

THEORY EXERCISES – C++ NOTES

1. Procedural Programming vs Object-Oriented Programming (OOP)

Procedural Programming (POP) is based on functions and procedures, where the program is divided into small parts called functions. Data is mostly global and can be accessed by any function. Security is low, and real-world modeling is difficult.

Object-Oriented Programming (OOP) is based on objects and classes. Data is encapsulated inside objects, making programs more secure and reusable. It supports concepts like inheritance, polymorphism, and abstraction.

Advantages of OOP over POP

1. Better data security through encapsulation.
2. Code reusability through inheritance.
3. Easy maintenance and modification.
4. Better real-world modeling using objects.
5. Modular structure improves readability.

Setting up a C++ Development Environment

1. Install a C++ compiler like GCC or MinGW.
2. Install an IDE such as Code::Blocks, Dev-C++, or Visual Studio.
3. Create a new C++ project or file.
4. Write your C++ program.
5. Compile and run the program to check output.

Input/Output in C++

Input is done using `cin`, and output is done using `cout`.

Example:

```
int a;
cout << "Enter a number: ";
cin >> a;
cout << "You entered: " << a;
```

Different Data Types in C++

1. `int` – Stores integers (e.g., `int a = 10;`)
2. `float` – Stores decimal numbers (e.g., `float b = 5.5;`)
3. `double` – Stores large decimal numbers.
4. `char` – Stores single character (e.g., `char c = 'A';`)
5. `bool` – Stores true or false.

Implicit vs Explicit Type Conversion

Implicit conversion is done automatically by the compiler, such as converting `int` to `float`.

Explicit conversion is done by the programmer using type casting, such as `(float)a`.

Types of Operators in C++

1. Arithmetic: +, -, *, /, %
2. Relational: >, <, ==, !=
3. Logical: &&, ||, !
4. Bitwise: &, |, ^, <<, >>

Constants and Literals

Constants are values that do not change, declared using const. Literals are fixed values written in the program, like 10 or 3.14.

Conditional Statements in C++

If-else is used for decision making. Switch is used when there are multiple choices based on a variable.

Loops in C++

For loop is used when number of iterations is known.
While loop is used when condition is checked before execution.
Do-while runs at least once before checking condition.

Break and Continue

break stops the loop completely.
continue skips the current iteration and moves to next.

Nested Control Structures

A loop inside another loop is called nested loop. Example: printing patterns.

Functions in C++

A function is a block of code that performs a specific task. It is declared, defined, and called in a program.

Scope of Variables

Local variables are declared inside a function.
Global variables are declared outside and accessible everywhere.

Recursion in C++

When a function calls itself, it is called recursion. Example: factorial calculation.

Function Prototypes

A function prototype is a declaration before main function. It tells the compiler about function name, return type, and parameters.

Arrays in C++

A single-dimensional array is a list of elements.
A multi-dimensional array is like a table or matrix.

String Handling in C++

Strings can be handled using string class or character arrays.

Array Initialization

1D: `int a[3] = {1,2,3};`
2D: `int b[2][2] = {{1,2},{3,4}};`

String Operations

Common functions: `length()`, `substr()`, `find()`, `compare()`.

Key Concepts of OOP

Encapsulation, Inheritance, Polymorphism, Abstraction.

Classes and Objects

Class is a blueprint; object is an instance of a class.

Inheritance

It allows a derived class to reuse properties of a base class.

Encapsulation

Wrapping data inside a class and protecting it using private access.