Session 7: Control Structures

7.1 Introduction

All programs in one way or another, are written using one or more of the three basic structures: sequence, selection, and repetition. These structures are known as control structures, because they control the flow of execution of program statements. It is important to note that the control structure used in every program is the sequence structure; meanwhile, the selection and repetition structures are used in most programs. The idea is that a program must be made of a combination of only these three structures: sequence, selection, and repetition. It has been proved there is no need for any other structure. Figure 7-1 shows the three structures using pseudocode.

```
action 1
action 2
   :
   :
action n
```

a. Sequence

```
if(condition)
   then
       action
       action
       ...
   else
       action
       action
       ...
End if
```

b. Selection

```
while(condition)
    action
    action
    ...
End while
```

c. Repetition

Figure 7-1: Pseudocode for three control structures

7.1.1 The Sequence Structure

You must be familiar with the sequence structure, since you use it every time you follow a set of instructions, in order, from the beginning to the end. A cookie recipe, for example, is a good candidate of the sequence structure. Since, to get a finished product (edible cookies), you need to follow each recipe instruction in order, beginning with the first instruction and ending with the last. An algorithm, and eventually a program is a sequence of instructions, which can be a simple instruction or either of the other two structures. In a program, the sequence structure directs the computer to process the program instructions, one after another, in the order listed in the program. It is the only control structure found in every program.

Example 1

Write an algorithm in pseudocode that finds the average of two numbers.

Solution

This is a simple problem that can be solved using only the sequence structure. Note that we number the instructions for easy of reference (see Algorithm 7-1). Note also that we name the algorithm, define the input to the algorithm, and at the end, show the output using a return instruction.

Algorithm 7-1: Average of two

```
    AverageOfTwo
    Input: Two numbers
 1. Add the two numbers
 2. Divide the result by 2
 3. Return the result of Step 2
    End
```

7.1.2 The Selection Structure

The selection structure indicates that a decision (based on some condition) needs to be made, followed by an appropriate action derived from that decision. Some problems cannot be solved with only a sequence of simple instructions. Sometimes you need to test a condition. If the result of testing is true, you follow a sequence of instructions; if false, you follow a different sequence of instructions.

When used in a program, the selection structure alerts the computer that a decision needs to be made, and it provides the appropriate action to take based on the result of that decision. The selection structure is also referred to as the decision structure.

Example 2

Write an algorithm to change a mark of unit to its equivalent a letter grade.

Solution

This problem needs more than one decision. The pseudocode in Algorithm 7-2 shows one way to solve the problem. Again, a mark is given which is between 0 and 100, and you want to change it to its equivalent letter grade (A, B, C, D, or E).

Algorithm 7-2: Letter grade

```
    LetterGrade
    Input: One mark
 1. if(the mark is between 70 and 100, both marks inclusive)
       then
           1.1   Set the grade to “A”
    End if
 2. if(the mark is between 60 and 70, 60 inclusive)
       then
           2.1   Set the grade to “B”
    End if
```

```
3. if(the mark is between 50 and 60, 50 inclusive)
      then
         3.1    Set the grade to "C"
   End if
4. if(the mark is between 40 and 50, 40 inclusive)
      then
         4.1    Set the grade to "D"
   End if
5. if(the mark is between 0 and 40, 0 inclusive)
      then
         5.1    Set the grade to "E"
   End if
6. Return the grade
   End
```

Note that the if instructions do need an else because you do nothing if the condition is false.

7.1.3 The Repetition Structure

In some problems, the same sequence of instructions must be repeated. You handle this with the repetition structure; which is also referred to as a loop or as an iteration structure. When used in a program, the repetition structure directs the computer to repeat one or more instructions until some condition is met, at which time the computer should stop repeating the instructions. The Example 3 below, illustrates the repetition using pseudocode.

Example 3

Write an algorithm to find the largest of a set of integers. You do not know the number of integers.

Solution

Use the concept in Figure 7-2 to write an algorithm for this problem (see Algorithm 7-3).

