Q1.   Define two classes namely student class having data members rollno, name, age and result class having data members mark1, mark2, mark3 derived from student implement these classes through main () showing the mechanism of single inheritance.

```cpp
#include <iostream>

using namespace std;

// Base class

class Student

{

protected:

  int rollno;

  string name;

  int age;

public:

  void getStudentData()

  {

    cout << "Enter Roll No: ";

    cin >> rollno;

    cin.ignore();  // to clear input buffer

    cout << "Enter Name: ";

    getline(cin, name);

    cout << "Enter Age: ";

    cin >> age;

  }

  void displayStudentData()

  {

    cout << "\nRoll No: " << rollno << endl;

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

    cout << "Age: " << age << endl;

  }

};

// Derived classclass Result : public Student

{

private:
```

```
    int mark1, mark2, mark3;
public:
    void getMarks()
    {
        cout << "Enter Mark 1: ";
        cin >> mark1;
        cout << "Enter Mark 2: ";
        cin >> mark2;
        cout << "Enter Mark 3: ";
        cin >> mark3;
    }
    void displayResult()
    {
        int total = mark1 + mark2 + mark3;
        float average = total / 3.0;
        displayStudentData();
        cout << "Marks: " << mark1 << ", " << mark2 << ", " << mark3 << endl;
        cout << "Total: " << total << endl;
        cout << "Average: " << average << endl;
    }
};
// Main function
int main()
{
    Result r;
    r.getStudentData();
    r.getMarks();
    r.displayResult();
    return 0;
}
```

```
∨ TERMINAL                                                                      ⊵ Code
PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q1.cpp -o Q1 } ; if ($?) { .\Q1 }
Enter Roll No: 01
Enter Name: ABC
Enter Age: 20
Enter Mark 1: 50
Enter Mark 2: 70
Enter Mark 3: 80

Roll No: 1
Name: ABC
Age: 20
Marks: 50, 70, 80
Total: 200
Average: 66.6667
PS C:\Users\LENOVO\Desktop\OOPS> ▌
```

Q2. Write a program for multiple inheritances. Define a class publisher that stores the name of the title and another class for sales detail, which stores the number of sales. Derive class book, which inherit both publisher and sales. Define function in the appropriate classes to get and print the details.

```cpp
#include <iostream>

using namespace std;

// Publisher class

class Publisher

{

protected:

    string title;

public:

    void getPublisherData()

    {

      // cout << "Enter Title Name: ";

        getline(cin, title);  // Properly takes full line input

    }

    void displayPublisherData()

    {

        cout << "Title: " << title << endl;

    }

};

// Sales class

class Sales

{

protected:

    int numberOfSales;

public:

    void getSalesData()

    {

        cout << "Enter Number of Sales: ";

        cin >> numberOfSales;

    }
```

```cpp
    void displaySalesData()
    {
        cout << "Number of Sales: " << numberOfSales << endl;
    }
};
// Book class using multiple inheritance
class Book : public Publisher, public Sales
{
public:
    void getBookData()
    {
        getPublisherData();   // Input title first
        getSalesData();       // Then input sales
    }
    void displayBookData()
    {
        cout << "\n--- Book Details ---" << endl;
        displayPublisherData();
        displaySalesData();
    }
};
// Main function
int main()
{
    Book b;
    // Fix: Clear input buffer before getline
    cout << "Enter title name:\n";
    b.getBookData();
    b.displayBookData();
    return 0;
}
```

```
∨ TERMINAL                                                                    ⟩_ Code

PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q2.cpp -o Q2 } ; if ($?) { .\Q2 }
Enter title name:
BHAGAVAD GITA
Enter Number of Sales: 10000

--- Book Details ---
Title: BHAGAVAD GITA
Number of Sales: 10000
PS C:\Users\LENOVO\Desktop\OOPS>
```

