Week 02

Stages of Computer Programming:

1. A*nalyzing and defining the problem*:

   In some organizations, programmers receive asset of specifications from system analysts. In others, the programmers meet directly with users to analyze the problem and determine the user's needs.

   In either case, the task of problem definition involves determining the input and output.

   Finally, you produce a written agreement that specifies, in detail, the input data, the required output, and the processing required converting the input into the output.

2. *Program design or planning the solution*:

   The next step is to design an *algorithm*, a detailed, step-by-step solution to the problem. An algorithm must have some characteristics: must be precise (unambiguous); must be effective; must have a finite number of instructions; execution must always terminate.

   There are number of design tools that a programmer can use to develop the algorithm: *flowchart, pseudo code, decision trees, decision tables, structure diagrams* etc.

   After completing the algorithm design, the programmer should perform a process, called *desk-checking* or *dry-run*, to verify that it produces the desired result. This involves sitting down with a pencil and paper and 'playing computer' by carrying out each step of the algorithm in the indicated sequence.

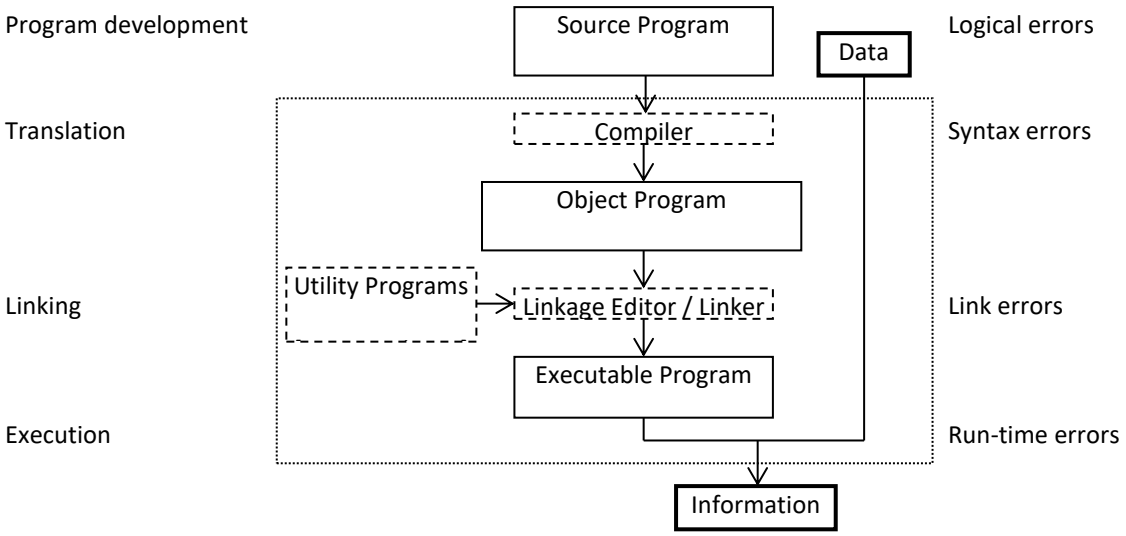
3. *Program coding*:

   Program coding means conversion of the algorithm into a set of instructions written in a computer programming language.

   This program is keyed into the computer by using a *text editor*, a program that is similar to a word processor.

4. *Program testing and debugging*:

   The following diagram illustrates the steps in preparing a program for execution. During these steps computer programs should be checked for program errors.



The original program written in a computer programming language is called as *source program*.

Initially, this source program is translated into its machine language form (called *object program)* by the language translator (*compiler*). If the source program violates the grammatical rules of that programming language, it cannot be converted into its machine language and these types of errors can be described as *syntax errors*, *compile time errors*, or *diagnostic errors*.

As data cannot be processed without some utility programs, object programs should be linked with them to produce an *executable program (load module)*, which can be executed by the computer. This job is done by another system program called a *linkage editor (linker)*.

When required utility programs are not available or when utility programs cannot be linked with the object program properly, an executable program cannot be produced. These errors are described as *link errors*.

When the executable program cannot be executed, such errors can be described as *run-time errors* or *execution-time errors*.

When an executable program is executed and if it produces undesirable results it is due to some errors in the program logic or algorithm. Such errors are described as *logical errors*. Programmers call these errors *bugs*, and the process of locating and correcting them is referred to as *debugging* the program.

If syntax, link or run-time errors are available computer can detect those errors, but the logical error should be detected by computer programmers.

During program testing, different types of test data (both valid and invalid) should be used as input data.

5. *Program implementation*:

   If program testing provides satisfactory results, the executable program can be used to handle data processing activities of the organization in the real environment. This is called *program implementation*.

6. *Program maintenance and update*:

   Program maintenance means to make simple corrective actions in order to run the computer program to meet data processing requirements of the organization. Generally software maintenance agreements can be signed or in-house maintenance personnel can be used for program maintenance.

   Program updating means making structured changes to the existing programs or introducing a new set of programs to meet new data processing requirements of the organization.

7. *Program documentation*:

   Program documentation means to keep or record information related to all the activities of the computer programming.

   Although documentation appears as the last step in the programming process, it is actually performed throughout the entire process.

   Documentation consists of materials generated during each step. E.g.: problem analysis report, algorithm, program listing, test reports, maintenance and update information, user manuals and training materials.

The Characteristics of a Good Computer Program:

1. *Reliability*: The program should provide correct results at all times and should be free from errors.
2. *Maintainability*: The existing program should be able to change or modify to meet new requirements.
3. *Portability*: The program should be able to transfer to a different computer system.
4. *Readability*: The program must be readable and understandable with the help of documentation.
5. *Performance*: The program should handle the task more quickly and efficiently.
6. *Storage saving*: The program should be written with the least number of instructions.