

**Name:** Paramprakash Makwana

## **Module 2 – Introduction to Programming**

2.1) Write an essay covering the history and evolution of C programming. Explain its importance and why it is still used today.

Ans:- C was developed by Dennis Ritchie in 1972 at Bell Labs. It evolved from earlier languages like B and BCPL to provide system-level programming features. It is still used because of its speed, portability, and ability to interact closely with hardware.

2.2) Describe the steps to install a C compiler (e.g., GCC) and set up an Integrated Development Environment (IDE) like DevC++, VS Code, or CodeBlocks.

Ans:- Install GCC using MinGW or TDM-GCC from their official websites. For IDE setup, download and install DevC++, VS Code (with C/C++ extension), or CodeBlocks, then link the compiler path. After configuration, you can write and run C programs easily.

2.3) Explain the basic structure of a C program, including headers, main function, comments, data types, and variables. Provide examples.

Ans:- A C program includes header files (like `stdio.h`), the `main()` function, and statements. It also contains comments for documentation and uses data types and variables to store values.

**Example:** `#include<stdio.h> int main(){ int a=10; printf("%d",a); }.`

2.4) Write notes explaining each type of operator in C: arithmetic, relational, logical, assignment, increment/decrement, bitwise, and conditional operators.

Ans:- Arithmetic operators perform calculations; relational operators compare values; logical operators combine conditions. Assignment updates variables, increment/decrement changes values by 1, bitwise works on bits, and conditional (?:) chooses between expressions.

2.5) Explain decision-making statements in C (if, else, nested if-else, switch). Provide examples of each.

Ans:- `if` checks a condition, `if-else` provides two options, and nested `if-else` handles multiple conditions. `switch` selects a block based on a constant value.  
Example: `if(a>0), switch(choice){ case 1: ... }.`

2.6) Compare and contrast while loops, for loops, and do-while loops. Explain the scenarios in which each loop is most appropriate.

Ans:- while checks the condition first and is used when iterations are unknown. for is ideal for fixed loops with counter control. do-while runs at least once because it checks the condition at the end.

2.7) Explain the use of break, continue, and goto statements in C. Provide examples of each.

Ans:- break exits a loop immediately; continue skips to the next iteration. goto jumps to a labeled statement, mainly used in exceptional cases.

2.8) What are functions in C? Explain function declaration, definition, and how to call a function. Provide examples.

Ans:- Functions are reusable blocks of code. They have a declaration (prototype), definition (body), and are called using the function name.  
Example: `int sum(int a,int b);` → `int sum(){...}` → `sum(5,3);`.

2.9) Explain the concept of arrays in C. Differentiate between one-dimensional and multi-dimensional arrays with examples.

Ans:- An array stores multiple values of the same type. One-dimensional arrays store data in a single row, while multi-dimensional arrays (like 2D) store data in rows and columns. Example: `int a[5];` `int b[3][3];`.

2.10) Explain what pointers are in C and how they are declared and initialized. Why are pointers important in C?

Ans:- Pointers store memory addresses of variables. They are declared using `*` (e.g., `int *p;`) and initialized using `&`. They are important for dynamic memory, arrays, and efficient programming.

2.11) Explain string handling functions like `strlen()`, `strcpy()`, `strcat()`, `strcmp()`, and `strchr()`. Provide examples of when these functions are useful.

Ans:- `strlen()` finds length, `strcpy()` copies one string to another, `strcat()` joins strings, `strcmp()` compares them, and `strchr()` finds a character. These functions make string manipulation easier.

2.12) Explain the concept of structures in C. Describe how to declare, initialize, and access structure members.

Ans:- A structure groups different data types under one name. Declare using struct, initialize members, and access them using the dot operator.  
Example: `struct student { int id; char name[20]; };`

2.13) Explain the importance of file handling in C. Discuss how to perform file operations like opening, closing, reading, and writing files.

Ans:- File handling allows storing data permanently. Files can be opened with `fopen()`, read with `fscanf()` or `fgets()`, written with `fprintf()`, and closed using `fclose()`. It is essential for backups and data storage.