MODULE :-3 INTRODUCTION TO OOPS PROGRAMMING

[LAB PRACTICAL]

1. Introduction to C++

1. First C++ Program: Hello World

- **Write a simple C++ program to display "Hello, World!".**

Solved:-

```cpp
// Hello World Program
#include <iostream>   // Include input/output header
using namespace std;  // Use standard namespace

int main() {
    cout << "Hello, World!";  // Output statement
    return 0;                 // End of program
}
```

Output :-

```
Hello, World!
```

2. Basic Input/Output

- Write a C++ program that accepts user input for their name and age and then displays a personalized greeting.

Solved:-

```cpp
#include <iostream>
using namespace std;

int main() {
    string name;
    int age;

    cout << "Enter your name: ";
    cin >> name;

    cout << "Enter your age: ";
    cin >> age;

    cout << "Hello, " << name << "! You are " << age << " years old." << endl;
    return 0;
}
```

Output :-

```
Enter your name: Jay
Enter your age: 22
Hello, Jay! You are 22 years old.
```

3.POP vs. OOP Comparison Program

o Write two small programs: one using Procedural Programming (POP) to calculate the area of a rectangle, and another using Object-Oriented Programming (OOP) with a class and object for the same task.

Solved :- **(a) POP (Procedural Programming)**

Code:

```cpp
#include <iostream>
using namespace std;

int main() {
    int length, width, area;

    cout << "Enter length: ";
    cin >> length;

    cout << "Enter width: ";
    cin >> width;

    area = length * width;

    cout << "Area of rectangle: " << area << endl;
    return 0;
}
```

Output:-

```
Enter length: 5
Enter width: 3
Area of rectangle: 15
```

◇ **(b) OOP (Object-Oriented Programming)**

Code:

```cpp
#include <iostream>
using namespace std;

class Rectangle {
public:
    int length, width;

    void input() {
        cout << "Enter length: ";
        cin >> length;
        cout << "Enter width: ";
        cin >> width;
    }

    void calculateArea() {
        int area = length * width;
        cout << "Area of rectangle: " << area << endl;
    }
};

int main() {
    Rectangle r1;
    r1.input();
    r1.calculateArea();
    return 0;
}
```

Output :-

```
Enter length: 6
Enter width: 4
Area of rectangle: 24
```

4. Setting Up Development Environment

- Write a program that asks for two numbers and displays their sum. Ensure this is done after setting up the IDE (like Dev C++ or CodeBlocks).

Solved :- **IDE Setup + Sum of Two Numbers**

Code:

```cpp
#include <iostream>
using namespace std;

int main() {
    int num1, num2, sum;

    cout << "Enter first number: ";
    cin >> num1;

    cout << "Enter second number: ";
    cin >> num2;

    sum = num1 + num2;

    cout << "The sum is: " << sum << endl;
    return 0;
}
```

Output :-

```
Enter first number: 15
Enter second number: 25
The sum is: 40
```

2. Variables, Data Types, and Operators

1.Variables and Constants

o Write a C++ program that demonstrates the use of variables and constants. Create variables of different data types and perform operations on them.

Solved:-

```cpp
#include <iostream>
using namespace std;

int main() {
    // Variables
    int age = 20;
    float height = 5.9;
    char grade = 'A';

    // Constant
    const float PI = 3.14;

    // Operations
    int nextYearAge = age + 1;
    float circumference = 2 * PI * 7;  // example circle with radius 7

    // Output
    cout << "Age: " << age << endl;
    cout << "Height: " << height << endl;
    cout << "Grade: " << grade << endl;
    cout << "PI (constant): " << PI << endl;
    cout << "Age next year: " << nextYearAge << endl;
    cout << "Circle circumference: " << circumference << endl;

    return 0;
}
```

Output :-

```
Age: 20
Height: 5.9
Grade: A
PI (constant): 3.14
Age next year: 21
Circle circumference: 43.96
```

2.Type Conversion

- **Write a C++ program that performs both implicit and explicit type conversions and prints the results.**

Solved :-

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 10;
    float b = 5.75;

    // Implicit conversion (int to float)
    float result1 = a + b;

