pseudocode of program that doubles a number

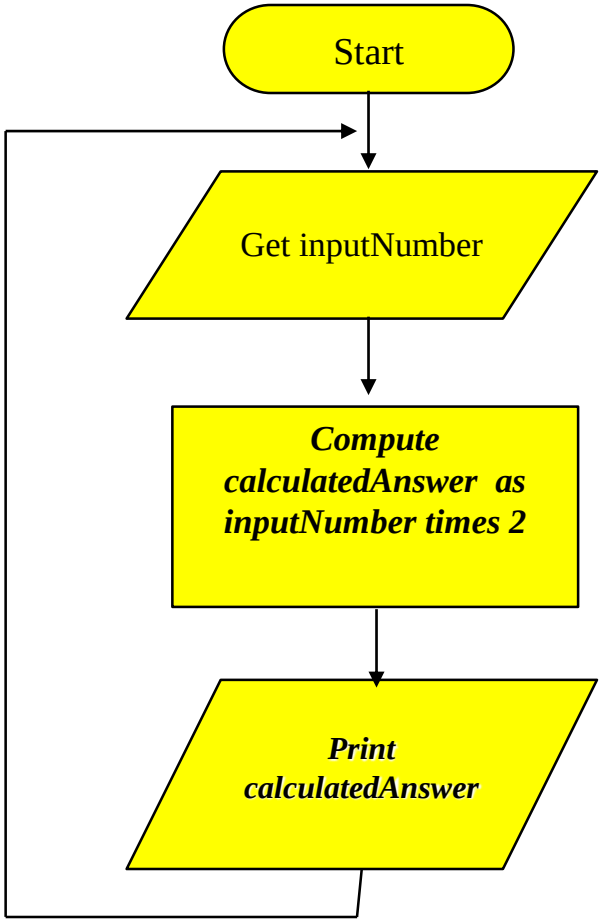
When Programs Are Practical

- Would you need to create the number doubling program if you only wanted to use one number as input?
- What if you had lots of input numbers to double? How would the program work?
- What would be the most practical way for a computer program to double a large amount of input numbers?

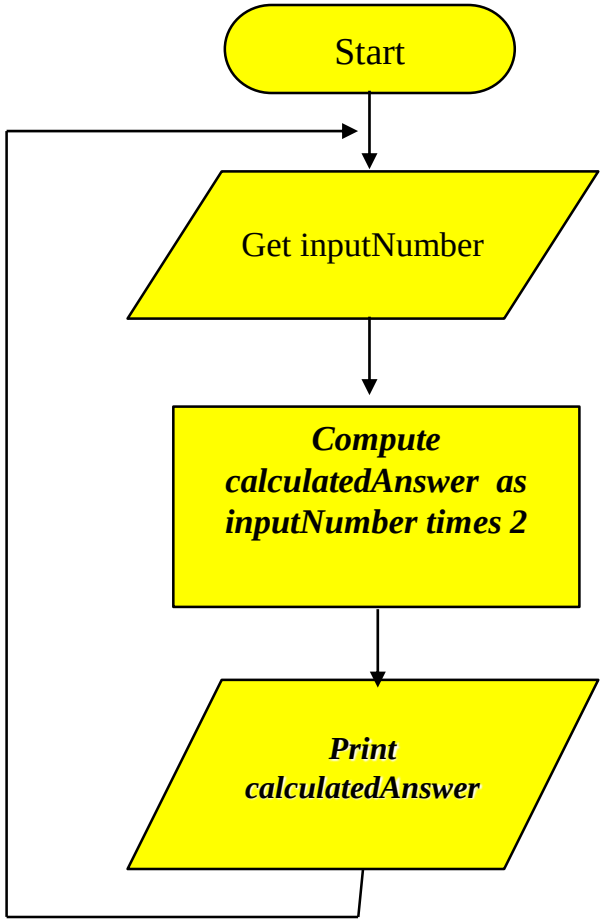
A Better Solution

- The number doubling program can be rewritten so that the steps can be executed over and over again.

- Is there a problem with this program?