Q3. Write a program that contains student as a base class, from which the three classes are arts, science and commerce have been derived illustrates the hierarchical inheritance with constructor. Display the student details along with their subject names.

```cpp
#include <iostream>

using namespace std;
// Base class
class Student
{
protected:
   string name;
   int rollNo;
public:
   // Constructor
   Student(string n, int r)
   {
      name = n;
      rollNo = r;
   }
   // Display common student details
   void displayStudent()
   {
      cout << "Name: " << name << endl;
      cout << "Roll No: " << rollNo << endl;
   }
};
// Derived class - Arts
class Arts : public Student
{
public:
   // Constructor
   Arts(string n, int r) : Student(n, r) {}
   // Display subject
   void displaySubjects()
```

```cpp
    {
        displayStudent();
        cout << "Subjects: History, Political Science, Sociology" << endl;
    }
};
// Derived class - Science
class Science : public Student
{
public:
    // Constructor
    Science(string n, int r) : Student(n, r) {}
    void displaySubjects()
    {
        displayStudent();
        cout << "Subjects: Physics, Chemistry, Mathematics" << endl;
    }
};
// Derived class - Commerce
class Commerce : public Student
{
public:
    // Constructor
    Commerce(string n, int r) : Student(n, r) {}
    void displaySubjects()
    {
        displayStudent();
        cout << "Subjects: Accounting, Economics, Business Studies" << endl;
    }
};
// Main function
int main()
{
    // Creating objects of each derived class
```

```cpp
    Arts a("Raj", 101);

    Science s("Priya", 102);

    Commerce c("Amit", 103);


    cout << "\n--- Arts Student ---\n";

    a.displaySubjects();

    cout << "\n--- Science Student ---\n";

    s.displaySubjects();

    cout << "\n--- Commerce Student ---\n";

    c.displaySubjects()

    return 0;
```

```
TERMINAL                                                                    >_ Code

PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q3.cpp -o Q3 } ; if ($?) { .\Q3 }

--- Arts Student ---
Name: Raj
Roll No: 101
Subjects: History, Political Science, Sociology

--- Science Student ---
Name: Priya
Roll No: 102
Subjects: Physics, Chemistry, Mathematics

--- Commerce Student ---
Name: Amit
Roll No: 103
Subjects: Accounting, Economics, Business Studies
PS C:\Users\LENOVO\Desktop\OOPS>
```

```cpp
}
```

Q4. Write a C++ program to design a base class Person (name, address, phone_no). Derive a class Employee (eno, ename) from Person. Derive a class Manager (Designation, Department name, basic salary) from Employee. Write a Menu Driven Program to:

a) Accept all details of 'n' Managers.

