**…….Module-3, :- Theory…….**

Introduction to OOPS Programming

**Q1. Key Differences: POP vs OOP ?**

Ans:-

| **Feature** | **POP** | **OOP** |
| --- | --- | --- |
| Approach | Top-down | Bottom-up |
| Focus | Functions | Objects |
| Data Access | Global | Private (encapsulated) |
| Security | Low | High |
| Reusability | Poor | High |
| Examples | C | C++, Java |

**Q2. Advantages of OOP over POP ?**

Ans:-

1. **Encapsulation** – Bundles data + methods.
2. **Abstraction** – Hides complex details.
3. **Inheritance** – Reuse code.
4. **Polymorphism** – Same name, different behavior.
5. **Modularity** – Easy to manage and debug.
6. **Reusability & Maintenance** – Better structure and updates.

**Q3. Steps to Set Up C++ Environment ?**

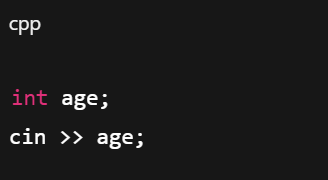
Ans:-

1. Install compiler (e.g., GCC or MinGW).
2. Install IDE (e.g., Code::Blocks, VS Code).
3. Configure PATH (if needed).
4. Write a simple program.
5. Compile and run.

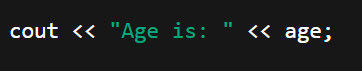
**Q4. Main Input/Output in C++ ?**

Ans:-

* **Input:**



* **Output:**



**Q5. C++ Data Types ?**

Ans:-

* **int → int a = 10;**
* **float → float pi = 3.14;**
* **char → char ch = 'A';**
* **bool → bool flag = true;**
* **double, string,** arrays, pointers, structs, etc.

**Q6. Implicit vs Explicit Conversion ?**

Ans:-

* **Implicit**: Auto by compiler → int a = 4.5;
* **Explicit**: Manual cast → int a = (int)4.5;

**Q7. C++ Operators (with example) ?**

Ans:-

* **Arithmetic**: **+, - → a + b**
* **Relational: ==, < → a == b**
* **Logical**: **&&, || → a && b**
* **Assignment: =, += → x += 2**
* **Increment/Decrement: x++, --y**
* **Bitwise: &, |, ^**

**Q8. Constants & Literals ?**

Ans:-

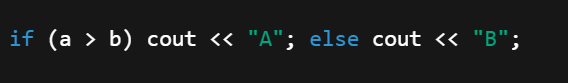
* **Constants**: Fixed value → const int a = 10;
* **Literals**: Actual values → 10, 'A', "Hi", true

### ****Q9. What are conditional statements in C++ ?****

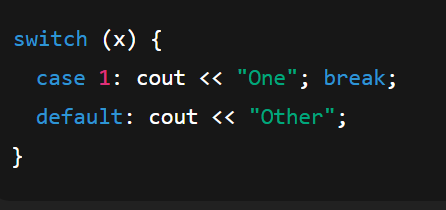
Ans:-

Conditional statements control program flow based on conditions.

* **if-else**: Executes one block if condition is true, another if false.



* **switch**: Selects one case from multiple based on a variable.



### ****Q10. Difference between for, while, and do-while loops ?****

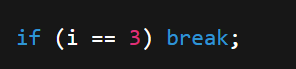
Ans:-

| **Loop** | **Condition Check** | **Use When** | **Runs at least once?** |
| --- | --- | --- | --- |
| for | Before | Known times | No |
| while | Before | Unknown times | No |
| do-while | After | Must run once | Yes |

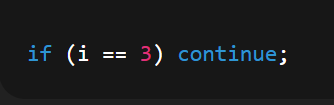
### ****Q11. How are break and continue used in loops ?****

Ans:-

* **break**: Exits the loop early.



* **continue**: Skips current loop step.

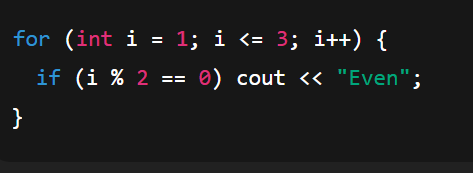


### ****Q12. What are nested control structures ?****

Ans:-

Control statements inside others.

**Example:**



### ****Q13. What is a function in C++ ?****

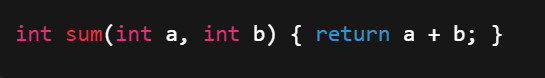
Ans:-

A function is a block of code that performs a task.

