

UNIT 3 FLOWCHARTS AND ALGORITHMS

CONTENTS

- 1.0 Introduction
- 2.0 Objective
- 3.0 Main Content
 - 3.1 Flowcharts
 - 3.2 Flowchart Symbols
 - 3.3 Guidelines for Drawing
 - 3.4 Flowcharting the Problem
 - 3.5 Algorithms
 - 3.6 Pseudo Codes
 - 3.7 Decision Tables
- 4.0 Conclusion
- 5.0 Summary
- 6.0 Tutor-Marked Assignment
- 7.0 References/Further Reading

1.0 INTRODUCTION

This unit introduces to the principles of flowcharts and algorithms. The importance of these concepts is presented and the detailed steps and activities involved are also presented.

2.0 OBJECTIVE

At the end of this unit you should be able to:

- explain the principles of good programming ethics through flowcharting and algorithms.

3.0 MAIN CONTENT


3.1 Flowcharts

A flowchart is a graphical representation of the major steps of work in process. It displays in separate boxes the essential steps of the program and shows by means of arrows the directions of information flow. The boxes, most often referred to as illustrative symbols, may represent documents, machines or actions taken during the process. The area of concentration is on where or who does what, rather than on how it is done. A flowchart can also be said to be a graphical representation of an algorithm,

that is, it is a visual picture which gives the steps of an algorithm and also the flow of control between the various steps.

3.2 Flowchart Symbols

Flowcharts are drawn with the help of symbols. The following are the most commonly used flowchart symbols and their functions:

Symbols	Function
	Used to show the START or STOP May show exit to a closed subroutine. Terminator
for arithmetic calculations of process. E.g. $\text{Sum} = X + Y + Z$	Used
Used for Input and Output instructions, PRINT, READ, INPUT AND WRITE.	
for decision making. Has two or more lines leaving the box. These lines are labeled with different decision results, that is, 'Yes', 'No', 'TRUE' or 'FALSE' or 'NEGATIVE' or 'ZERO'.	Used
	Used for one or more named operations or program steps specified in a subroutine or another set of flowchart.
for entry to or exit from another part of flowchart. A small circle identifies a junction point of the program	Used

Used for entry to or exit from a page	
Used to show the direction of travel. They are used in linking symbols. These show operations sequence and data flow directions.	

Fig.15: Flowchart symbols and their functions

3.3 Guidelines for Drawing Flowcharts

- Each symbol denotes a type of operation: Input, Output, Processing, Decision, Transfer or Branch or Terminal.
- A note is written inside each symbol to indicate the specific function to be performed.
- Flowcharts are read from top to bottom.
- A sequence of operations is performed until a terminal symbol designates the end of the run or “branch” connector transfers control.

3.4 Flowcharting the Problem

The digital computer does not do any thinking and cannot make unplanned decisions. Every step of the problem has to be taken care of by the program. A problem which can be solved by a digital computer need not be described by an exact mathematical equation, but it does need a certain set of rules that the computer can follow. If a problem needs intuition or guessing, or is so badly defined that it is hard to put into words, the computer cannot solve it. You have to define the problem and set it up for the computer in such a way that every possible alternative is taken care of. A typical flowchart consists of special boxes, in which are written the activities or operations for the solution of the problem. The boxes, linked by means of arrows, show the sequence of operations. The flowchart acts as an aid to the programmer, who follows the flowchart design to write his programs.

3.5 Algorithms

Before a computer can be put to any meaningful use, the user must be able to come out with or define a unit sequence of operations or activities (logically ordered) which gives an unambiguous method of solving a problem or finding out that no solution exists. Such a set of operations is known as an ALGORITHM.