The Never-Ending Program

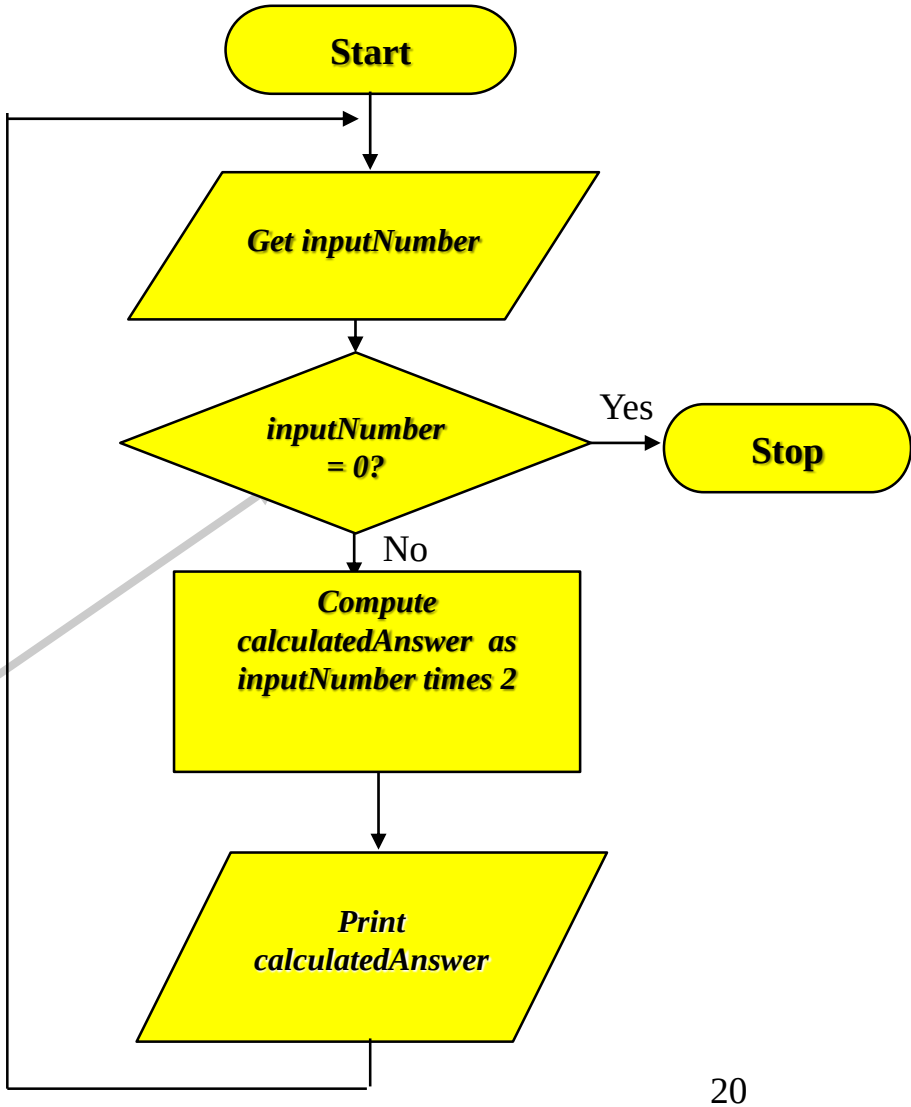


- Remember this program - it never ends!
- How can we get this program to end after the last input number is doubled?

Ending a Program By Using Sentinel Values

- Using a predetermined input number to stop the program
- To determine if the program should stop, a decision symbol or diamond is used

  - Dummy or sentinel value



Ending a Program by Using Sentinel Values

- A superior way to end the program is to set a predetermined value for `inputNumber` that means "Stop the program!"
- The program could then test any incoming value for `inputNumber` and, if it is a zero, stop the program
- Testing a value is also called making a decision
- You represent a decision in a flowchart by drawing a decision symbol or a diamond

