Module 2 – Introduction to Programming

(THEORY EXERCISE)

1. **Write an essay covering the history and evolution of C programming. Explain its importance and why it is still used today.**  
   C was developed in the early 1970s by Dennis Ritchie at Bell Labs as an evolution of the B language. It’s still widely used today because it’s fast, close to hardware, portable, and forms the base for many modern languages. Its importance lies in system programming, embedded systems, and performance-critical applications.
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**

To set up C, install a compiler like GCC and choose an IDE such as DevC++, Code::Blocks, or VS Code. Install the compiler, configure it in the IDE, and then test by running a simple “Hello, World!” program. This ensures the environment is ready for development.

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

A C program includes header files (e.g., #include ), a main() function, variable declarations, and statements. Comments explain the code, while data types define variable types. The program executes statements inside main() sequentially.

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

C operators include arithmetic (+,-,\*), relational (>,<>), and conditional (?:). They are used for performing calculations, comparisons, and decision-making in programs.

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

Decision-making statements include if, if-else, nested if-else, and switch. They help execute code based on conditions, such as checking values or selecting from multiple options.

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

while loops run while a condition is true, for loops run for a fixed number of iterations, and do-while loops run at least once before checking the condition. Choice depends on whether the number of repetitions is known beforehand.

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

Break exits a loop early, continue skips the current iteration, and goto jumps to a labeled part of code. They provide control over the normal flow of loops.

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

Functions are reusable code blocks with a declaration, definition, and call. They improve code organization, reduce repetition, and can return values or perform actions.

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

Arrays store multiple elements of the same type. One-dimensional arrays are simple lists, while multi-dimensional arrays (like 2D) store data in grids or tables.

1. **Explain what pointers are in C and how they are declared and initialized. Why are pointers important in C?**

Pointers store memory addresses of variables and are declared with \*. They allow direct memory access, efficient data handling, and dynamic memory management.

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

String handling functions like strlen() (length), strcpy() (copy), strcat() (concatenate), strcmp() (compare), and strchr() (find character) simplify text processing. They are useful for formatting, searching, and manipulating text.

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

Structures group different data types under one name. They are declared with struct, initialized with values, and accessed using the dot (.) operator.

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

File handling allows programs to store and retrieve data permanently. Common operations are opening, reading, writing, and closing files using functions like fopen(), fread(), fprintf(), and fclose()