UNIT 2 BASIC PRINCIPLES OF COMPUTER PROGRAMMING

CONTENTS

1.0 INTRODUCTION

Computer programming is both an art and a science. In this unit you wil be exposed to some arts and science of computer programming, including principles of programming and stages of programming.

2.0 OBJECTIVES

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

- state the principles of computer programming
- explain the stages involved in writing computer programs.

3.0 MAIN CONTENT

3.1 Problem Solving With the Computer

The computer is a general-purpose machine with a remarkable ability to process information. It has many capabilities, and its specific function at any particular time is determined by the user. This depends on the program loaded into the computer memory being utilised by the user.

There are many types of computer programs. However, the programs designed to convert the general-purpose computer into a tool for a specific task or applications are called “Application programs”. These are developed by users to solve their peculiar data processing problems. Computer programming is the act of writing a program which a computer can execute to produce the desired result. A program is a series of instructions assembled to enable the computer to carry out a specified procedure. A computer program is the sequence of simple instructions into which a given problem is reduced and which is in a form the computer can understand, either directly or after interpretation.

3.4 Programming Methodology

Principles of Good Programming

It is generally accepted that a good computer program should have the characteristics shown below:

- **Accuracy:** The program must do what it is supposed to do correctly and must meet the criteria laid down in its specification.
- **Reliability:** The program must always do what it is supposed to do, and never crash.
- **Efficiency:** Optimal utilisation of resources is essential. The program must use the available storage space and other resources in such as way that the system speed is not wasted.
- **Robustness:** The program should cope with invalid data and not stop without an indication of the cause of the source of error.
- **Usability:** The program must be easy enough to use and be well documented.
- **Maintainability:** The program must be easy to amend, having good structuring and documentation.
- **Readability:** The code of a program must be well laid out and explained with comments.

3.3 Stages of Programming

The preparation of a computer program involves a set of procedure.
These steps can be classified into eight major stages, viz

(i) Problem definition
(ii) Devising the method of solution
(iii) Developing the method using suitable aids, e.g. pseudo code or flowchart.
(iv) Writing the instructions in a programming language (v) Transcribing the instructions into “machine sensible” form
(v) Debugging the program
(vi) Testing the program
(vii) Documenting all the work involved in producing the program.

(i) **Problem definition**

The first stage requires a good understanding of the problem. The programmer (i.e. the person writing the program) needs to thoroughly understand what is required of a problem. A complete and precise unambiguous statement of the problem to be solved must be stated. This will entail the detailed specification which lays down the input, processes and output required.

(ii) Devising the method of solution

The second stage involved is spelling out the detailed algorithm. The use of a computer to solve problems (be it scientific or business data processing problems) requires that a procedure or an algorithm be developed for the computer to follow in solving the problem.

(iii) Developing the method of solution

There are several methods for representing or developing methods used in solving a problem. Examples of such methods are: algorithms, flowcharts, pseudo code, and decision tables.

(iv) Writing the instructions in a programming language

After outlining the method of solving the problem, a proper understanding of the syntax of the programming language to be used is necessary in order to write the series of instructions required to get the problem solved.

(v) Transcribing the instructions into machine sensible form

After the program is coded, it is converted into machine sensible form or machine language. There are some manufacturers written programs that translate users programs (source programs) into machine language (object code). These are called translators and instructions that machines can execute at a go, while interpreters accept a program and execute it line-by-line.

During translation, the translator carries out syntax check on the source program to detect errors that may arise from wrong use of the programming language.

(vi) Program debugging

A program seldomly executes successfully the first time. It normally contains a few errors (bugs). Debugging is the process of locating and correcting errors. There are three classes of errors.

- **Syntax errors:** Caused by mistake coding (illegal use of a feature of the programming language).

- **Logic errors:** Caused by faulty logic in the design of the program. The program will work but not as intended.