    // Explicit conversion (float to int)
    int result2 = a + (int)b;

    // Output
    cout << "Implicit conversion (10 + 5.75): " << result1 << endl;
    cout << "Explicit conversion (10 + (int)5.75): " << result2 << endl;

    return 0;
}
```

Output :-

```
Implicit conversion (10 + 5.75): 15.75
Explicit conversion (10 + (int)5.75): 15
```

3.Operator Demonstration

o Write a C++ program that demonstrates arithmetic, relational, logical, and bitwise operators. Perform operations using each type of operator and display the results.

Solved :-

-

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 5, b = 2;

    // Arithmetic Operators
    cout << "Addition: " << (a + b) << endl;
    cout << "Subtraction: " << (a - b) << endl;
    cout << "Multiplication: " << (a * b) << endl;
    cout << "Division: " << (a / b) << endl;
    cout << "Modulus: " << (a % b) << endl;

    // Relational Operators
    cout << "a > b: " << (a > b) << endl;
    cout << "a == b: " << (a == b) << endl;

    // Logical Operators
    cout << "(a > 0 && b > 0): " << ((a > 0) && (b > 0)) << endl;
    cout << "(a < 0 || b > 0): " << ((a < 0) || (b > 0)) << endl;
    cout << "!(a == b): " << (!(a == b)) << endl;

    // Bitwise Operators
    cout << "a & b: " << (a & b) << endl;
    cout << "a | b: " << (a | b) << endl;
    cout << "a ^ b: " << (a ^ b) << endl;

    return 0;
}
```

Output :-

```
Addition: 7
Subtraction: 3
Multiplication: 10
Division: 2
Modulus: 1
a > b: 1
a == b: 0
(a > 0 && b > 0): 1
(a < 0 || b > 0): 1
!(a == b): 1
a & b: 0
a | b: 7
a ^ b: 7
```

3. Control Flow Statements

1.Grade Calculator

- **Write a C++ program that takes a student's marks as input and calculates the grade based on if-else conditions.**

Solved :-

```cpp
#include <iostream>
using namespace std;

int main() {
    int marks;
    cout << "Enter your marks (0 to 100): ";
    cin >> marks;

    if (marks >= 90) {
        cout << "Grade: A";
    } else if (marks >= 80) {
        cout << "Grade: B";
    } else if (marks >= 70) {
        cout << "Grade: C";
    } else if (marks >= 60) {
        cout << "Grade: D";
    } else {
        cout << "Grade: F (Fail)";
    }

    return 0;
}
```

Output :-

```
Enter your marks (0 to 100): 78
Grade: C
```

2. **Number Guessing Game**

   o **Write a C++ program that asks the user to guess a number between 1 and 100. The program should provide hints if the guess is too high or too low. Use loops to allow the user multiple attempts.**

Solved:-

```cpp
#include <iostream>
using namespace std;

int main() {
    int secretNumber = 45;
    int guess;

    cout << "Guess the number (1 to 100): ";
    cin >> guess;

    while (guess != secretNumber) {
        if (guess < secretNumber) {
            cout << "Too low! Try again: ";
        } else {
            cout << "Too high! Try again: ";
        }
        cin >> guess;
    }

    cout << "Congratulations! You guessed it right!";
    return 0;
}
```

Output:-

```
Guess the number (1 to 100): 30
Too low! Try again: 60
Too high! Try again: 45
Congratulations! You guessed it right!
```

3. **Multiplication Table**

   o **Write a C++ program to display the multiplication table of a given number using a for loop.**

Solved:-

Code:-

```cpp
#include <iostream>
using namespace std;

int main() {
    int number;
    cout << "Enter a number to display its multiplication table: ";
    cin >> number;

    for (int i = 1; i <= 10; i++) {
        cout << number << " x " << i << " = " << number * i << endl;
    }

    return 0;
}
```

Output :-

```
Enter a number to display its multiplication table: 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
...
5 x 10 = 50
```

4. **Nested Control Structures**

   o **Write a program that prints a right-angled triangle using stars (*) with a nested loop.**

Solved :-

Code:-

```cpp
#include <iostream>
using namespace std;

int main() {
    int rows;
    cout << "Enter number of rows: ";
    cin >> rows;

    for (int i = 1; i <= rows; i++) {
        for (int j = 1; j <= i; j++) {
            cout << "* ";
        }
        cout << endl;
    }

    return 0;
}
```

Output :-

```
Enter number of rows: 5
*
* *
* * *
* * * *
* * * * *
```

4. Functions and Scope

1. Simple Calculator Using Functions

o Write a C++ program that defines functions for basic arithmetic operations (add, subtract, multiply, divide). The main function should call these based on user input.