* **Declaration**: Tells the compiler.

int sum(int, int);

* **Definition**: Actual code.



* **Calling**: Use the function.

sum(5, 3);

### ****Q14. What is scope of variables in C++ ?****

Ans:-

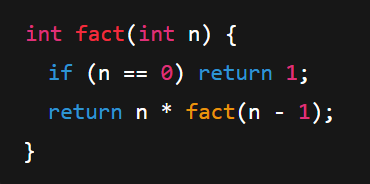
Scope = where a variable is accessible.

* **Local**: Inside a function/block.  
  Only used there.
* **Global**: Outside all functions.  
  Used anywhere in the program.

### ****Q15. What is recursion in C++ ?****

Ans:-

A function calling itself is **recursion**.

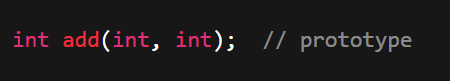
**Example**:

### ****Q16. What is a function prototype in C++ ?****

Ans:-

A prototype tells the compiler about a function **before** it's used.

**Example**:



### ****Q17. What are arrays in C++ ?****

Ans:-

Arrays store **multiple values** of the **same type**.

* **1D array**: Linear

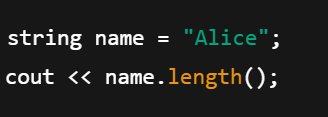
int a[3] = {1, 2, 3};

* **2D array**: Table (rows × columns)

int b[2][2] = {{1, 2}, {3, 4}};

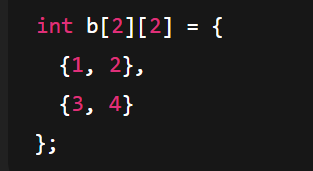
### ****Q18. String handling in C++ ?****

Ans:-

* **C-style string**: **char name[10] = "John";**
* **C++ string class**:

### ****Q19. Array initialization ?****

Ans:-

* **1D**: int a[4] = {10, 20, 30, 40};
* **2D**:

### ****Q20. String operations ?****

Ans:-

| **Operation** | **Function** |
| --- | --- |
| Length | str.length() |
| Add/Join | str1 + str2 |
| Compare | str1 == str2 |
| Substring | str.substr(0, 2) |
| Find | str.find("hi") |

### ****Q21. Key Concepts of OOP ?****

Ans:-

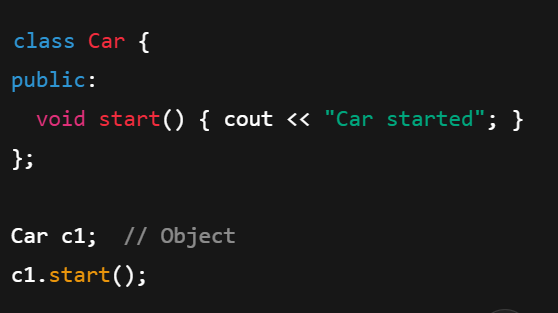
1. **Class** – Blueprint for objects
2. **Object** – Instance of a class
3. **Encapsulation** – Hiding data using classes
4. **Inheritance** – Reuse code from another class
5. **Polymorphism** – One function, many forms
6. **Abstraction** – Hiding complex details

### ****Q22. What are classes and objects in C++?****

Ans:-

* **Class**: User-defined data type
* **Object**: Variable of the class

**Example:**

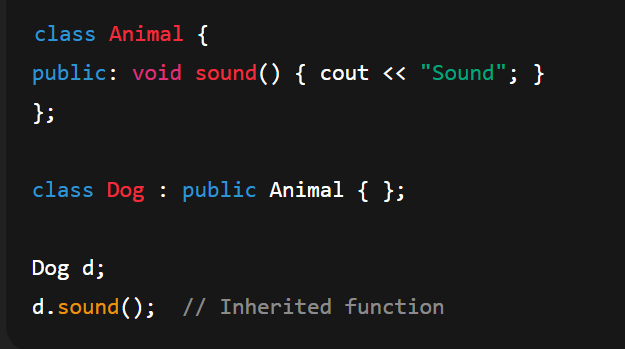


### ****Q23. What is inheritance in C++?****

Ans:-

One class **inherits** features of another.

**Example:**



### ****Q24. What is encapsulation in C++?****

Ans:-

Wrapping data and functions in a class & hiding it from outside.

**Achieved using**:

* **Private** data members
* **Public** methods to access them

**Example:**