b) display manager having highest salary.

```cpp
#include <iostream>
#include <string>
using namespace std;
class Person
{
protected:
  string name;
  string address;
  string phone_no;
public:
  void getPersonDetails()
  {
    cout << "Enter Name: ";
    getline(cin, name);
    cout << "Enter Address: ";
    getline(cin, address);
    cout << "Enter Phone Number: ";
    getline(cin, phone_no);
  }
  void displayPersonDetails()
  {
    cout << "Name: " << name << endl;
    cout << "Address: " << address << endl;
    cout << "Phone Number: " << phone_no << endl;
  }
};
```

```cpp
class Employee : public Person
{
protected:
   int eno;
   string ename;
public:
   void getEmployeeDetails()
   {
      cout << "Enter Employee Number: ";
      cin >> eno;
      cin.ignore(); // To consume newline
      cout << "Enter Employee Name: ";
      getline(cin, ename);
      getPersonDetails();
   }
   void displayEmployeeDetails()
   {
      cout << "Employee No: " << eno << endl;
      cout << "Employee Name: " << ename << endl;
      displayPersonDetails();
   }
};
class Manager : public Employee
{
private:
   string designation;
   string department;
   float basic_salary;
public:
   void getManagerDetails()
   {
      getEmployeeDetails();
      cout << "Enter Designation: ";
```

```cpp
        getline(cin, designation);

        cout << "Enter Department Name: ";

        getline(cin, department);

        cout << "Enter Basic Salary: ";

        cin >> basic_salary;

        cin.ignore();

    }

    void displayManagerDetails()

    {

        displayEmployeeDetails();

        cout << "Designation: " << designation << endl;

        cout << "Department: " << department << endl;

        cout << "Basic Salary: " << basic_salary << endl;

        cout << "-------------------------\n";

    }

    float getSalary()

    {

        return basic_salary;

    }

};

int main()

{

    int n, choice;

    cout << "Enter number of managers: ";

    cin >> n;

    cin.ignore();

    Manager managers[100]; // Assuming max 100 managers

    bool running = true;

    while (running)

    {

        cout << "\n--- MENU ---\n";

        cout << "1. Accept Details of Managers\n";

        cout << "2. Display Manager with Highest Salary\n";
```

```cpp
        cout << "3. Exit\n";

        cout << "Enter your choice: ";

        cin >> choice;

        cin.ignore();

        switch (choice)

        {

        case 1:

            for (int i = 0; i < n; i++)

            {

                cout << "\nEnter details for Manager " << i + 1 << ":\n";

                managers[i].getManagerDetails();

            }

            break;

        case 2:

        {

            if (n == 0)

            {

                cout << "No manager data entered.\n";

                break;

            }

            float maxSalary = managers[0].getSalary();

            int index = 0;

            for (int i = 1; i < n; i++)

            {

                if (managers[i].getSalary() > maxSalary)

                {

                    maxSalary = managers[i].getSalary();

                    index = i;

                }

            }

            cout << "\nManager with Highest Salary:\n";

            managers[index].displayManagerDetails();

            break;
```

```
        }
    case 3:
        running = false;
        break;
    default:
        cout << "Invalid choice. Try again.\n";
    }
}
return 0;
}
```

```
PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q4.cpp -o Q4 } ; if ($?) { .\Q4 }
Enter number of managers: 1

--- MENU ---
1. Accept Details of Managers
2. Display Manager with Highest Salary
3. Exit
Enter your choice: 1

Enter details for Manager 1:
Enter Employee Number: 01
Enter Employee Name: raj
Enter Name: raj
Enter Address: avenu34
Enter Phone Number: 9975678862
Enter Designation: employee
Enter Department Name: AutoMobile
Enter Basic Salary: 60000
```

```
--- MENU ---
1. Accept Details of Managers
2. Display Manager with Highest Salary
3. Exit
Enter your choice: 2

Manager with Highest Salary:
Employee No: 1
Employee Name: raj
Name: raj
Address: avenu34
Phone Number: 9975678862
Designation: employee
Department: AutoMobile
Basic Salary: 60000
------------------------

--- MENU ---
1. Accept Details of Managers
2. Display Manager with Highest Salary
3. Exit
Enter your choice: 3
PS C:\Users\LENOVO\Desktop\OOPS>
```

