Introduction to Programming

Dr. Tanmay Basu and Dr. Akash Anil
Dept. of Data Science and Engineering
Indian Institute of Science Education and Research Bhopal

Administrative Details [Lecture 1]

Key Learning Objectives:

- Understanding the requirements of Computer Programming.
- Understand and implement the basic components of a programming language such as conditional statements, loops, arrays, etc.
- Understand the important components and their use cases of C programming language.
- Implement data structures using the C programming language.
- Attain an intermediate level of programming skills in C.

Administrative Details

<u>Timings</u>

- Slot C
  - Monday 10:00 - 10:55
  - Wednesday 09:00 - 09:55
  - Friday 09:00 - 09:55

- Instructors
  - Dr. Akash Anil
  - Dr. Tanmay Basu

Administrative Details

Evaluation

- Quiz: 10%
- Mid Semester Examination (Written): 30 %
- Assignment 10 %
- End Semester Examination (Written): 50 %

Administrative Details

Text Books

- Schaum's Outline of Programming with C by Byron Gottfried, McGraw-Hill India.
- The C Programming Language by Kernighan and Ritchie, Prentice- Hall India.

References

- Programming in ANSI C by Balaguruswamy.
- C: The Complete Reference by Herbert Schildt
- https://archive.nptel.ac.in/courses/106/104/106104128/
- https://archive.nptel.ac.in/courses/106/105/106105085/

Administrative Details

Syllabus - I

- **Module 1: Introduction:** Why Computer Programming, Programming Languages, Flowchart, The C Programming Language, Components of a simple C program, Data Types, Identifiers, Variables, Constants, Variable Declarations and Assignments, ASCII Codes, Operators and Expressions, Operator's Expressions Associativity and Precedence, Expression Evaluation **[7L]**
- **Module 2: Conditional and Iteration:** If-Else, Conditional Operator (?:), Switch Case, While Loop, Do-While Loop, For Loop, Break and Continue, Goto. **[3L]**
- **Module 3: Functions:** Introduction to Functions, Function Execution, Function Prototyping and Definition, Parameter passing in functions (Call by Value, Call by Reference), Recursion. **[5L]**

Administrative Details

Syllabus - II

- **Module 4: Pointers:** Introduction, Pointer Syntax, Pointer Arithmetic, Pointers to Pointer, Function Pointer, Dynamic Memory Allocation, sizeof **[4L]**
- **Module 5: Array and String:** Array Initialization, Pointers and Arrays, Array of Pointers, Character Arrays and Strings, Multi-dimensional Arrays **[4L]**
- **Module 6: Input and Output:** File Handling in C, Standard IO Functions **[4L]**
- **Module 7: Structure, Union, and Data Structures:** Structure Declarations and Initialization, Referencing Structure and Structure Members, Union Declaration and Initialization, Nested Structure and Union, Singly Linked List, Doubly Linked List. **[7L]**

Administrative Details

Lab Sessions

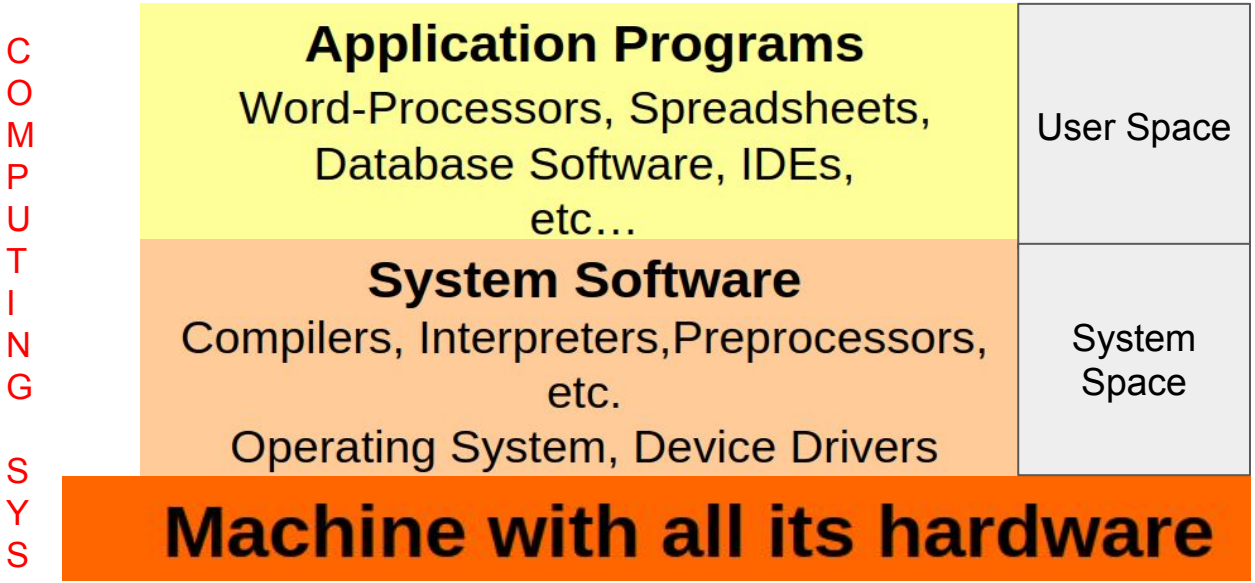
- Lab Practice Sessions are optional.
- Lab examination is mandatory for all.
- You may skip the practice session if you are confident in programming.

