PF-FALL-2022

Lecture-3 outline (30th August 2022)

Topic 1: Problem solving

GOAL: "Become a good Problem Solver"

Concept: **The process of finding a solution of any problem/complex problem in an efficient and effective way**.

- Finding an ***effective*** solution of a problem.
- Important feature of a scientist (computer scientist, Cyber security Expert)
- Problem solving skill is a learning skill can improve with practice, awareness and hardship.

Steps involved in problem solving:

Usually, four steps but can be vary.

1) **Identification of problem**: (Super important step)
    a) Defining the problem: (Thinking Step)
        - Foremost and very Important to define and understanding the problem. Define "what, when, how, where, who and why" about the problem.
        - Clarity about problem.
    b) Define goals, output and objectives for solving problem
    c) Find limitations/constraints and dependencies of problem.
    d) Brainstorming.
    e) Sometime also, consult with expert in that area and explore state of the art regarding that problem.
2) **Designing the solution**: (Thinking Step)
    - Appropriate plan, model or design of the solution.
    - It defines the complete flow or steps from start to end to solve the problem.
    - Device some algorithm: It gives a ***step-by-step*** model/solution to solve a problem
        - Divide a problem into small steps. Those steps should be:
            - ▪ Appropriate
            - ▪ Meaningful
            - Concrete
            - Accurate
            - Sequence

        - Efficient/Fast
        - Scalable
        - Finite steps
        - Atomic steps
    - We can test an algorithm (DRY RUN)
    - Helpful in creating/clearing concepts and modules of a complete system

3) **Build/implement the solution**:

- Actual implementing the design/plan and creating an actual a working prototype.
- Codding (implementing the solution in a programming language to communicate it with the computer).

4) **Test and Debug**:

- **Test:** Finding that our implemented solution meets requirements and goals defined in step 1.
- Testing is very rigorous mechanism and is a complete field (Quality assurance, Tester or **Penetration testing** is used)
- If our solution is not working according to requirements as defined then it is debugged (that is look back and find out where is the actual problem is).
- **Debug:** Debugging is actually removing all mistakes and error from the solution to make it work properly.

Topic 2: Algorithm vs Flow-chart

Flow chart is pictorial representation of solution that represents a flow and plan of solution. A flowchart can also be defined as a diagrammatic representation of an algorithm, a systematic approach to solving a task.

Some examples of algorithms and flow charts.

Topic 3: Computer Programs

Concept: **Computers can do many different jobs because they are programmable. Moreover, we have to write effective programs for them.**

A program is a set of instructions a computer follows in order to perform a task. A programming language is a special language used to write computer programs.

When a program executes, at lower level it changes the electrical impulses and alter the internal states of computer. At higher level, more abstract level

computer users accomplish real world problems or tasks to derive actual pleasure.

For some people computer is a calculator, gamming, fun, entertainment and networking tool. For some people it is research, learning and teaching/study tool.

Computer programs are written in programming languages. These are:

1. Low Level Languages (Machine Language): Target or binary code
2. Middle level Language: Assemble code
3. High Level language: Source code

- Assemble code have one to one mapping with Machine code whereas on instruction of high-level language is mapped to more than one instruction of Machine code (or even assembly code)
- We will C++ as our HLL.

Translators: translate source code to machine or object code

- Interpreter: Line by line translation
- Compiler: Translate whole source code to object code/ Machine code
- Assembler
    - Syntax (Grammar or rules of program)
    - Semantics (meaning or Logic of a program)
    - Syntax error: error of grammar or syntax can be detected by translator
    - Semantic error: Error of logic cannot be detected by translator
        - Debugging
    - Interpreted vs Compiled languages
    - General purpose vs Special purpose languages