Lecture 7.3

Topics

1. A (Simplified) Problem Solving Process in Computer Programming

---

1. A (Simplified) Problem Solving Process in Computer Programming

In general, the program development can be depicted in the following figure.

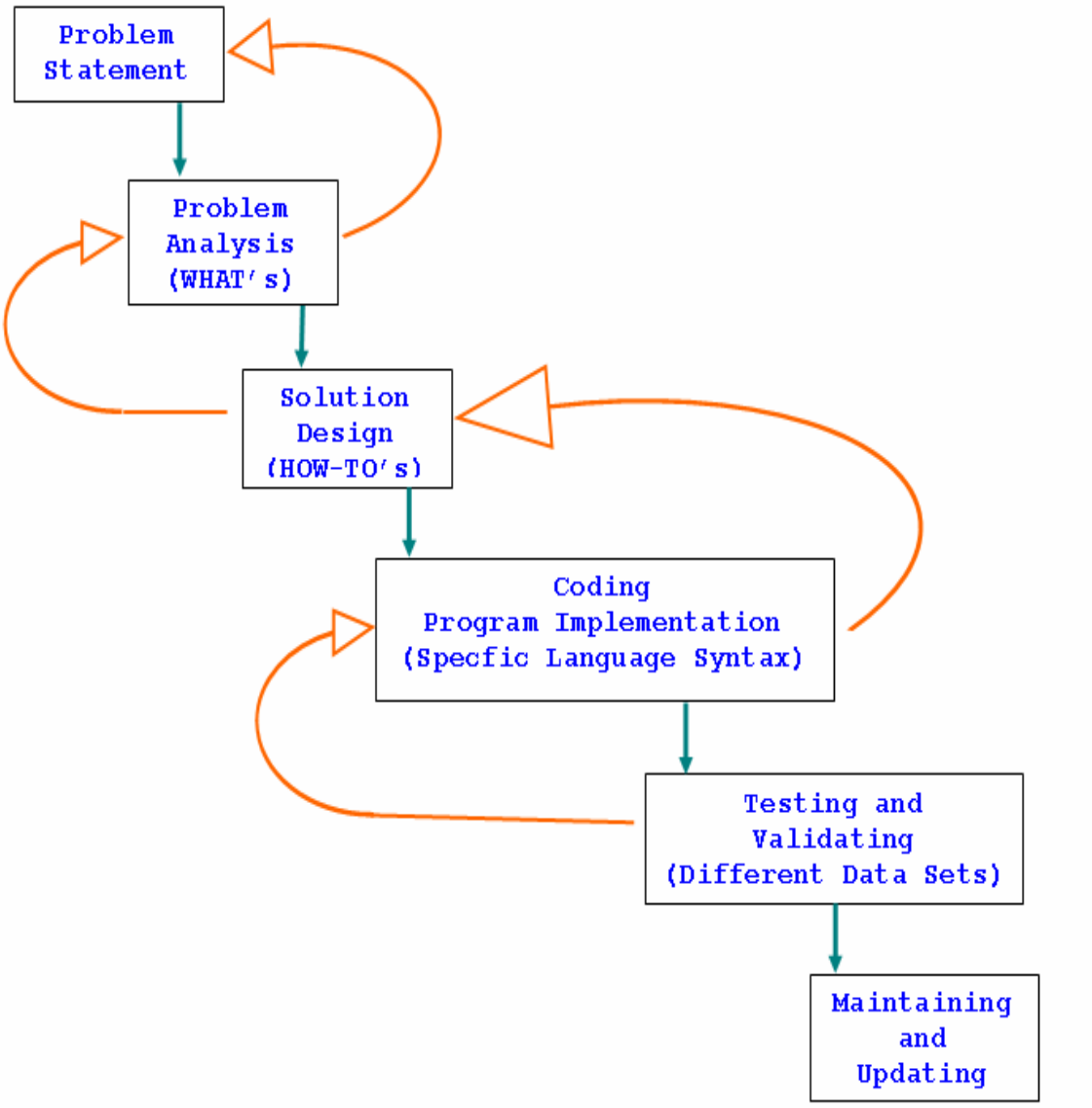

Figure 1 Program Development Process

However, a simplified version or description can be looked at in the following digression.

When doing/coding your assignment (and any work), the following items and points will be considered:

(1) Problem Statement

Each assignment must have the problem statement (from the assignment sheet or rephrasing the assignment properly). The problem statement may be precise or "a bit" vague based on the complexity of the problem.

(2) Analysis

A rough analysis must be provided. It must conform to the nature of problem statement, the underlined assumption and hypothesis.

The analysis will provide the answers for the questions of "what we have/what are the input", "what are required", “what are the assumptions/constraints”, and "what are the objectives/output".