Solved :-

```cpp
#include <iostream>
using namespace std;

float add(float a, float b) {
    return a + b;
}

float subtract(float a, float b) {
    return a - b;
}

float multiply(float a, float b) {
    return a * b;
}

float divide(float a, float b) {
    if (b != 0)
        return a / b;
    else {
        cout << "Cannot divide by zero!" << endl;
        return 0;
    }
}

int main() {
    float num1, num2;
    char op;

    cout << "Enter first number: ";
    cin >> num1;

    cout << "Enter operator (+, -, *, /): ";
    cin >> op;

    cout << "Enter second number: ";
    cin >> num2;

    switch(op) {
        case '+': cout << "Result: " << add(num1, num2); break;
        case '-': cout << "Result: " << subtract(num1, num2); break;
        case '*': cout << "Result: " << multiply(num1, num2); break;
        case '/': cout << "Result: " << divide(num1, num2); break;
        default: cout << "Invalid operator!";
    }

    return 0;
}
```

Output :-

```
Enter first number: 10
Enter operator (+, -, *, /): *
Enter second number: 3
Result: 30
```

2. Factorial Calculation Using Recursion

o Write a C++ program that calculates the factorial of a number using recursion.

Solved :-

```cpp
#include <iostream>
using namespace std;

int factorial(int n) {
    if (n == 0)
        return 1;
    else
        return n * factorial(n - 1);
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;

    if (num < 0)
        cout << "Factorial not defined for negative numbers.";
    else
        cout << "Factorial of " << num << " is " << factorial(num);

    return 0;
}
```

Output :-

```
Enter a number: 5
Factorial of 5 is 120
```

4. **Variable Scope**
   o Write a program that demonstrates the difference between local and global variables in C++. Use functions to show scope.

Solved :-

```cpp
#include <iostream>
using namespace std;

int globalVar = 100; // Global variable

void showLocalScope() {
    int globalVar = 50; // Local variable with same name
    cout << "Inside function, local variable: " << globalVar << endl;
}

int main() {
    cout << "Global variable before function: " << globalVar << endl;
    showLocalScope();
    cout << "Global variable after function: " << globalVar << endl;
    return 0;
}
```

Output :-

```
Global variable before function: 100
Inside function, local variable: 50
Global variable after function: 100
```

5.Arrays and Strings

1. Array Sum and Average

o Write a C++ program that accepts an array of integers, calculates the sum and average, and displays the results.

Solved :-

```cpp
#include <iostream>
using namespace std;

int main() {
    int numbers[5], sum = 0;
    float average;

    cout << "Enter 5 integers: ";
    for (int i = 0; i < 5; i++) {
        cin >> numbers[i];
        sum += numbers[i];
    }

    average = sum / 5.0;

    cout << "Sum = " << sum << endl;
    cout << "Average = " << average << endl;

    return 0;
}
```

Output :-

```
Enter 5 integers: 10 20 30 40 50
Sum = 150
Average = 30
```

2. Matrix Addition

o Write a C++ program to perform matrix addition on two 2x2 matrices.