- **Execution errors:** The program works as intended but illegal input or other circumstances at run-time makes the program stop. There are two basic levels of debugging. The first level called desk checking or dry running is performed after the program has been coded and entered or key punched. Its purpose is to locate and remove as many logical and clerical errors as possible.

The program is then read (or loaded) into the computer and processed by a language translator. The function of the translator is to convert the program statements into the binary code of the computer called the object code. As part of the translation process, the program statements are examined to verify that they have been coded correctly, if errors are detected, a series of diagnostics referred to as an error message list is generated by the language translator. With this list in the hand of the programmer, the second level of debugging is reached.

The error message list helps the programmer to find the cause of errors and make the necessary corrections. At this point, the program may contain entering errors, as well as clerical errors or logic errors. The programming language manual will be very useful at this stage of program development.

After corrections have been made, the program is again read into the computer and again processed by the language translator. This is repeated over and over again until the program is error-free.

(vii) Program testing

The purpose of testing is to determine whether a program consistently produces correct or expected results. A program is normally tested by executing it with a given set of input data (called test data), for which correct results are known.

For effective testing of a program, the testing procedure is broken into three segments.

a. The program is tested with inputs that one would normally expect for an execution of the program.

b. Valid but slightly abnormal data is injected (used) to determine the capabilities of the program to cope with exceptions. For example, minimum and maximum values allowable for a salesamount field may be provided as input to verify that the program processed them correctly.

c. Invalid data is inserted to test the program's error-handling routines. If the result of the testing is not adequate, then minor logic errors still abound in the program. The programmer can use any of these three alternatives to locate the bugs.

Other methods of testing a program for correctness include:

- **Manual walk-through:** The programmer traces the processing steps manually to find the errors, pretending to be the computer, following the execution of each statement in the program, noting whether or not the expected results are produced.

- **Use of tracing routines:** If this is available for the language, this is similar to (1) above but is carried out by the computer; hence it takes less time and is not susceptible to human error.

- **Storage dump:** This is the printout of the contents of the computer's storage locations. By examining the contents of the various locations, when the program is halted, the instruction at which the program is halted can be determined. This is an important clue to finding the error that caused the halt.

- **Program documentation:** Documentation of the program should be developed at every stage of the programming cycle. The following are documentations that should be done for each program.

(a) Problem Definition Step

- A clear statement of the problem
- The objectives of the program (what the program is to accomplish)
- Source of request for the program.
- Person/official authorising the request

(b) Planning the Solution Step

- Flowchart, pseudo code or decision tables
- Program narrative
- Descriptive of input, and file formats

(c) Program source coding sheet
(d) User's manual to aid persons who are not familiar with the program to apply it correctly. The manual it contains a description of the program and what it is designed to achieve.

(e) Operator's manual to assist the computer operator to successfully run the program. This manual contains:

(i) Instructions about starting, running and terminating the program.
(ii) Message that may be printed on the console or VDU (terminal) and their meanings.
(iii) Setup and take down instruction for files.

Advantages of Program Documentation

a. It provides all necessary information for anyone who comes in contact with the program.
b. It helps the supervisor in determining the program's purpose, how long the program will be useful and future revision that may be necessary.
c. It simplifies program maintenance (revision or updating).
d. It provides information as to the use of the program to those unfamiliar with it.
e. It provides operating instructions to the computer operator.

4.0 CONCLUSION

The intelligence of a computer derives to a large extent from the quality of the programs. In this unit, we have attempted to present, in some detail, the principles and the stages involved in writing a good computer program.

5.0 SUMMARY

We have discussed the following:

- Principles of computer programming
- Stages of computer programming
- The interrelationship between different stages of programming.

6.0 TUTOR-MARKED ASSIGNMENT

1. Differentiate between program debugging and program testing.
2. What are the differences between syntax errors and logic errors? Give examples of each.
3. Is it possible to detect logic errors during program compilation?
   Explain the reason for your answer.

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.

Holmes, B.J. (1989). *Basic Programming* (3rd ed.) ELBS.

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