PROGRAMMING

Unit Overview

By the end of this unit, the student will be able to define the term computer program and explain the process by which programs are developed. The student will also identify types of programming languages and describe their differences.

Key Terms

Computer Program: A computer program is a group of instructions that dictate the specific tasks that the computer will perform.	**Program Development Life Cycle:** The program development life cycle (PDLC) is a six-step process for developing computer programs.
Programming Language: A programming language is a set of keywords, punctuations, and spellings that instruct the computer how to complete tasks.	**Syntax Errors:** An error in the spelling or grammar of the computer language.
Source Program: The program that contains the code of an assembly language.	**Semantic Error:** An error in logic that occurs when a command that does not make sense is entered into a program.

What is a Computer Program?

A ***computer program*** is a group of instructions that dictate the specific tasks that the computer will perform. The instructions contained in the computer program are performed in a step-by-step sequential order. Usually, a computer program is saved in the computer’s memory making it accessible to the computer.

How Computer Programs are Written?

There are three different phases of program development. They are the ***problem solving phase, the implementation phase, and the maintenance phase***.

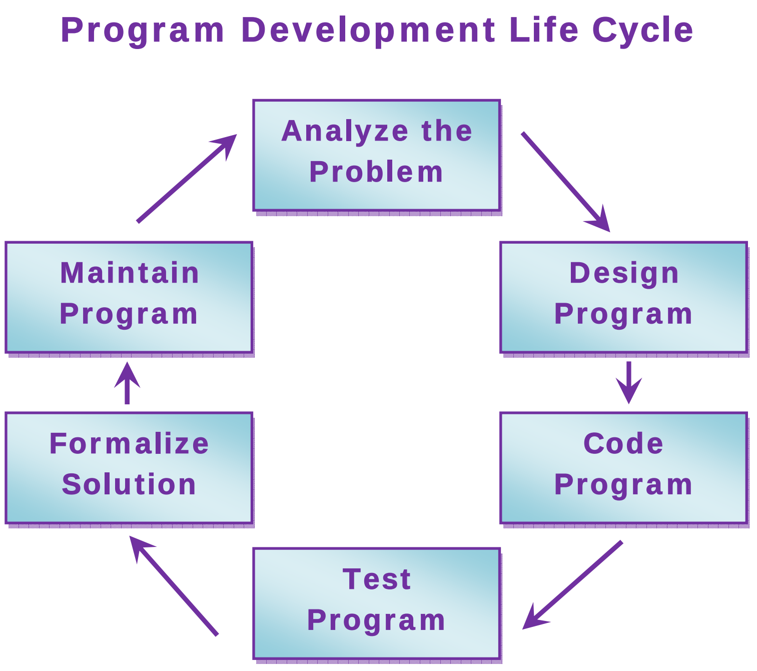
During the ***problem solving phase***, computer programmers define the problem, think of possible solutions to the problem, develop a set of steps to solve the problem, and then test the steps to see if they work.

In the ***implementation phase***, procedures are converted into a programming language that is then entered into the computer. The program is analyzed to identify and correct errors. Then the program documentation and code are evaluated for final approval.

In the ***maintenance phase***, the program continues to be evaluated for errors and is modified based on changing requirements or user preferences.

Program Development Life Cycle

The ***program development life cycle (PDLC)*** is another way to describe the development of computer programs. This six step process is in alignment with the three implementation phases listed above. Read and compare the six steps of the PDLC with the three phases of program development.



Step 1: Analyze the Problem
During this step, the programmer usually meets with the systems analyst and users so he can thoroughly define the problem and its solution. This enables the programmer to define the program specifications and identify the program's input, output, and processing components.

Step 2: Design Program
In this step, programmers develop modules or sets of program activities. Then a formula is designed for each module. Finally, the formulas or algorithms are tested to see if they work.

