Q-1: Implement a base class "Shape" with two derived classes "Circle" and "Rectangle."

Calculate and display the area of a circle and rectangle using single inheritance.

Sample test case:

|  |
| --- |
| Input: Circle circle(5.0)  Output:Area of Circle: 78.5  Input: Rectangle rectangle(4.0, 6.0)  Output: Area of Rectangle: 24 |

Q-2: Create a base class "Vehicle" with a derived class "Car" and another derived class "SportsCar."Implement the "Vehicle" class to store the vehicle's name and display it.

The "Car" class should store the car's model and display it.The "SportsCar" class should store the top speed of the sports car and display it.

Sample test case:

|  |
| --- |
| Input:  SportsCar myCar("Ferrari", "F430", 320);  myCar.display();  Output:  Vehicle Name: Ferrari  Car Model: F430  Top Speed: 320 km/h |

Q-3: Implement a base class "Person" with two derived classes "Employee" and "Student."

Create another class "Faculty" that is derived from both "Employee" and "Student."

Display the details of a faculty member, including their name, employee ID, and student ID.

Sample test case:

|  |
| --- |
| Input:  Faculty faculty("John Doe", 12345, 98765)  Output:  Name: John Doe  Employee ID: 12345  Name: John Doe  Student ID: 98765 |
|  |

Q-4: Implement a base class "Animal" with two derived classes "Mammal" and "Bird."

Add a function "sound()" to the "Animal" class that displays "Unknown sound." Override the "sound()" function in both "Mammal" and "Bird" to display their respective sounds.

Sample test case:

|  |
| --- |
| Input:  Animal.sound()  Mammal.sound()  Bird.sound()  Output:  Unknown sound  Mammal sound: Roar  Bird sound: Chirp |

Q-5: Create a base class "Vehicle" with two derived classes "Car" and "Bike."

Add a virtual function "start()" in the base class and override it in the derived classes to display their respective starting messages.

Sample test case:

|  |
| --- |
| Input: Car.start()  Output: Car starting...  Input: Bike.start()  Output: Bike starting... |

Q-6: Implement a base class "Shape" with a virtual function "display()" that displays "Shape". Implement two derived classes "Circle" and "Rectangle" that override the "display()" function to display "Circle" and "Rectangle," respectively. Perform upcasting and downcasting with base class pointers.

Sample test case:

|  |
| --- |
| Shape\* shape1 = new Circle;  shape1->display(); // Output: Circle  Shape\* shape2 = new Rectangle;  shape2->display(); // Output: Rectangle |

Q-7: Implement a base class Rectangle with attributes length and width and a method to calculate the area. Create a derived class Box from Rectangle with an additional height attribute and a method to calculate the volume of the box.

Sample test case:

|  |
| --- |
| Input:  Enter the length of the box: 5  Enter the width of the box: 3  Enter the height of the box: 2  Output:  Box's Volume: 30 |

Q-8: Create a base class Shape with a method to calculate the area. Derive three classes Circle, Rectangle, and Triangle from Shape, and implement methods to calculate their respective areas.

Sample test case:

|  |
| --- |
| circle.calculateArea() // Circle's Area: 78.5  rectangle.calculateArea() // Rectangle's Area: 24  triangle.calculateArea() // Triangle's Area: 12 |

Q-9: Create a base class Calculator with methods for addition and subtraction.

Derive a class ScientificCalculator from Calculator and add methods for multiplication and division.

Sample test case:

|  |
| --- |
| Input:  ScientificCalculator sc;  a = 10, b = 3  sc.add(a, b)  sc.subtract(a, b)  sc.multiply(a, b)  sc.divide(a, b)  Output:  Addition: 13  Subtraction: 7  Multiplication: 30  Division: 3 |

Q-10: Create a base class NumberSeries with a method to generate the next term in the series. Derive two classes FibonacciSeries and PrimeSeries from NumberSeries to generate the next Fibonacci number and the next prime number, respectively.Sample test case:

|  |
| --- |
| Input: n=10  Output: Fibonacci Series: 1 2 3 5 8 13 21 34 55 89  Prime Series: 3 4 5 7 11 13 17 19 23 29 |

Q-11: In a virtual reality game, the players are exploring a fantasy world filled with magical objects. As part of their adventure, they come across enchanted circles and cylinders with unique properties. To help the players understand these objects, you need to create a program that calculate area of circle and volume of the cylinder. Cylinder should inherit properties of Circle.

