# Q1

//(Operators)

//Write a program to store your name and salary in an Employee object.

//(Don�t prompt for input; just initialize the object in the program).

//Output your name and current salary. Then give yourself a 2,500 pay rise and output your new salary.

#include <iostream>

#include <string>

class Employee {

public:

    std::string name;

    double salary;

    Employee(std::string empName, double empSalary) {

        name = empName;

        salary = empSalary;

    }

    void givePayRise(double amount) {

        salary += amount;

    }

    void display() {

        std::cout << "Name: " << name << std::endl;

        std::cout << "Salary: RS: " << salary << std::endl;

    }

};

int main() {

    // Initialize Employee object

    Employee emp("Phura sona", 50000);

    // Output name and current salary

    std::cout << "Current details:" << std::endl;

    emp.display();

    // Give a pay rise of 2,500

    emp.givePayRise(2500);

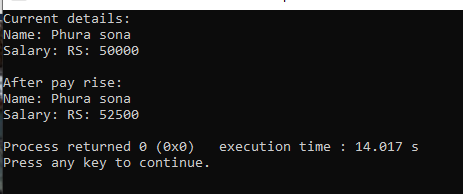
    // Output new salary

    std::cout << "\nAfter pay rise:" << std::endl;

    emp.display();

    return 0;

}



# Q2

// Write a program that creates two simple circle objects. Use default constructor on one and initiate

// the other with the value 7.

// Then print the area of each circle.

#include <iostream>

#include <cmath>

class Circle {

private:

    double radius;

public:

    // Default constructor

    Circle() : radius(1.0) {}

    // Parameterized constructor

    Circle(double r) : radius(r) {}

    // Function to calculate the area of the circle

    double area() const {

        return M\_PI \* radius \* radius;

    }

};

int main() {

    // Create circle objects

    Circle circle1;           // Default constructor

    Circle circle2(7.0);      // Parameterized constructor with radius 7

    // Print the area of each circle

    std::cout << "Area of circle1: " << circle1.area() << std::endl;

    std::cout << "Area of circle2: " << circle2.area() << std::endl;

    return 0;

}

output

Area of circle1: 3.14159

Area of circle2: 153.938

# Q3

Write a program that copies data member of one object to another with the help of a copy constructor.

// Write a program that copies data member of one object to another with the help of a copy constructor.

#include <iostream>

class MyClass {

private:

    int data;

public:

    // Parameterized constructor

    MyClass(int value) : data(value) {}

    // Copy constructor

    MyClass(const MyClass &obj) {

        data = obj.data;

        std::cout << "Copy constructor called!" << std::endl;

    }

    // Function to display the value of data

    void display() const {

        std::cout << "Data: " << data << std::endl;

    }

};

int main() {

    MyClass obj1(42);  // Create an object with data = 42

    MyClass obj2 = obj1;  // Use copy constructor to create obj2 as a copy of obj1

    // Display data of both objects

    std::cout << "Object 1: ";

    obj1.display();

    std::cout << "Object 2: ";

    obj2.display();

    return 0;

}

output

Copy constructor called!

Object 1: Data: 42

Object 2: Data: 42

# Q4

// Create a class Int (note different spelling). The only data in this class is an int variable.

// Include member functions to initialize an Int to 0, to initialize it to an int value, to display it,

// and to add two Int values.

#include <iostream>

class Int {

private:

    int value;

public:

    // Constructor to initialize Int to 0

    Int() : value(0) {}

    // Constructor to initialize Int to a given int value

    Int(int v) : value(v) {}

    // Member function to display the value

    void display() const {

        std::cout << value << std::endl;

    }

    // Member function to add two Int values

    Int add(const Int& other) const {

        return Int(value + other.value);

    }

};

int main() {

    // Create Int objects

    Int num1;          // Initialized to 0

    Int num2(10);      // Initialized to 10

    // Display values

    std::cout << "num1: ";

    num1.display();    // Output: 0

    std::cout << "num2: ";

    num2.display();    // Output: 10

    // Add num1 and num2

    Int num3 = num1.add(num2);

    // Display the result

    std::cout << "num3 (num1 + num2): ";

    num3.display();    // Output: 10

    return 0;

}

output

num1: 0

num2: 10

num3 (num1 + num2): 10

# Q5

// Create a class called time that has separate int member data for hours, minutes, and seconds.

// One constructor should initialize this data to 0 and another should initialize it to fixed values.

// A member function should display it in hh:mm:ss format.

