TOPIC 7 – PROGRAMMING OVERVIEW

LEARNING OUTCOMES

By the end of this topics, you will be able to:

1. define the features of high-level programming languages and program editors.
2. describe the concept of program translation, compilers, interpreters and assemblers.
3. explain the linking, loading and debuggers.

INTRODUCTION

This topic explains the features of high-level programming languages, program editors, concept of program translation, compilers, interpreters and assemblers, linking, loading and debuggers.

7.1 FEATURES OF HIGH-LEVEL PROGRAMMING LANGUAGES

Mostly, in computer architecture, contemporary programming is carried out using high-level languages. It has the following characteristics:

- Require translation
- Portable
- Easier to read, write and maintain as commands are similar to English
- Allow access to module libraries
- Use data types and data structures, selection statements and repetition/iteration constructs
- Use logic operators and functions that are built into the language

SELF CHECK 7.1

1 List THREE (3) characteristics of high-level programming language.

7.2 PROGRAM EDITORS

The program editors or text editors are software programs that enable the user to create and edit text files. In the programming, the program editor usually refers to source code editors that include many special features for writing and editing code. The Notepad and Wordpad are common program editors on Window operating system and vi, emacs, Jed, pico are the program editors on UNIX operating system. The text editors' features are moving the cursor, deleting, replacing, pasting, finding, finding and replacing, saving etc.

SELF CHECK 7.2

1 Define the function of program editors.

2 List an example of text editor on Window and Unix operating systems.

7.3 THE CONCEPT OF PROGRAM TRANSLATION

There is a translator software in random access memory (RAM) during the program execution. The translator software will take the code written in a high-level language and translate it into an executable code. The executable code is in binary form and can be understood by the processor. In high level programming language, there is no one to one conversion between code and binary instructions.

There are two forms of translator software; compiler or an interpreter.

SELF CHECK 7.3

1. What is program translation?

7.4 COMPILERS

The compiler will translate the source code into an object code. It is held in random access memory (RAM) only for the time that it takes to translate the source code into object code. The compilers will translate lines one after the other however they do not execute a line immediately after translation. Instead, the object code file can be used to run the program after translation. The run time is more efficient when the object code creation by compiler however if there are errors in the source code, the programmer will be responsible to re-compile for the corrected code.

SELF CHECK 7.4

1. Explain how the compilers work?

7.5 INTERPRETERS

The translation with interpreter will translate each line of code one line at a time. The code will be executed once the line of code is translated and if there is error, the interpreter will not move on to translate the next line. The notification on the error will be sent immediately to user. The interpreter is less efficient than a compiler because it is only presenting in main memory during the program execution and for the execution of every line only. The benefit of using an interpreter is that the execution will stop immediately if the interpreter translates a line of source code that results in an error. It will help the programmers to identify and rectify the errors. It cannot determine the actual error but can indicate the line that was not translated into executable machine code.

SELF CHECK 7.5

1. What are interpreters?
2. Describe the benefit of using interpreters over the compilers.

7.6 ASSEMBLERS

Assemblers are the program that translates assembly program (input) into object code (output).

The assembler is a software utility that takes an assembly program as input and produces object code as output. The assembler views this file as a block of memory starting at relative location 0.

There are two general approaches of assemblers; Two-pass Assembler (more common) and One pass Assembler.

In the first pass of two-assembler, the assembler is only concerned with label definitions which construct a symbol table that consists of label and location counter values. The second pass reads the program again from the beginning. The, each instruction is translated into the appropriate binary machine code.

It is possible to implement as assembler that makes only a single pass through the source code however the main difficulty in trying to assemble a program in one pass involves forward references to labels. The instruction operands may be symbols that have not yet been defined in the source program

SELF CHECK 7.6

1. List TWO (2) types of assemblers.
2. Explain the assembler listed in Question 1.

7.7 LINKING AND LOADING

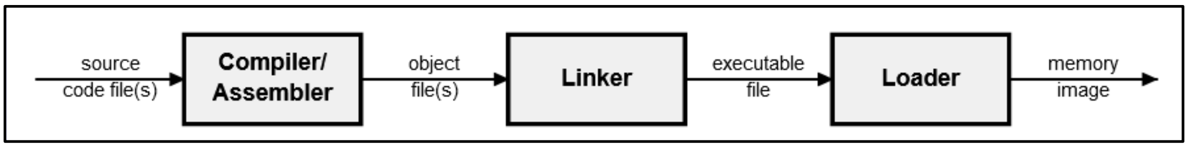

Figure 7.43

A linker or link editor (Figure 7.43) is a software that supporting separate compilation which requires operating system software to combine the code from multiple compilation steps. It produces an executable file from several object files, relocates separately compiled code segments and resolves external references.

The object files are the result of compiling single source code files which contain data and text sections like executable files. The start address is omitted from all object files except the one containing the main program. It also contains symbol tables for resolution of external references and relocation tables for code addresses that need to be relocated.

The loading is a process that performed by loader (Figure 7.43) which takes an executable file and copies its sections into memory. Then the process control block is produces by loader to control program execution. Finally, it starts executing the code by jumping to its main address. The loader must be able to set up text and initialized data in memory, initialize register copies in the process control block and initialize the PC copy in the process control block.

SELF CHECK 7.7

1. Explain the function of linker and loader.
2. What is the file that produce by linker?

7.8 DEBUGGERS

In programming, the debugger is a program or tool that can run other program one line at a time. The debugger can show exactly how the computer **s**ees the code. The debugger shows what is happening in the program at any moment in time, and allows to perform step by step through the program. Proper use of the debugger is essential to finding semantic (logical) errors in how the program behaves.

SELF CHECK 7.8

1. How the debuggers work in the program?

SUMMARY

In this topic you have learnt that:

The features of high-level programming languages, program editors, concept of program translation, compilers, interpreters and assemblers, linking, loading and debuggers.

KEY TERMS

Object Code	The machine language representation of programming source code. Object code is created by a compiler or assembler and is then turned into executable code by the linker
Linker	A utility program that combines one or more files containing object code from separately compiled program modules into a single file containing loadable or executable code.
Loader	A program routine that copies an executable program into memory for execution.

REFERENCES

Stallings, W., (2019). *Computer Organization and Architecture Designing for Performance.* 11th ed. New York: Pearson.