Q5. Write a C++ program to use pointer for both base and derived classes and call the member function. Use Virtual keyword

```cpp
#include <iostream>
using namespace std;
// Base class
class Base
{
public:
    virtual void show()
    {
        cout << "Base class show() function called." << endl;
    }
};
// Derived class
class Derived : public Base
{
public:
    void show() override
    {
        cout << "Derived class show() function called." << endl;
    }
};
int main()
{
    Base* basePtr;       // Pointer of base class
    Base baseObj;        // Object of base class
    Derived derivedObj;   // Object of derived class
    // Pointing to base class object
    basePtr = &baseObj;
    basePtr->show();     // Calls Base class version
    // Pointing to derived class object
    basePtr = &derivedObj;
```
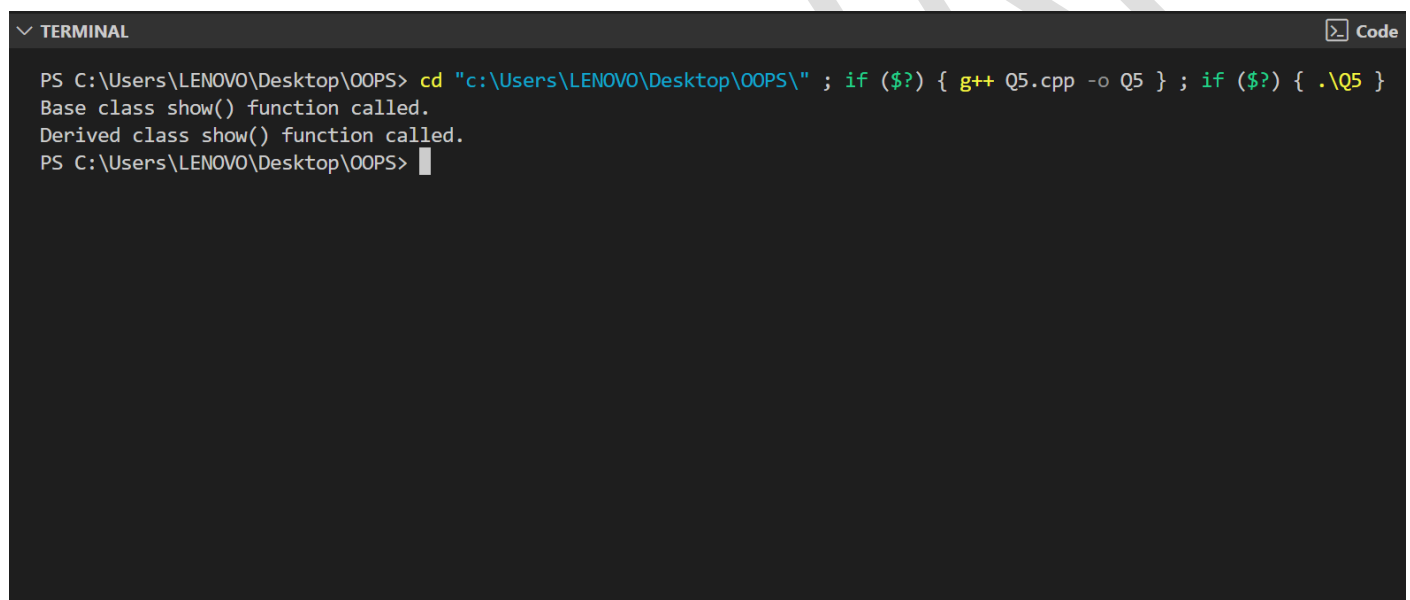
```
    basePtr->show();      // Calls Derived class version due to virtual function

    return 0;

}
```

```
∨ TERMINAL                                                                              ≥_ Code

 PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q5.cpp -o Q5 } ; if ($?) { .\Q5 }
 Base class show() function called.
 Derived class show() function called.
 PS C:\Users\LENOVO\Desktop\OOPS> █
```

Q6. Define a class Weight having data members Kg and Gram and constructor for accepting weights and a member function to display the weights. Your program must also have a capability so that if you enter a weight in from of total grams then it converts these grams to equivalent kg and remaining gram and stores it in respective data members.

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

class Weight {
private:
    int kg;
    int gram;

public:
    // Constructor that accepts kg and gram
    Weight(int k, int g) {
        int totalGrams = k * 1000 + g;
        kg = totalGrams / 1000;
        gram = totalGrams % 1000;
    }

    // Static method to create object from total grams
    static Weight fromGrams(int totalGrams) {
        int k = totalGrams / 1000;
        int g = totalGrams % 1000;
        return Weight(k, g);
    }

    void display() {
        cout << "Weight: " << kg << " kg and " << gram << " gram" << endl;
    }
};

int main() {
```

```cpp
    int choice;
    cout << "1. Enter weight as Kg and Gram\n";
    cout << "2. Enter weight as total Grams\n";
    cout << "Enter your choice: ";
    cin >> choice;

    if (choice == 1) {
        int kg, g;
        cout << "Enter Kilograms: ";
        cin >> kg;
        cout << "Enter Grams: ";
        cin >> g;