// The final member function should add two objects of time passed as arguments.

// Create two, initialized time objects in the main program and one that isn’t initialized.

// Then it should add the two initialized values together leaving the result in the third time variable.

#include <iostream>

#include <iomanip>

class Time {

private:

    int hours;

    int minutes;

    int seconds;

public:

    // Constructor to initialize to 0

    Time() : hours(0), minutes(0), seconds(0) {}

    // Constructor to initialize to fixed values

    Time(int h, int m, int s) : hours(h), minutes(m), seconds(s) {}

    // Function to display time in hh:mm:ss format

    void display() const {

        std::cout << std::setw(2) << std::setfill('0') << hours << ":"

                  << std::setw(2) << std::setfill('0') << minutes << ":"

                  << std::setw(2) << std::setfill('0') << seconds << std::endl;

    }

    // Function to add two time objects

    void add(const Time &t1, const Time &t2) {

        seconds = t1.seconds + t2.seconds;

        minutes = t1.minutes + t2.minutes + seconds / 60;

        hours = t1.hours + t2.hours + minutes / 60;

        seconds %= 60;

        minutes %= 60;

        hours %= 24;  // Optional: to keep the hours within 0-23 range

    }

};

int main() {

    // Creating initialized time objects

    Time t1(2, 45, 30);

    Time t2(1, 20, 40);

    // Creating an uninitialized time object

    Time t3;

    // Adding the two initialized time objects and storing the result in t3

    t3.add(t1, t2);

    // Displaying the results

    std::cout << "Time 1: ";

    t1.display();

    std::cout << "Time 2: ";

    t2.display();

    std::cout << "Time 3 (Sum): ";

    t3.display();

    return 0;

}

output

Time 1: 02:45:30

Time 2: 01:20:40

Time 3 (Sum): 04:06:10

# Q6

// Write a constructor CHECK that initializes an integer variables x,y and prints “This is COSTRUCTOR example”.

// Print the value of x and y using display function.

#include <iostream>

class CHECK {

private:

    int x, y;

public:

    // Constructor

    CHECK(int a, int b) : x(a), y(b) {

        std::cout << "This is CONSTRUCTOR example" << std::endl;

    }

    // Display function to print x and y

    void display() const {

        std::cout << "x = " << x << ", y = " << y << std::endl;

    }

};

int main() {

    // Create an object of CHECK with x = 10 and y = 20

    CHECK obj(10, 20);

    // Display the values of x and y

    obj.display();

    return 0;

}

output

This is CONSTRUCTOR example

x = 10, y = 20

# Q7

// Write a constructor INI that initializes a variable to 15 and destructor that destroy it.

#include <iostream>

class MyClass {

private:

    int myVariable;

public:

    // Constructor

    MyClass() : myVariable(15) {

        std::cout << "Constructor called. myVariable initialized to " << myVariable << std::endl;

    }

    // Destructor

    ~MyClass() {

        std::cout << "Destructor called. myVariable destroyed." << std::endl;

    }

    // Method to display the variable

    void display() const {

        std::cout << "The value of myVariable is " << myVariable << std::endl;

    }

};

int main() {

    {

        MyClass obj;

        obj.display();

    } // The destructor will be called automatically here when obj goes out of scope

    return 0;

}

output

Constructor called. myVariable initialized to 15

The value of myVariable is 15

Destructor called. myVariable destroyed.

# Q8

// Write a constructor function test that initializes integer variable a to 10 and b to 20 and print “This is constructor example” and member functions,

// Inc that increment the variable a and b by 5.

// Dec that decrements the variable a and b by 3.

// Ret that returns the sum of a and b.

#include <iostream>

using namespace std;

class OperatorsExample {

private:

    int a, b;

public:

    // Constructor

    OperatorsExample() {

        a = 10;

        b = 20;

        cout << "This is constructor example" << endl;

    }

    // Member function to increment a and b by 5

    void Inc() {

        a += 5;

        b += 5;

    }

    // Member function to decrement a and b by 3

    void Dec() {

        a -= 3;

        b -= 3;

    }

    // Member function to return sum of a and b

    int Ret() {

        return a + b;

    }

};

int main() {

    // Create an object of OperatorsExample class

    OperatorsExample obj;

    // Perform increment and decrement operations

    obj.Inc();

    obj.Dec();

    // Output the sum of a and b

    cout << "Sum of a and b: " << obj.Ret() << endl;

    return 0;

}

output

This is constructor example

Sum of a and b: 32