Definition: An algorithm, named after the ninth century scholar Abu Jafar Muhammad Ibn Musu Al-Khowarizmi, is defined as follows:

- An algorithm is a set of rules for carrying out calculations either by hand or a machine.
- An algorithm is a finite step-by-step procedure to achieve a required result.
- An algorithm is a sequence of computational steps that transform the input into the output.
- An algorithm is a sequence of operations performed on data that have to be organised in data structures.
- An algorithm is an abstraction of a program to be executed on a physical machine (model of computation).

The most famous algorithm in history dates well before the time of the ancient Greeks: this is Euclid's algorithm for calculating the greatest common divisor of two integers. Before we go into some otherwise complex algorithms, let us consider one of the simplest but common algorithms that we encounter everyday.

The classic multiplication algorithm

For example to multiply 981 by 1234, this can be done using two methods (algorithms) viz:

a. Multiplication the American way:

Multiply the multiplicand one after another by each digit of the multiplier taken from right to left.

$$\begin{array}{r}
 981 \\
 1234 \\
 \hline
 3924 \\
 2943 \\
 1962 \\
 981 \\
 \hline
 1210554
 \end{array}$$

b. Multiplication, the British way:

Multiply the multiplicand one after another by each digit of the multiplier taken from left to right.

$$\begin{array}{r}
 981 \\
 1234 \\
 \hline
 \end{array}$$

981
1962
2943
3924
1210554

An algorithm therefore can be characterised by the following:

- (i) A finite set or sequence of actions
- (ii) This sequence of actions has a unique initial action
- (iii) Each action in the sequence has unique successor
- (iv) The sequence terminates with either a solution or a statement that the problem is unresolved.

An algorithm can therefore be seen as a step-by-step method of solving a problem.

Examples

1. Write an algorithm to read values for three variables. U, V, and W and find a value for RESULT from the formula: $RESULT = U + V^2/W$. Draw the flowchart.

Solution

Algorithm

- (i) Input values for U, V, and W
- (ii) Computer value for result
- (iii) Print value of result
- (iv) Stop

2. Suppose you are given 20 numbers. Prepare the algorithm that adds up these numbers and find the average. Draw the flowchart.

Solution

Algorithm

- Set up a Counter (1) which counts the number of times the loop is executed. Initialise Counter (1) to 1.
 - Initialize sum to zero.
 - Input value and add to sum.
 - Increment the counter (1) by 1.
 - Check how many times you have added up the number, if it is not up to the required number of times, to step (iii).
 - Compute the average of the numbers.
 - Print the average.
 - Stop.
3. Prepare an algorithm that indicates the logic for printing the name and telephone number for each female in a file (Code field is 2 for female). Draw the flowchart.

Solution

Algorithm

- (i) Read a record
- (ii) Determine if the record pertains to a female (that is, determine if the code field is equal to 2).
- (iii) If the code field is not equal to 2, then do not process this record any further, since it contains data for a male. Instead, read the next record; that is, go back to step (i).
- (iv) If the record contains data for a female (that is, code is equal to 2), then print out the following fields: first name, last name, telephone number.
- (v) Go back to step (i) to read the next record.

Note: Nothing indicates the end of records processing here.

4. Prepare an algorithm that prints name and weekly wages for each employee out of 10 where name, hours worked, and hourly rate are read in. Draw the flowchart.

Solution

Algorithm

- (i) Initialise Counter (A) to 1
- (ii) Read name, hours and rate and number of workers (iii) Let the wage be assigned the product of hours and rate (iv) Print name and wages.
- (iii) Increment the counter (A) by 1.
- (iv) Make a decision (Check how many times you have calculated the wages).

(v) Stop processing, if you have done it the required number of times.

Flowchart

YES

NO

START

A 1

INPUT NAME HRS,

RATE

WA
GES

HRS X RATE

A

A + 1

IS

A <= 10

PRINT NAME WAGES

STOP

3.6 Pseudocodes

A pseudocode is a program design aid that serves the function of a flowchart in expressing the detailed logic of a program. Sometimes a program flowchart might be inadequate for expressing the control flow and logic of a program. By using Pseudocodes, program algorithms can be expressed as English-language statements. These statements can be used both as a guide when coding the program in a specific language and as documentation for review by others. Because there is no rigid rule for constructing pseudocodes, the logic of the program can be expressed in a manner that does not conform to any particular programming language. A series of structured words is used to express the major program functions. These structured words are the basis for writing programs using a technical term called “structure programming”.

