

# Chapter 2

## Problem Solving Using Computer

Software development process involves a systematic process which when not applied may turn computer into a dangerous machine. Software is just a collection of computer programs and programs are a set of instructions. These instructions need to be well specified so as to be directed only towards solving the problem. After its creation, the software should be fault free and well documented. Software development is the process of creating such software, which satisfies end user's requirements and needs. The following six steps must be followed to develop software:

1. Problem Analysis
2. Algorithm Development and Flowchart (Program Design)
3. Coding
4. Compilation and Execution
5. Debugging and Testing
6. Program Documentation

### 1. Problem Analysis

Problem analysis is the process of defining a problem. If a problem is well defined, then half the problem is solved. This task is further subdivided into six subtasks namely.

- i. Specifying the objective of the program
- ii. Specifying the outputs
- iii. Specifying the input requirements
- iv. Specifying the processing requirements
- v. Evaluating the feasibility of the program
- vi. Documenting the program analysis

#### i. Specifying the objective of the program

To avoid having the right solution to the wrong problem, we need to know what the problem actually is. Making a clear statement of the problem depends upon the size and complexity of the problem. Smaller problems not involving multiple subsystems can easily be stated and then we can move onto the next step of "Program Design". However, a problem interacting with various subsystems and series of programs require complex analysis, in-depth research and careful coordination of people, procedures and programs.

#### ii. Specifying outputs

Before knowing what needs to go into the system, we need to what comes out of the system. The best way to specify output is to prepare some output forms and required format for displaying information. The best person to judge an output form is the end user of the system i.e. the one who uses the software to his benefit. Various forms may be designed by the programmer which must be examined to see that they are useful and have neither too much nor too little detail.

#### iii. Specifying the input requirements

After having determined the outputs, the input and data need to be specified as well. One needs to list the inputs required and the source of data. For example in a students' record program, the inputs

could be the student's name, id, roll-numbers, etc. The sources could be the students themselves or the supervisors.

#### **iv. Specifying the processing requirements**

Now that the output and inputs are specified, the process of converting the specified inputs into desired output must be determined. If the proposed program is to replace or supplement an existing one, a careful evaluation of the present processing procedures needs to be made, noting any improvements that could be made. If the proposed system is not designed to replace an existing system, then it is well advised to carefully evaluate another system that addresses a similar problem.

#### **v. Evaluating the feasibility of the program**

After the completion of all the above four steps one needs to see whether the things accomplished so far in the process of software development are practical and feasible. To replace an existing system one needs to determine how the potential improvements outweigh the costs and possible problems.

#### **vi. Documenting the program analysis.**

Before concluding the program analysis stage, it is best to record whatever has been done so far in the first phase of program development. The record should contain the statement of program objectives, output and input specifications, processing requirements and feasibility.

### **2. Algorithm and Flow Chart (Program Design)**

The second stage in software development cycle is the stage of program design. This stage consists of preparing algorithms, flowcharts and pseudo codes. Basically this stage intends to make the program more user friendly, feasible and optimized. Programmer just requires a pen and pencil in this step in which the jobs are first converted into a structured layout without the involvement of computer.

In structured programming, a given task is divided into number of sub-tasks which are termed as modules. Each process is further divided until no further divisions are required. This process of dividing a program into modules and then into sub-modules is known as "top down" design approach. Dividing a program into modules(functions) breaks down a given programming task into small, independent and manageable tasks. The key to structured programming is modularity with single entry and exit point. An unconditional jump using GOTO violates the notion of structured programming.

#### **Algorithms**

Algorithms are written form of the program i.e. a description of instructions carried out to solve a given task. For instance preparing tea requires the following steps to be performed.

- Start
- Fetch water and tea leaves along with sugar and milk
- Boil the water
- Put tea leaves and sugar in boiled water
- Mix with milk with tea
- Serve the tea
- Stop

**Basic guidelines to prepare algorithms**

- Use plain language
- Do not use any language specific syntax. One must be able to code the algorithm in any programming language.
- Do not assume anything stating everything clearly and explicitly.
- Ensure each algorithm has a single entry and exit point.

*An algorithm should be*

**Finite** - It must terminate after finite number of steps

**Definite** - Each step must be well defined

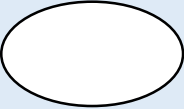


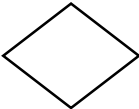
**Inputs** - An algorithm may have many inputs or no inputs at all

**Output** - An algorithm may result in one or many outputs

**Effective** - Operations must be basic so as to be implemented even with the help of pencil & paper.

**Flowcharts**

A flowchart is one of the widely used tools for designing programs. It graphically illustrates the logic needed to solve a problem. In other words it is a diagrammatic representation of an algorithm. The fundamental blocks for drawing flowcharts are:

Structure	Purpose
	Start/Stop
	Inputs/Outputs
	Processing
	Decision

**3. Coding**

In this stage the process of actual program writing takes place. A coded program is most popularly referred to as a source code. The coding process can be done in any language (high level and low level). The actual use of computer takes place in this stage in which the programmer writes a sequence of instructions ready for execution. Here are some qualities of a good program.