Lab Timing

Will be announced shortly…

Why Computer Programming

- Programming: A process of setting instructions for a computer to perform a task.
  - Example *c = add(a,b),*
  - *if c>5 then print "Hello Programmers"*



Learning Objectives [Lecture 2]

- ❑ Types of Programming Languages
- ❑ Execution of a typical program
- ❑ Compiler
- ❑ Interpreter
- ❑ Code demo
- ❑ The C Programming Language: Features
- ❑ Components of a simple C program

Programming Languages

High-Level Language (HLL)

Uses English-like language
Machine independent
Portable (but must be compiled for different platforms)
Examples: Pascal, C++, Java, Fortran, . . .

Assembly Language

Uses mnemonics
Machine-dependent
Not usually portable

Machine Language

Uses binary code
Machine-dependent
Not portable

Machine Language

- The representation of a computer program which is actually read and understood by the computer.
  - A program in machine code consists of a sequence of machine instructions.
- Instructions:
  - Machine instructions are in binary code
  - Instructions specify operations and memory cells involved in the operation

Example:

Operation	Address
0010	0000 0000 0100
0100	0000 0000 0101
0011	0000 0000 0110

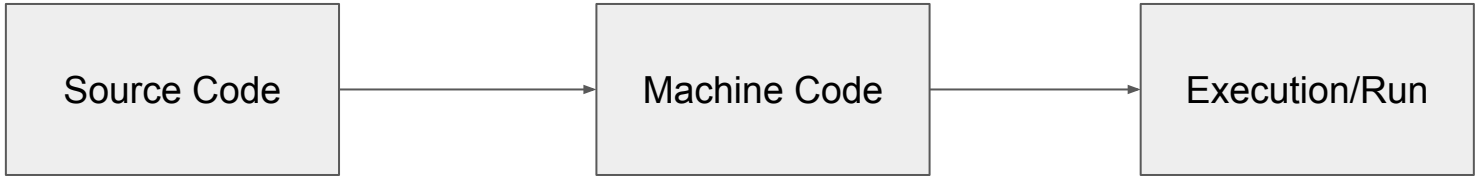
Assembly Language

- A symbolic representation of the machine language of a specific processor.
- Is converted to machine code by an assembler.
- Usually, each line of assembly code produces one machine instruction (One-to-one correspondence).
- Programming in assembly language is slow and error-prone but is more efficient in terms of hardware performance.
- Mnemonic representation of the instructions and data
- Example:
  - Load Price
  - Add Tax
  - Store Cost

High-level language

- A programming language which use statements consisting of English-like keywords such as "FOR", "PRINT" or “IF“, ... etc.
- Each statement corresponds to several machine language instructions (one-to-many correspondence).
- Much easier to program than in assembly language.
- Data are referenced using descriptive names
- Operations can be described using familiar symbols
- Example: Cost := Price + Tax
- HLL Examples: C++, Python, Java, and many more…

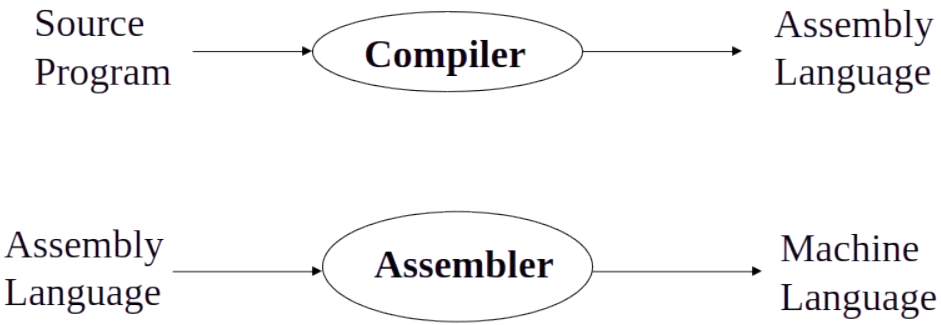
Typical Program Execution



- Some programming languages use two step process [Compiler]
  - First the program is checked for errors and converted to machine code.
  - Now the machine code is executed.
  - Examples: C, C++, Java
- Some programming languages does in one step [Interpreter]
  - Python, Javascript, PHP

Compiler (HLL -> LLL)

- A program that converts another program from some source language (or high-level programming language / HLL) to machine language.
- Some compilers output assembly language which is then converted to machine language by a separate assembler.
- Is distinguished from an assembler by the fact that each input statement, in general, correspond to more than one machine instruction.