Flowchart for Number-Doubling Program with Sentinel Value of Zero

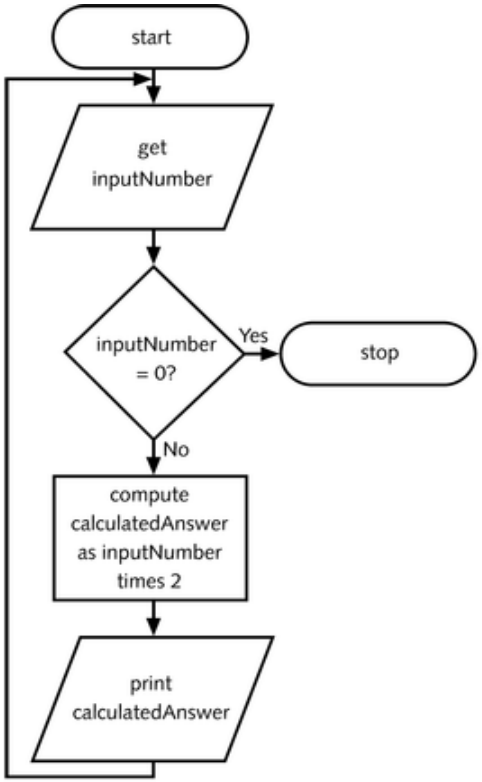


Figure 1-9 Flowchart for number-doubling program with sentinel value of zero

```
START
    GET inputNumber
    IF inputNumber is equal to zero THEN stop the program
    COMPUTE calculatedAnswer as inputNumber times 2
    PRINT calculatedAnswer
    CONTINUE to get inputNumber
END
START
    REPEAT
    GET inputNumber
    IF inputNumber is not equal to zero THEN
       COMPUTE calculatedAnswer as inputNumber times 2
       PRINT calculatedAnswer
    ENDIF
    UNTIL inputNumber is equal to zero  (inputNumber == 0)
END
```

Ending a Program by Using Sentinel Values

- One drawback to using zero to stop a program is that it won't work if the user *does* need to find the double of zero
- The diamond usually contains a question, the answer to which is either yes or no
- All good computer questions have two mutually exclusive answers like yes and no or true and false

Ending a Program by Using Sentinel Values

- A preselected value that stops the execution of a program is often called a dummy value because it does not represent real data, but just a signal to stop
- Sometimes such a value is called a sentinel value because it represents an entry or exit point like a sentinel that guards a fortress
- Many programming languages use the term eof (for “end of file”) to talk about this marker

Flowchart Using EOF

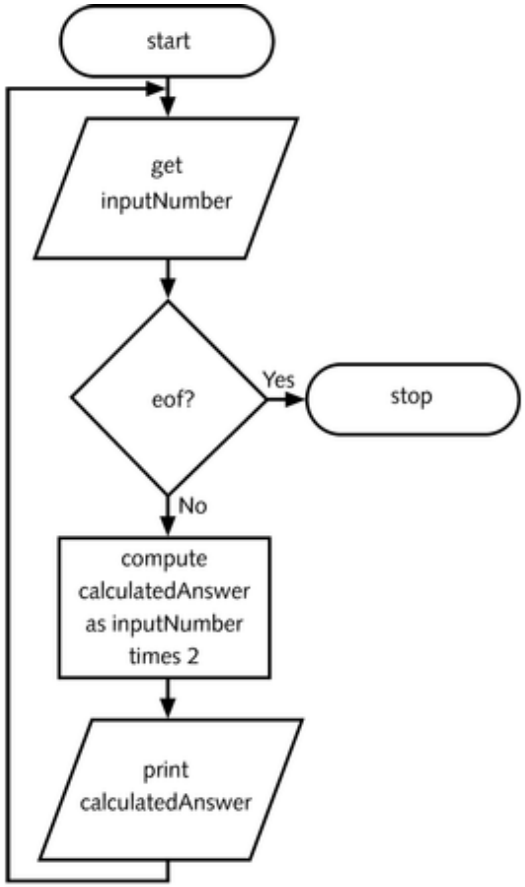


Figure 1-10 Flowchart using eof

Summary

- A programmer's job involves understanding the problem, planning the logic, coding the program, translating the program into machine language, testing the program, and putting the program into production
- When programmers plan the logic for a solution to a programming problem, they often use flowcharts or pseudocode
- Testing a value involves making a decision
- You represent a decision in a flowchart by drawing a diamond-shaped decision symbol which contains a question

Exercises

- Consider the following problem statement; please write a pseudocode and draw a flowchart for it.
- Write a program that prompts the user to enter the distance to drive, the fuel efficiency of the car in miles per gallon, and the price per gallon then displays the cost of the trip. Here is a sample run:
  - Enter the driving distance: 150
  - Enter miles per gallon: 20
  - Enter price per gallon: 1.80
  - The cost of driving is RM13.5