Solved :-

```cpp
#include <iostream>
using namespace std;

int main() {
    int a[2][2], b[2][2], sum[2][2];

    cout << "Enter elements of first 2x2 matrix:\n";
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            cin >> a[i][j];

    cout << "Enter elements of second 2x2 matrix:\n";
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            cin >> b[i][j];

    // Matrix addition
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            sum[i][j] = a[i][j] + b[i][j];

    // Output result
    cout << "Resultant Matrix (Sum):\n";
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            cout << sum[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

Output :-

```
Enter elements of first 2x2 matrix:
1 2
3 4
Enter elements of second 2x2 matrix:
5 6
7 8
Resultant Matrix (Sum):
6 8
10 12
```

3. String Palindrome Check

o Write a C++ program to check if a given string is a palindrome (reads the same forwards and backwards).

Solved :-

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    string str, reversed = "";
    cout << "Enter a string: ";
    cin >> str;

    // Reverse the string
    for (int i = str.length() - 1; i >= 0; i--) {
        reversed += str[i];
    }

    // Check if palindrome
    if (str == reversed) {
        cout << str << " is a palindrome." << endl;
    } else {
        cout << str << " is not a palindrome." << endl;
    }

    return 0;
}
```

Output :-

```
Enter a string: madam
madam is a palindrome.
```

6.Introduction to Object-Oriented Programming

1. Class for a Simple Calculator

o Write a C++ program that defines a class Calculator with functions for addition, subtraction, multiplication, and division. Create objects to use these functions.

Solved :-

```cpp
#include <iostream>
using namespace std;

class Calculator {
public:
    float add(float a, float b) {
        return a + b;
    }

    float subtract(float a, float b) {
        return a - b;
    }

    float multiply(float a, float b) {
        return a * b;
    }

    float divide(float a, float b) {
        if (b != 0)
            return a / b;
        else {
            cout << "Cannot divide by zero!" << endl;
            return 0;
        }
    }
};

int main() {
    Calculator calc;
    float x = 10, y = 5;

    cout << "Add: " << calc.add(x, y) << endl;
    cout << "Subtract: " << calc.subtract(x, y) << endl;
    cout << "Multiply: " << calc.multiply(x, y) << endl;
    cout << "Divide: " << calc.divide(x, y) << endl;

    return 0;
}
```

Output :-

```
Add: 15
Subtract: 5
Multiply: 50
Divide: 2
```

2. Class for Bank Account

o Create a class BankAccount with data members like balance and member functions like deposit and withdraw. Implement encapsulation by keeping the data members private.

Solved :-

```cpp
#include <iostream>
using namespace std;

class BankAccount {
private:
    float balance;

public:
    BankAccount() {
        balance = 0;
    }

    void deposit(float amount) {
        if (amount > 0) {
            balance += amount;
            cout << "Deposited: " << amount << endl;
        }
    }

    void withdraw(float amount) {
        if (amount <= balance) {
            balance -= amount;
            cout << "Withdrawn: " << amount << endl;
        } else {
            cout << "Insufficient balance!" << endl;
        }
    }

    void showBalance() {
        cout << "Current Balance: " << balance << endl;
    }
};

int main() {
    BankAccount acc;
    acc.deposit(1000);
    acc.withdraw(400);
    acc.showBalance();

    return 0;
}
```

Output:-

```
Deposited: 1000
Withdrawn: 400
Current Balance: 600
```

3. Inheritance Example

o Write a program that implements inheritance using a base class Person and derived classes Student and Teacher. Demonstrate reusability through inheritance.

Solved :-

```cpp
#include <iostream>
using namespace std;
class Person {
public:
    string name;
    int age;

    void getDetails() {
        cout << "Enter name: ";
        cin >> name;
        cout << "Enter age: ";
        cin >> age;
    }
    void showDetails() {
        cout << "Name: " << name << ", Age: " << age << endl;
    }
};

class Student : public Person {
public:
    void study() {
        cout << name << " is studying." << endl;
    }
};
class Teacher : public Person {
public:
    void teach() {
        cout << name << " is teaching." << endl;
    }
};
int main() {
    Student s;
    Teacher t;

    cout << "--- Student Details ---" << endl;
    s.getDetails();
    s.showDetails();
    s.study();

    cout << "\n--- Teacher Details ---" << endl;
    t.getDetails();
    t.showDetails();
    t.teach();

    return 0;
}
```

Output:-

```
--- Student Details ---
Enter name: Rahul
Enter age: 20
Name: Rahul, Age: 20
Rahul is studying.

--- Teacher Details ---
Enter name: Neha
Enter age: 35
Name: Neha, Age: 35
Neha is teaching.
```