Sample test case:

|  |
| --- |
| Input: radius: 5, height: 7  Output: Cylinder's Volume: 549.5 |

Q-12: Write a C++ program by creating three classes: A, B, and C.

Class A is the base class and contains two data members x and y, and a member function getdata() to input values for x and y.

Class B is derived from class A and contains a member function product() that calculates and displays the product of x and y.

Class C is also derived from class A and contains a member function sum() that calculates and displays the sum of x and y.

In the main() function, create objects of classes B and C, input values for x and y for each object using the getdata() function,

and then calculate and display the product for object B and the sum for object C.

Sample test case:

|  |
| --- |
| Input: Enter value of x and y: 4 5  Output: Product= 20  Input: Enter value of x and y: 2 4  Output: Sum= 6 |

Q-13: Create a base class "Person" with two derived classes "Student" and "Employee."

Implement functions to display details of students and employees by use of multiple inheritance.

Sample test case:

|  |
| --- |
| Student student("John Doe", 12345)  student.displayDetails() // Name: John Doe, Roll Number: 12345  Employee employee("Jane Smith", "E789")  employee.displayDetails() // Name: Jane Smith, Employee ID: E789 |

Q-14: Create a base class "Animal" and two derived classes "FlyingAnimal" and "SwimmingAnimal." Implement a class "Bird" that inherits from both "FlyingAnimal" and "Animal" by using virtual base classes.

Sample test case:

|  |
| --- |
| Input: Bird parrot("Duck")  Output: Duck can fly. Duck can swim. |

Q-15: Implement a base class "Base" and two derived classes "A" and "B." Add a static member variable to "Base" and use it in "A" and "B."

Sample test case:

|  |
| --- |
| Base::staticVar // Output:Static Variable Value: 0  A objA1;  A objA2;  B objB1;  B objB2;  Base::staticVar // Output:Static Variable Value: 4 |

Q-16: Create a program that manages animal information, including species and characteristics such as legs for mammals and flying ability for birds. Create a base class Animal Create two derived classed that inherit from Base as class Mammal and class Bird. Print Species and Legs for Mammals and print Species and Can fly for Birds.

Sample test case:

|  |
| --- |
| Input:  Mammal lion("Lion", 4)  Output:  Species: Lion  Legs: 4  Input:  Bird sparrow("Sparrow", true)  Output:  Species: Sparrow  Can Fly: Yes |

Q-17: Create a base class "A" with two derived classes "B" and "C". Implement a class "D" that inherits from both "B" and "C," and call base class constructors during multiple inheritance

Sample test case:

|  |
| --- |
| Input: D obj;  Output:  A's Constructor  B's Constructor  C's Constructor  D's Constructor |

Q-18: Create a base class "Device" and two derived classes "Keyboard" and "Mouse."

Implement a class "ComboDevice" that inherits from both "Keyboard" and "Mouse" and perform function overriding with virtual base classes.

Sample test case:

|  |
| --- |
| ComboDevice combo;  combo.type();// Output: Combo Device |

Q-19: Create a program that classifies animals into carnivores, herbivores, and omnivores using hybrid inheritance.

Sample test case:

|  |
| --- |
| Omnivore omnivore("Bear");  omnivore.display(); // Output: Species: Bear |

Q-20: Create a base class called Vehicle, and derive two classes Car and Bike from it.

Implement a virtual function startEngine() in the Vehicle class. Override the startEngine() function in both Car and Bike classes to start their respective engines.

Sample test case:

|  |
| --- |
| Vehicle\* vehicle1 = new Car();  vehicle1->startEngine(); // Output: Car engine started.  Vehicle\* vehicle2 = new Bike();  vehicle2->startEngine(); // Output: Bike engine started. |

Q-21: You work as a software engineer for a company specializing in CAD (Computer-Aided Design) software. Your team has been assigned a task to create a C++ program that handles 3D shapes. The program should support two types of 3D shapes: spheres and cones. The main goal is to calculate and display the volumes of these shapes using method overriding.

Sample test case:

|  |
| --- |
| Input:  Sphere(radius=5)  Cone(radius=3, height=7)  Output:  Volume of Sphere: 523.333  Volume of Cone: 65.94 |

Q-22: You work as a software engineer for a company specializing in CAD (Computer-Aided Design) software. Your team has been assigned a task to create a C++ program that handles 3D shapes. The program should support two types of 3D shapes: spheres and cones. The main goal is to calculate and display the volumes of these shapes using method overriding.