- Comment clauses in the program help to make the program readable and understandable by people other than the original programmer
- It should be efficient.
- It must be reliable enough to work under all reasonable conditions to provide a correct output

- It must be able to detect unreasonable error conditions and report them to the end user or programmer without crashing the system
- It should be easy to maintain and support after installation

#### 4. Compilation and Execution

Generally coding is done in high level language or low level language (assembly language). For the computer to understand these languages, they must be translated into machine level language. The translation process is carried out by a compiler/interpreter (for high level language) or an assembler (for assembly language program). The machine language code thus created can be saved and run immediately or later on.

In an interpreted program, each program statement is converted into machine code before program is executed. The execution occurs immediately one statement at a time sequentially. BASIC is one of the frequently used interpreted language. In contrast to interpreter, a compiler converts given source code into object code. Once an object code is obtained, the compiled programs can be faster and more efficient than interpreted programs.

##### The Compilation Process

A source code must go through several steps before it becomes an executable program. In the first step the source code is checked for any syntax errors. After the syntax errors are traced out a source file is passed through a compiler which first translates high level language into object code (A machine code not ready to be executed). A linker then links the object code with pre-compiled library functions, thus creating an executable program. This executable program is then loaded into the memory for execution. During the execution, inputs(if any)are asked for by the user and then the outputs are generated accordingly.

Every High level language comes with its own compiler or interpreter. A compiler therefore in itself is a compiled executable program which checks whether the instructions entered are syntactically correct and then translates a source code into object code. Many compilers are available for the same language. For example compilers available for C are TURBO-C, GCC, etc. Even for FORTRAN, several compilers are available for windows and MAC.

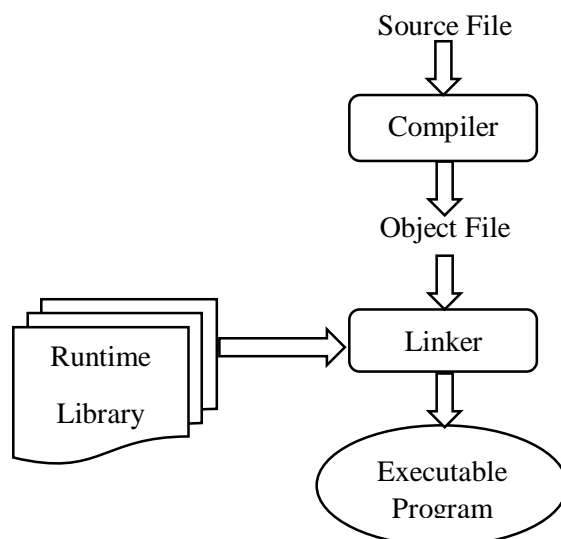


Figure 2.4.1: Compilation Process

## 5. Debugging and Testing

Testing means running the program and fixing it if it does not work i.e. debugging the program (getting the errors or bug out) as called by programmers. There are two types of bugs/errors syntax errors and logical errors.

Syntax error is a violation of programming rules while writing it. A syntax error is easily detected during the compilation process which does not allow the code to be run. A logic error occurs when a programmer has used an incorrect calculation or left out a programming procedure. A program will still work but fails to produce a correct result if a logic error occurs. A number of methods are available for remove syntax or logical errors.

**Structured walkthroughs** - Structured walkthrough process is a formal way of checking a program for errors in which the overall documentation and design of the program is tested. In this process the coded program is submitted to a team of programmers as well as the one who produced the code. After review of the program, the overall completeness, accuracy and quality of the design are discussed. More often a program is put through desk checking in which the entire program is reviewed i.e. the programmers check the printouts, each line statement by statement for syntax and logical errors. The program is further checked manually running sample sets of data manually through the program. More than just checking the program errors, structured walkthrough process is also an attempt to standardize the design and coding process.

**Checking for syntax errors** - After the structured walkthrough process, a programmer attempts to compile and run the program. If any syntax errors may have slipped through structured walkthrough phase, they will be identified by the translator (compiler or interpreter).

**Checking for logical errors** - After the program is checked for syntax errors, the source code is successfully translated into machine language and tested for logical errors. The program is tested for logical correctness, by using a set of sample input data. Then it is checked whether the sample input data produces the expected results or not. If desired results are obtained for each sample data, then the program can be assumed to be logically correct.

It is important to capture syntax and logical errors as early as possible. The cost associated with correcting these errors, increase exponentially the more later they are discovered.

## 6. Program Documentation

The documentation process is the process of collecting information about the program. It starts from the first phase of the program development cycle. Without proper documentation it is very difficult even for the original programmer to update and maintain the program.

A programmers' documentation contains the necessary information that a programmer requires to update and maintain the program. These information include

- Program analysis document, with a concise statement of program's objectives, outputs and processing procedures.
- Program design documents with appropriate flowcharts and diagrams.
- Program verification documents for outlining, checking, testing and correction procedures along with the list of sample data and results.
- Log used to document future program revision and maintenance activity.

User documentation is required for the end user who installs and uses the program.

