Basics of C Programming

1. Syntax and Keywords

- **Syntax**: The rules and structure to write valid C programs.
  - Every C statement ends with a semicolon ;.
  - Blocks of code are enclosed within curly braces {}.
  - C is case-sensitive (e.g., Main and main are different).
- **Keywords**: Reserved words that have a predefined meaning in C and cannot be used as identifiers (e.g., int, return, if).
  - Example keywords: int, float, char, double, if, else, for, while, return, void, switch, case, struct, etc.

2. Data Types

Primitive Data Types

1. **int**: Used for integer numbers (e.g., 1, 100, -50).
   - Example: int age = 25;
2. **float**: Used for single-precision floating-point numbers (e.g., 3.14, -0.99).
   - Example: float pi = 3.14;
3. **char**: Used for single characters (e.g., 'A', 'b').
   - Example: char grade = 'A';
4. **double**: Used for double-precision floating-point numbers (more accurate than float).
   - Example: double distance = 12345.678;

Derived Data Types

1. **Arrays**: A collection of elements of the same data type.
   - Example: int numbers[5] = {1, 2, 3, 4, 5};
2. **Pointers**: Variables that store the memory address of another variable.
   - Example: int *ptr = &number;
3. **Structures**: A user-defined data type that groups related variables of different types.
   - Example:
   - struct Student {
   - int id;
   - char name[50];
   - float marks;
   - };

3. Variables and Constants

Declaration and Initialization

- **Variable Declaration**: Allocating memory for a variable and specifying its type.
  - Syntax: data_type variable_name;
  - Example: int age;
- **Variable Initialization**: Assigning an initial value to the variable.
  - Syntax: data_type variable_name = value;
  - Example: int age = 25;

Scope and Lifetime

- **Scope**:
  - Local: Declared within a block and accessible only inside it.
  - Global: Declared outside all functions and accessible throughout the program.
- **Lifetime**:
  - Automatic: Variables are created when the block is entered and destroyed when it exits.
  - Static: Variables retain their value even after the block ends.

4. Input and Output

printf()

- Used to display output on the screen.
- Syntax: printf("format string", variable_list);
- Example:
- int age = 25;
- printf("Age: %d", age);
- Common format specifiers:
  - %d for integers
  - %f for floats
  - %c for characters
  - %s for strings

scanf()

- Used to take input from the user.
- Syntax: scanf("format string", &variable);
- Example:
- int age;
- printf("Enter your age: ");
- scanf("%d", &age);
- Always use & (address-of operator) for variables in scanf().
