Lecture – 3 -

C++ is a middle-level programming language developed by Bjarne Stroustrup starting in 1979 at Bell Labs. C++ runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. This C++ course adopts a simple and practical approach to describe the concepts of C++ for beginners.

Why to Learn C++

C++ is a MUST for students and working professionals to become a great programmer. Some of the key advantages of learning C++ can be:

- **C++** is very close to hardware, so you get a chance to work at a low level which gives you lot of control in terms of memory management, better performance and finally a robust software development.
- **C++ programming** gives you a clear understanding about Object Oriented Programming. You will understand low level implementation of polymorphism when you will implement virtual tables and virtual table pointers, or dynamic type identification.
- **C++** is one of the ever-green programming languages and loved by millions of software developers. If you are a great C++ programmer, then you will never sit without work and more importantly you will get highly paid for your work.
- **C++** is the most widely used programming languages in application and system programming. So, you can choose your area of interest of software development.
- **C++** really teaches you the difference between compiler, linker and loader, different data types, storage classes, variable types their scopes, … etc.

Applications of C++ Programming

As mentioned before, C++ is one of the most widely used programming languages. It has it's presence in almost every area of software development. I'm going to list few of them here:

- **Application Software Development** - C++ programming has been used in developing almost all the major Operating Systems like Windows, Mac OSX and Linux. Apart from the operating systems, the core part of many browsers like Mozilla Firefox and Chrome have been written using C++. C++ also has been used in developing the most popular database system called MySQL.
- **Programming Languages Development** - C++ has been used extensively in developing new programming languages like C#, Java, JavaScript, Perl, UNIX's C Shell, PHP and Python, and Verilog etc.
- **Computation Programming** - C++ is the best friends of scientists because of fast speed and computational efficiencies.
- **Games Development** - C++ is extremely fast which allows programmers to do procedural programming for CPU intensive functions and provides greater control over hardware, because of which it has been widely used in development of gaming engines.
- **Embedded System** - C++ is being heavily used in developing Medical and Engineering Applications like software application for MRI machines, high-end CAD/CAM systems etc.

Typical Programming (software development) Method

1. Determine the problem requirements
2. Analyze the problem to be solved
3. Design an algorithm or a flowchart to solve the problem
4. Implement the algorithm by writing your C++ code
5. Compile your program

. If there are compile errors, go back to 4 and debug your code

6. Run your executable, test and verify
   . If the program does not run as expected go back to 2, 3 or 4 as appropriate
7. Maintain and update

Problem Requirements

Specifying the problem requirements: state the problem clearly and unambiguously and to gain a clear understanding of what is required for its solution.

Analyzing the problem: involves identifying:

- all the inputs - the data you have to work with,
- all the outputs - the desired results,
- any additional requirements or constraints.

Design

An algorithm is generally a high level abstraction that is not actually code. However, generally an algorithm expresses the ideas of a program.

Designing the algorithm: develop the actual list of steps (the algorithm) to solve the problem, and then to verify that the algorithm solves the problem as intended.

- Usually the hardest step in the development process.
- Best not to try to solve every detail at the beginning - use top-down design: list the major steps (subproblems) first then solve each subproblem.

Most algorithms consist of at least the following subproblems:

1. Get the (input) data
2. Perform the computations
3. Display the results

Implementation, Testing, and Maintenance

- **Implementing the algorithm:** involves writing your algorithm (pseudo-code) as a program in your chosen programming language.
- **Testing and verifying:** requires thorough testing (on various inputs) to verify that your program works as desired.
- **Maintenance and updating:** as things change over time (ie. TAX increase) a program may need to be updated and maintained

Some concepts used in programming:

Loops: to repeat a statement or group of statements for number of times or as long as a certain condition is true.

Counters: to count things, usually counters need an initial value (could be zero or one or any other value). Some examples: number of students passed in an exam, number of positive integers in 100 random numbers, number of even numbers in 100 random numbers.

Summations: to find the summation (sum) of series of numbers or series of expressions. Some examples: summation of numbers 1-100, summation of $(x-1)^2$.