        Weight w(kg, g);
        w.display();
    } else if (choice == 2) {
        int totalGrams;
        cout << "Enter total grams: ";
        cin >> totalGrams;

        Weight w = Weight::fromGrams(totalGrams);
        w.display();
    } else {
        cout << "Invalid choice." << endl;
    }
    return 0;
}
```

```
∨ TERMINAL                                                                            >_ Code
PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q6.cpp -o Q6 } ; if ($?) { .\Q6 }
1. Enter weight as Kg and Gram
2. Enter weight as total Grams
Enter your choice: 1
Enter Kilograms: 100
Enter Grams: 200
Weight: 100 kg and 200 gram
PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q6.cpp -o Q6 } ; if ($?) { .\Q6 }
1. Enter weight as Kg and Gram
2. Enter weight as total Grams
Enter your choice: 2
Enter total grams: 3450
Weight: 3 kg and 450 gram
PS C:\Users\LENOVO\Desktop\OOPS> █
```

Q7. Write a C++ program to illustrate the concept of Pointer to Objects.

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


class Student
{
private:
    int rollNo;
    string name;
public:
    void getData()
    {
        cout << "Enter Roll Number: ";
        cin >> rollNo;
        cin.ignore(); // Clear newline from buffer
        cout << "Enter Name: ";
        getline(cin, name);
    }
    void displayData()
    {
        cout << "Roll Number: " << rollNo << endl;
        cout << "Name: " << name << endl;
    }
};
int main()
{
    Student s1;        // Create object
    Student* ptr;      // Declare pointer to object
    ptr = &s1;         // Assign address of object to pointer
    cout << "Enter student details\n";
    ptr->getData();     // Access using pointer
```
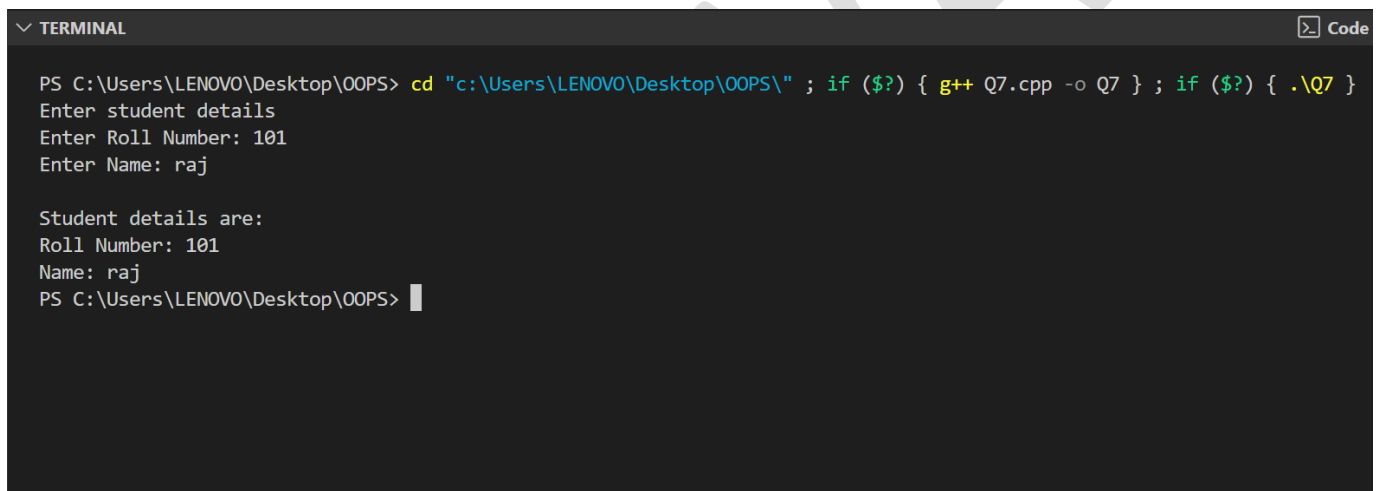
```cpp
    cout << "\nStudent details are:\n";

    ptr->displayData();  // Access using pointer

    return 0;

}
```

```
TERMINAL                                                                                        >_ Code

PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q7.cpp -o Q7 } ; if ($?) { .\Q7 }
Enter student details
Enter Roll Number: 101
Enter Name: raj

Student details are:
Roll Number: 101
Name: raj
PS C:\Users\LENOVO\Desktop\OOPS>
```