(3) Algorithm & Design

The design step will detail HOW-TO do or perform the tasks (i.e., “WHAT-NEED-TO-BE-DONE” that are obtained through the analysis).

A set of logical steps (or TASKS) leading toward the solution may also be called the algorithm.

There are several ways to describe a design:

- A flowchart,
- Structure chart (with execution flow for each function), or
- Other functional/object-oriented diagrams (UML).

(4) Implementation Approach

The implementation should have the ***synopsis*** for each method/function – **Input, Output**, and **What the function does**. One might want to specify/indicate the mathematical expressions or techniques being used.

The program should work and produce the results as required. You should provide comments as needed throughout the code.

If your program DOES NOT WORK then you must provide some insight and hint on why it does not work. To show a “GOOD” effort, you should propose a possible fix to each of the errors or bugs of the program.

Note that , at the very least, you must be consistent with the coding style when writing code. You can get the hints from me, from books, or from anyone else. Or, you can develop and justify your own style; with that, keep the style consistent throughout the program.

Hint! Read the article on “Programming Convention” that was handed out in class.

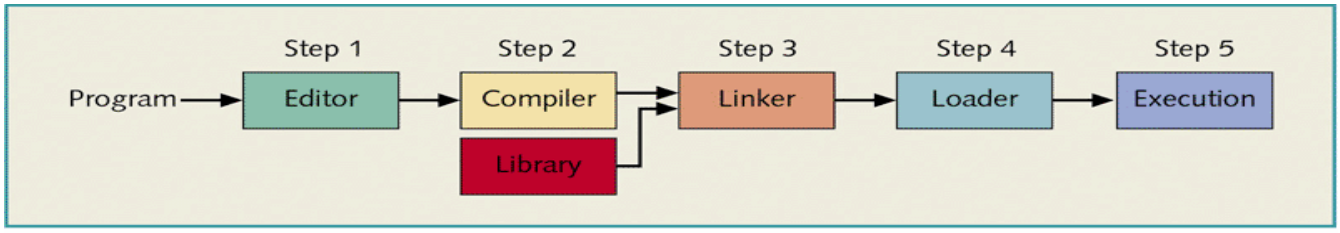

Figure 2 Creating and processing a high-level language program

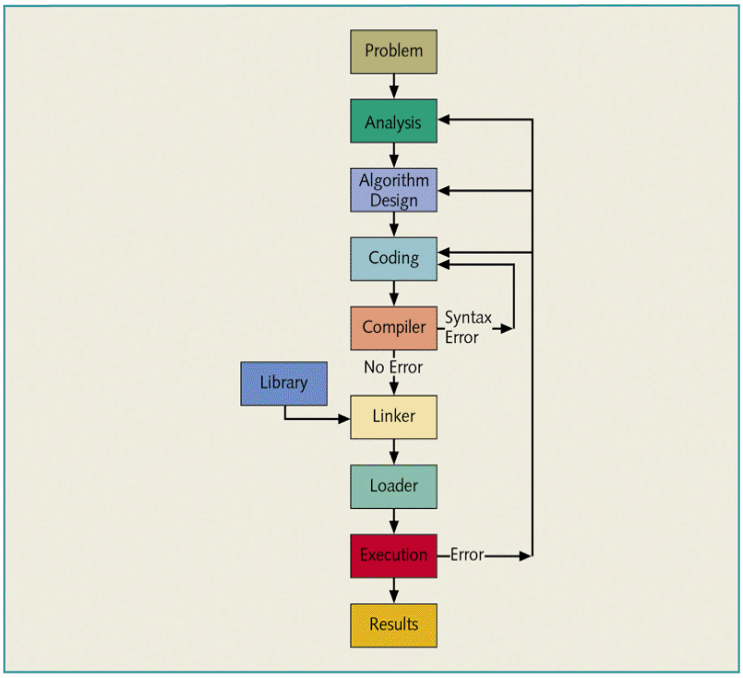

Figure 3 Development process -- Problem analysis-design-coding-execution cycle

Example -- Finding the Smallest Value

Problem Statement

Write a program to determine the smallest value from a given sequence of values. As the user enters values, the program will keep track of the smallest value entered so far. The program will print this smallest value when the user enters **`eof (ctrl+z)`**.

Analysis:

What do we have?
- A sequence of values

What do we want?
- The smallest value

What do we assume about the input values?
- To be numeric and entered from keyboard

Design:

How to get the smallest value?

Read in first value
- Assume this first value is the smallest one

Read in next value
- Compare with the previous smallest value
- Swap, if necessary, and store new smallest value

Read in another value
- Compare with the previous smallest value
- Swap, if necessary, and store new smallest value

Keep repeating the process of read/compare/swap until `ctrl+z`.