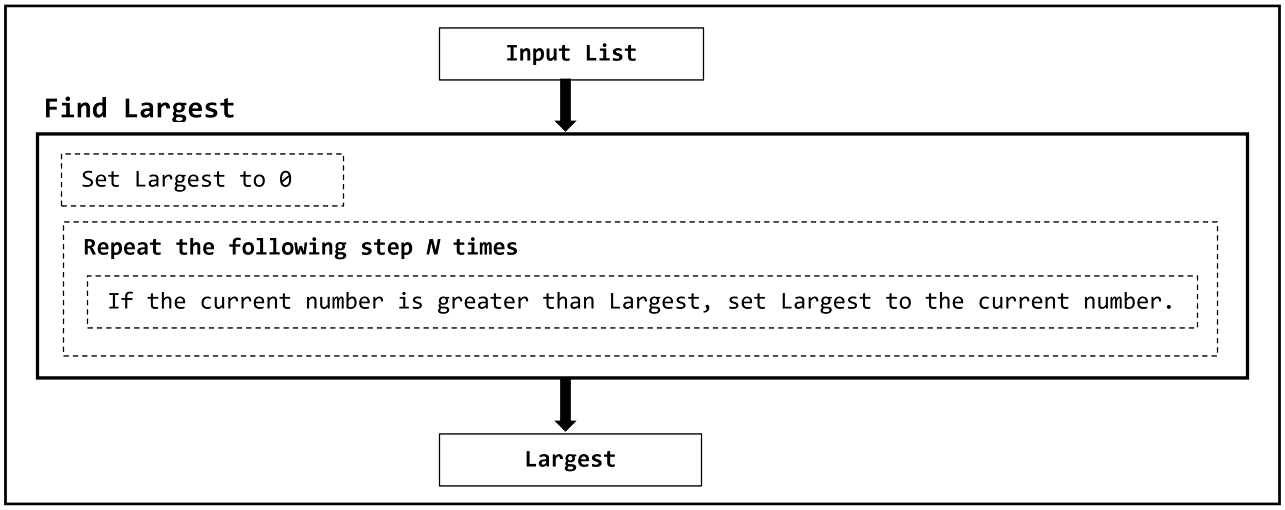
Input List
Find Largest
Set Largest to 0
Repeat the following step N times
If the current number is greater than Largest, set Largest to the current number.
Largest

Figure 7-2: Generalization of the FindLargest algorithm

Algorithm 7-3: Find Largest

```
    FindLargest
    Input: A list of positive integers
 1. Set Largest to 0
 2. while (more integers)
       2.1 if (the integer is greater than Largest)
                then
                     2.1.1 Set Largest to the value of the integer
           End if
    End while
 3. Return Largest
    End
```

7.2 Pseudocode

We had discussed this tool for algorithm representation earlier in Session 6; however, it is important to have a formal specification of pseudocode that we will use in this unit of study.

As said earlier, pseudocode is an English-like representation of the algorithm required. It is part English and part structured code. The English part provides a relaxed syntax that is easy to read. The code part consists of an extended version of the basic control structures: sequence, selection and iteration. Algorithm 7-4 illustrates an example of pseudocode.

Algorithm 7-4: Finding Smallest

```
Algorithm:   Find Smallest
Purpose: This algorithm finds the smallest number among a list of numbers.
Pre: List of numbers
Post: None
Return: The smallest

1. Set smallest to the first number
2. loop (not end of list)
      2.1 if (next number < smallest)
              2.1.1 Set smallest to next number
          end if
   end loop
3. return smallest
End Finding Smallest
```

7.2.1 Pseudocode Components

An algorithm written in pseudocode can be decomposed into several elements and control structures.

A. ALGORITHM HEADER

Each algorithm begins with a header that names it. For example, in Algorithm 7-4, the header starts with the Algorithm, which names the algorithm as "Finding Smallest".

B. PURPOSE, CONDITIONS AND RETURN

After the header, you normally mention the purpose, the pre – and post conditions, and the data returned from the algorithm.

Purpose

The purpose is a short statement about what the algorithm does. It needs to describe only the general algorithm processing. It should not attempt to describe all the processing. In Algorithm 7-4, the purpose starts with the word Purpose and continues with goal of the algorithm.

Precondition

The precondition lists any precursor requirements. For example, in Algorithm 7-4, we require that the list be available to the algorithm.

Postcondition

The postcondition identifies any effect created by the algorithm. For example, perhaps the algorithm specifies the printing of data.

Return

We believe that every algorithm should show what is returned from the algorithm. If there is nothing to be returned, we advise that null be specified. In Algorithm 7-4, the smallest value found is returned.

C. STATEMENT NUMBERS

Statements are numbered as shown in Algorithm 7-4 (1, 2, 3, ….). The dependent statements are numbered in a way that shows their dependencies (1.1, 2.4, ….).

D. CONTROL STRUCTURES

When Niklaus Wirth first proposed the structured programming model, he stated that any algorithm could be written with only three programming control structures: sequence, selection, and repetition. Our pseudocode contains only these three basic control structures as discussed above.

Exercise

(a.) The real roots of a quadratic equation of the form:

$ax^2 + bx + c = 0$, where a, b, c are constants,

can be computed using the following formula:

$$x = \frac{-b \pm \sqrt{(b^2 - 4ac)}}{2a}$$

Write an algorithm in pseudocode that computes and displays the real roots of the equation using the formula given above.

(b.) Write an algorithm in pseudocode that computes and displays the sum of the first hundred positive integers.