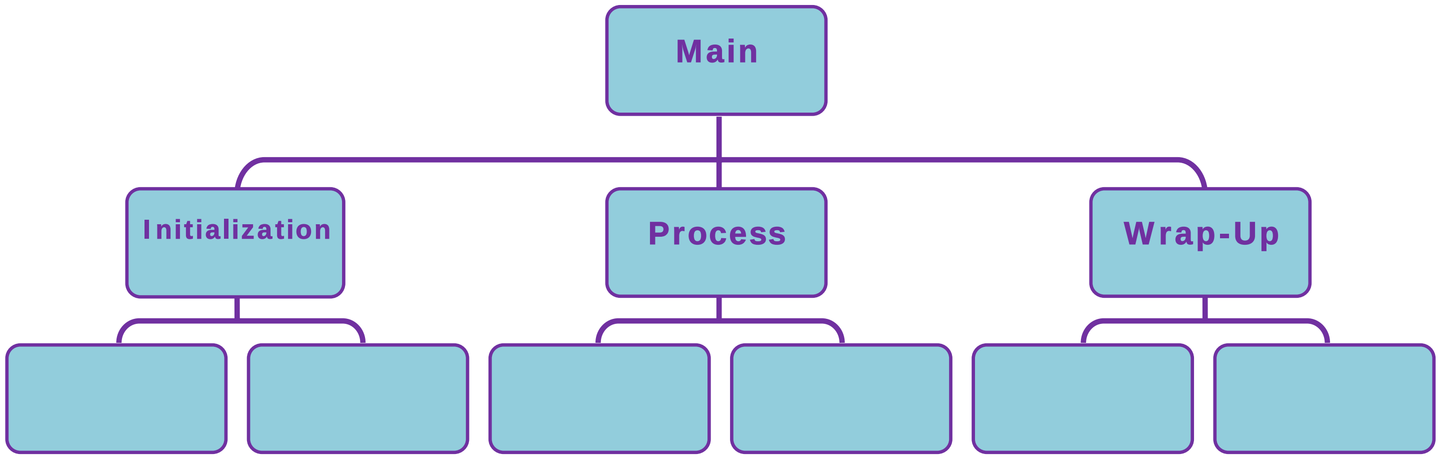
Step 3: Code Program
In this step, the formulas or algorithms are converted into a programming language that is then entered into the computer.

Step 4: Test Program
In this step, the program is evaluated and any errors in grammar (syntax) or logic (semantic) are corrected.

Step 5: Formalize Solution
The program documentation and code is evaluated and then the program is run once more to be sure it is functioning properly.

Step 6: Maintain Program
The program continues to be evaluated for errors and is modified or enhanced based on changing requirements or user preferences.

Top-Down Design Hierarchy Chart



Elements of Program Design

In step 2 of the PDLC, the program design is developed. During this process, a ***top-down design*** is used to clarify the program specifications by breaking them into smaller, more manageable units. The program continues to be broken down until each section of the program dedicated to performing a single function is identified (module). Programmers use a ***hierarchy chart*** to identify the main program activity and its subordinate components.

Structured design is the method for determining the order in which the computer will carry out the procedures that have been developed for each module. The three basic control structures are:

1) Sequence control structure represents the order of actions
2) Selection control structure represents the action to be taken
3) Repetition structure represents repeated actions

What is a Programming Language?

A ***programming language*** is a set of keywords, punctuations, and spellings that instruct the computer how to complete tasks. Each programming language has its own exclusive keywords and format for communicating with the computer.

Types of Programming Languages

Programming languages are classified as ***machine language, assembly language, third generation languages (3GL), fourth-generation languages (4GL), and natural languages***. These groups are further divided into two categories: ***low-level languages and high-level languages.***

Low-level languages include machine language and assembly language. These programming languages can only be run on the computer for which it was designed.

- ***Machine language*** is comprised of only numbers. Machine language has the ability to be read directly by the computer.

- ***Assembly Language*** uses symbols, abbreviations, and codes to communicate instructions to the computer. In order for the computer to understand assembly language it must be translated into machine language.

High-level languages include third generation languages, fourth generation languages, and natural languages. These program languages can be used on many different computers, simplify the process of writing programs, and use terminology that is closely related to English.

- ***Third Generation Languages (3GL)*** uses English-like words and arithmetic operators to tell the computer what needs completed and how to perform tasks to generate the desired result. For this reason, third generation language is sometimes called procedural language. Like assembly language, it must be translated into machine language for the computer to execute the program

- ***Fourth Generation Languages (4GL)*** is a non-procedural language that uses English-like words to convey to the computer what needs to be accomplished. These program languages are extremely user friendly and can be utilized by computer users with virtually no programming experience.

- ***Natural Language*** is a computer language that does not use a definite structure or precise regulations for communicating with the computer. This programming language most closely resembles our everyday patterns of speech.

How Programming Languages are Translated

As mentioned in the previous section, most programming languages have to be ***translated*** into machine language in order to be understood and utilized by the computer. There are a variety of programs that are used to translate programming languages including an ***assembler, a compiler, or an interpreter***.

An ***assembler*** is a computer program that translates the source program of an assembly language into machine language.

A ***compiler*** is used to translate the source program of a third-generation language into machine language. Using a compiler, the whole source program is translated at once. The end result of the compilation is called an ***object program***.

An ***interpreter*** also translates the source program of a third-generation language into machine language. Unlike the compiler, the interpreter translates one line of code at a time.

Popular Programming Languages

Name	Type of Language	Primary Use
BASIC	Interactive	Business applications
FORTRAN	high-level	Scientific applications
C	procedural	Operating Systems and applications
C++	object-oriented	Applications software
JavaScript	interpreted programming language	Website development
C	procedural	Operating Systems and applications