If Compilation is successful, the program is executed/run.

Learning Objectives [Lecture 3]

- History of Computing [Video https://www.youtube.com/watch?v=-M6lANfzFsM] [10 Minues]
- History of Programming [Video Interview of **Brian Kernighan** by Lex Fridman https://www.youtube.com/watch?v=KUGtBzws-ic] [6 Minutes]
- The C Programming Language: Features
- Components of a simple C program
- Compiler vs Interpreter [Code Demo over Execution Style]

Interpreter

- A computer program that directly executes a source program.
  - Usually done by implicitly translating the source to intermediate machine language, and executed.

Compiler

- Translates the whole code to low level at once.
- Generates an executable file if compilation is successful and can be run.
- Compiled programs usually run faster.
- No requirement of source code after compilation.

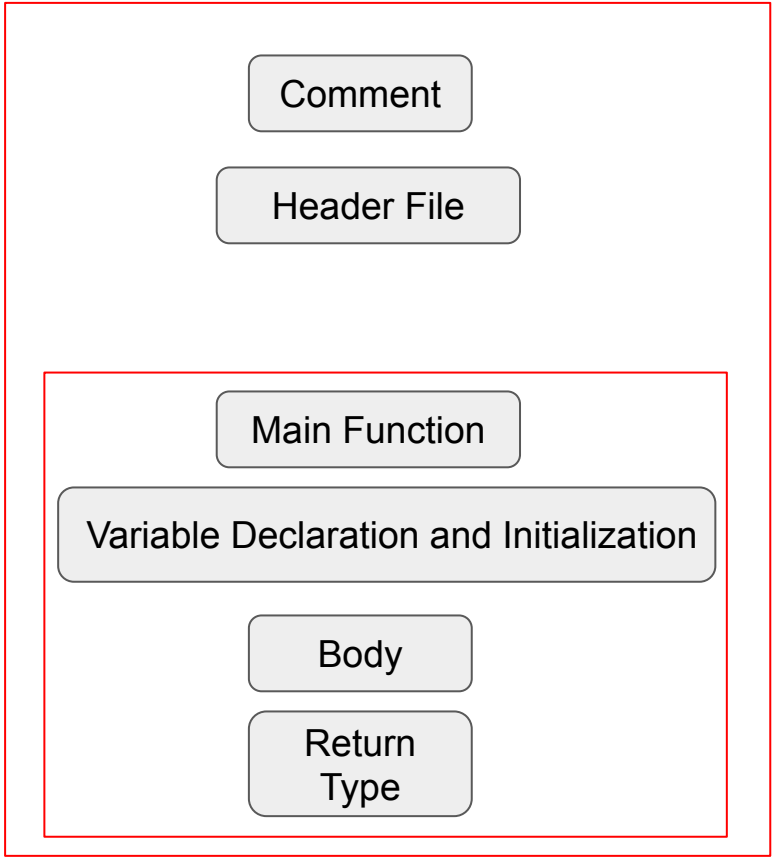
Interpreter

- Translates each statement of the code one by one
- Does not generate any executable. Runs part of the program having no error.
- Slow compared to compiled programs.
- Requires source code each time for running the program.

The C Programming Language: Features

- A Middle Level Programming Language
  - Direct interactions to hardwares like memory and registers.
  - Close to assembly language in terms of efficiency and control
  - Can be used to write operating systems, e.g., UNIX, Linux, Windows
  - High level features such as function, structure ensuring modularity
  - Portable
- Structural and Procedural
  - Using functions, loops, control statements
  - Maintains readability and maintainability
  - Often seen as Top Down approach of programming
- Typed Programming
  - Each variable must have a defined type, easy to debug while compile
- Open Source and Rich library
- Efficient Memory management
  - Functions like Malloc / Calloc provide user more control over memory management.

Components of a simple C Program




```
// Print Happy New Year

#include<stdio.h>




int main(){
        int year=2025;
        printf("Happy New Year %d", year);
        return 0;
}
```