C++ Programming [Day 4]

UID: 24MCI10204

Name: Rahul Saxena

Branch: 24MCA – AI & ML

Question 1: Design a class hierarchy for a university system:

- Base class:
  - Person Members: name, age
  - Virtual function: display()
- Derived Classes:
  - Student:
    - Additional Members: rollNo, course
    - Override display() to show student info
  - Faculty:
    - Additional Members: employeeId, subject
    - Override display() to show faculty info

In main(), use base class pointers to refer to derived class objects and call the display() function.

Expected Concepts

- Single Inheritance
- Virtual Functions & run time polymorphism
- Pointers to base class

Code:

```
#include <iostream>
using namespace std;
class Person {
protected:
   string name;
   int age;
public:
   void setDetails(string n, int a) {
      name = n;
      age = a;
   }
   virtual void display() {
      cout << "Name: " << name << endl;
      cout << "Age: " << age << endl;
   }
};
class Student : public Person {
private:
   int rollNo;
```

```
    string course;
public:
    void setStudentDetails(string n, int a, int r, string c) {
        setDetails(n, a);
        rollNo = r;
        course = c;
    }
    void display() override {
        cout << "Student Details:" << endl;
        cout << "Name: " << name << endl;
        cout << "Age: " << age << endl;
        cout << "Roll No: " << rollNo << endl;
        cout << "Course: " << course << endl;
    }
};
class Faculty : public Person {
private:
    int empId;
    string subject;
public:
    void setFacultyDetails(string n, int a, int e, string s) {
        setDetails(n, a);
        empId = e;
        subject = s;
    }
    void display() override {
        cout << "Faculty Details:" << endl;
        cout << "Name: " << name << endl;
        cout << "Age: " << age << endl;
        cout << "Employee ID: " << empId << endl;
        cout << "Subject: " << subject << endl;
    }
};
int main() {
    Person* ptr;
    Student s1;
    s1.setStudentDetails("Rahul", 20, 101, "Computer Science");
    Faculty f1;
    f1.setFacultyDetails("Dr. Mehta", 45, 501, "Mathematics");
    ptr = &s1;
    ptr->display();
    cout << endl;
    ptr = &f1;
    ptr->display();
    return 0;
}
```

Question 2: Create a class Complex that represents a complex number. Overload the +, -, and * operators to perform operations between two complex numbers. Each object should store:

- Real Part
- Imaginary part

Add a member function display() to show the result.

Expected Concepts:

- Operator overloading
- Member Functions
- Object passing

Code:

```
#include <iostream>
using namespace std;
class Complex {
private:
   float real;
   float imag;
public:
   Complex() {
      real = 0;
      imag = 0;
   }
   Complex(float r, float i) {
      real = r;
      imag = i;
   }
   Complex operator+(Complex c) {
      Complex temp;
      temp.real = real + c.real;
      temp.imag = imag + c.imag;
      return temp;
   }
   Complex operator-(Complex c) {
      Complex temp;
      temp.real = real - c.real;
      temp.imag = imag - c.imag;
      return temp;
   }
   Complex operator*(Complex c) {
      Complex temp;
      temp.real = (real * c.real) - (imag * c.imag);
      temp.imag = (real * c.imag) + (imag * c.real);
      return temp;
   }
   void display() {
      if (imag >= 0)
         cout << real << " + " << imag << "i" << endl;
      else
         cout << real << " - " << -imag << "i" << endl;
   }
};
int main() {
```

```
    Complex num1(4, 3);
    Complex num2(2, -1);
    Complex sum, diff, prod;
    sum = num1 + num2;
    diff = num1 - num2;
    prod = num1 * num2;
    cout << "Sum: ";
    sum.display();
    cout << "Difference: ";
    diff.display();
    cout << "Product: ";
    prod.display();
    return 0;
}
```