Example

Construct a pseudocode for the problem in the example above.

```
BEGIN
STORE 0 TO SUM
STORE 1 TO COUNT
    DO WHILE COUNT not greater than 10
        ADD COUNT to SUM
    INCREMENT COUNT by 1
ENDWILE
END
```

3.7 Decision Tables

Decision tables are used to analyse a problem. The conditions applying in the problem are set out and the actions to be taken, as a result of any combination of the conditions arising are shown. They are prepared in conjunction with or in place of flowcharts. Decision tables are a simple yet powerful and unambiguous way of showing the actions to be taken when a given set of conditions occur. Moreover, they can be used to verify that all conditions have been properly catered for. In this way they can reduce the possibility that rare or unforeseen combinations of conditions will result in confusion about the actions to be taken.

Decision tables have standardised formats and comprise of four sections.

- (a) **Conditions Stub:** This section contains a list of all possible conditions which could apply in a particular problem.
- (b) **Conditions Entry:** This section contains the different combination of the conditions, each combination being given a number termed a 'Rule'.
- (c) **Action Stub:** This section contains a list of the possible actions which could apply for any given combinations of conditions.
- (d) **Action Entry:** This section shows the actions to be taken for each combination of conditions. **Writing the instructions in a programming language (program coding)**

The instructions contained in the algorithm must be communicated to the computer in a language it will understand before it can execute them. The first step is writing these instructions in a programming language (program coding). Program coding is the process of translating the planned solution to the problems, depicted in a flowchart, pseudocode or decision table, into statements of the program. The program flowchart,

pseudocode decision table as the case may be, is as a guide by the programmer as he describes the logic in the medium of a programming language. The coding is usually done on coding sheets or coding forms.

4.0 CONCLUSION

Flowcharts, decision tables, pseudocodes and algorithms are essential ingredients to the writing of good programs. If they are done properly they lead to a reduction in errors in programs. They help minimise the time spent in debugging. In addition, they make logic errors easier to trace and discover.

5.0 SUMMARY

This unit teaches that:

- a. A flowchart is a graphical representation of the major steps of work in process. It displays in separate boxes the essential steps of the program and shows by means of arrows the directions of information flow.
- b. Decision tables are used to analyse a problem. The conditions applying in the problem are set out and the actions to be taken, as a result of any combination of the conditions arising, are shown.
- c. A pseudocode is a program design aid that serves the function of a flowchart in expressing the detailed logic of a program.
- d. An algorithm is a set of rules for carrying out calculation either by hand or a machine.

6.0 TUTOR-MARKED ASSIGNMENT

1. Draw the flowchart of the program which prints each two-digit odd number N, its square, and its cube.
2. Draw a flowchart to input the scores of a student in 8 courses and find the average of the scores.

7.0 REFERENCES/FURTHER READING

Akinyokun, O.C. (1999). *Principles and Practice of Computing Technology*. Ibadan: International Publishers Limited.

Balogun, V.F., Daramola, O.A., Obe, O.O. Ojokoh, B.A. and Oluwadare S.A., (2006). *Introduction to Computing: A Practical Approach*. Akure: Tom-Ray Publications.

Francis Scheid (1983). *Schaum's Outline Series: Introduction to Computer Science*. Singapore: McGraw-Hill Book Company.

Chuley, J.C. (1987). *Introduction to Low Level Programming for Microprocessors*. Macmillan Education Ltd.

Gray, S. Popkin and Arthur H. Pike (1981). *Introduction to Data Processing with BASIC* (2nded). Boston: Houghton Mifflin Company.