Q8. Write a program to overload the operator '*' .
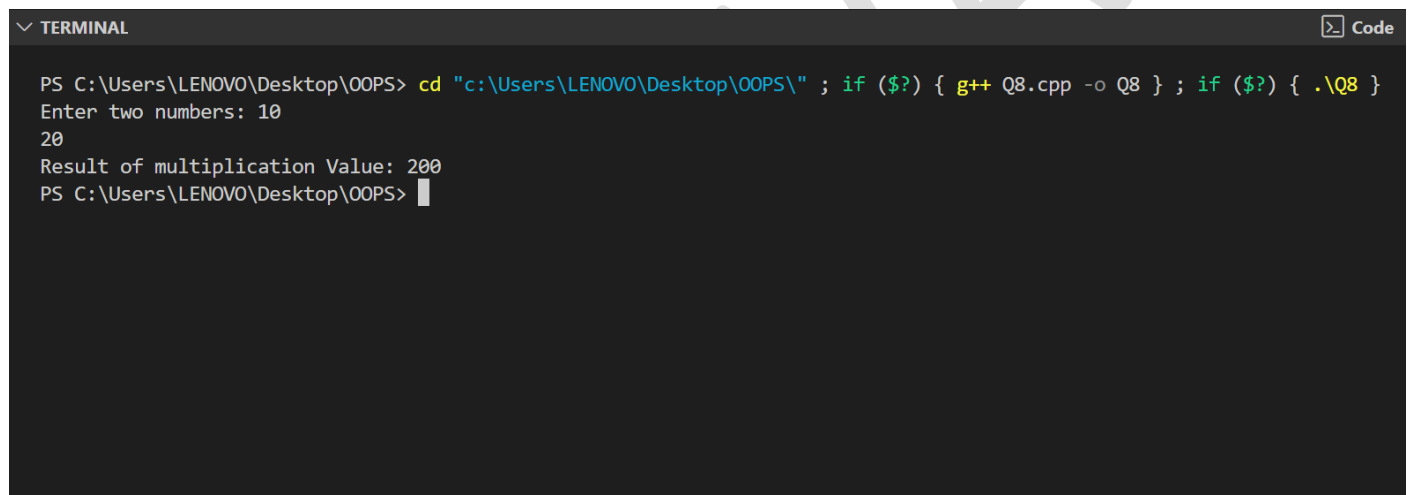
```cpp
#include <iostream>
using namespace std;

class Number
{
private:
    int value;
public:
    // Constructor
    Number(int v = 0)
    {
        value = v;
    }
    // Overloading '*' operator
    Number operator*(const Number& obj)
    {
        Number result;
        result.value = this->value * obj.value;
        return result;
    }
    void display()
    {
        cout << "Value: " << value << endl;
    }
};
int main()
{
    int a, b;
    cout << "Enter two numbers: ";
    cin >> a >> b;
```

```cpp
    Number num1(a), num2(b);

    Number result;

    result = num1 * num2;  // Using overloaded '*' operator

    cout << "Result of multiplication ";

    result.display();

    return 0;

}
```

```
∨ TERMINAL                                                                                    ⟩_ Code

PS C:\Users\LENOVO\Desktop\OOPS> cd "c:\Users\LENOVO\Desktop\OOPS\" ; if ($?) { g++ Q8.cpp -o Q8 } ; if ($?) { .\Q8 }
Enter two numbers: 10
20
Result of multiplication Value: 200
PS C:\Users\LENOVO\Desktop\OOPS> ▮
```