Sample test case:

|  |
| --- |
| Input:  Sphere(radius=5)  Cone(radius=3, height=7)  Output:  Volume of Sphere: 523.333  Volume of Cone: 65.94 |

Q-23: Create a base class called Person, and derive two classes Student and Teacher from it. Implement a virtual function introduce() in the Person class, which prints "I am a person." Override the introduce() function in both Student and Teacher classes to print their roles and names.

Sample test case:

|  |
| --- |
| Person\* person1 = new Student("Alice");  person1->introduce(); // Output: I am a student named Alice.  Person\* person2 = new Teacher("Mr. Smith");  person2->introduce(); // Output: I am a teacher named Mr. Smith. |

Q-24: Create one Base class and derive Derived class from it. Add default and parameterized constructors in both classes. Pass Arguments to constructor and monitor it.

Sample test case:

|  |
| --- |
| Input: Derived d(25, 15)  Output: Param of Base 25  Param of Derived 15 |

Q-25: Create a program to calculate sine, cosine, and tangent of an angle in radians using hierarchical inheritance.

Sample test case:

|  |
| --- |
| angleInRadians = 3.14 / 4.0  Sine sine(angleInRadians);  Cosine cosine(angleInRadians);  Tangent tangent(angleInRadians);  sine.calculate() //Output: Sine of 0.785 radians: 0.706825  cosine.calculate() // Output: Cosine of 0.785 radians: 0.707388  tangent.calculate() //Output: Tangent of 0.785 radians: 0.999204 |

Q-26: Implement a hierarchical inheritance system for educational institutions such as schools, colleges, and universities. Each educational institution has common attributes like name, location, and establishment year, along with specific attributes like the number of students and courses offered.

Sample test case:

|  |
| --- |
| Input:  EducationalInstitution\* institution1 = new School("ABC School", "City-A", 1990, 500);  institution1->displayInfo();  Output:  School Name: ABC School  Location: City-A  Establishment Year: 1990  Number of Students: 500 |

Q-27: Design a hierarchy of employees, including a base class for Employee and derived classes for Manager, Engineer, and Technician.Each class should have specific attributes like designation and salary.

Sample test case:

|  |
| --- |
| Input:  Employee\* emp1 = new Manager("John Doe", 101, "Senior Manager", 80000);  emp1->displayDetails();  Output:  Name: John Doe  Employee ID: 101  Designation: Senior Manager  Salary: $80000 |

Q-27: Design a hierarchy for electronic devices, including a base class for Device and derived classes for Smartphone, Laptop, and Television. Each device should have specific attributes like screen size and battery life.

Sample test case:

|  |
| --- |
| Input:  Device\* device1 = new Smartphone("Apple", 6.1, 20);  device1->displaySpecs();  Output:  Brand: Apple  Screen Size: 6.1 inches  Battery Life: 20 hours |

Q-28: Design a hierarchy of vehicles, including a base class for Vehicle and derived classes for Car and Bike. Each class should have specific attributes like the number of wheels and fuel type.

Sample test case:

|  |
| --- |
| Input:  Car car(4, "Petrol", 4);  car.displayInfo();  Output:  Car Info:  Number of Wheels: 4  Fuel Type: Petrol  Number of Doors: 4 |

Q-29: Create a hierarchy of game characters with a base class Character and derived classes Warrior, Mage, and Archer. Implement a method specialAbility() in the base class and override it in the derived classes to display the special ability of

each character type.

Sample test case:

|  |
| --- |
| Input:  Character\* warrior = new Warrior("Aragorn");  warrior->specialAbility();  Output:  Aragorn is a Warrior. Special Ability: Devastating Sword Strike!  Input:  Character\* mage = new Mage("Gandalf");  mage->specialAbility();  Output:  Gandalf is a Mage. Special Ability: Fireball Attack!  Input:  Character\* archer = new Archer("Legolas");  archer->specialAbility();  Output:  Legolas is an Archer. Special Ability: Precise Arrow Shot! |

Q-30: Create a media library in C++ with a base class MediaItem and two derived classes Book and Movie. Each media item has a title and a year of release. Implement a virtual function display() in the MediaItem class, and override it in the derived classes to display specific details of a book and a movie.

Sample test case:

|  |
| --- |
| Input:  MediaItem\* item1 = new Book("The Great Gatsby", 1925, "F. Scott Fitzgerald");  item1->display();  Output:  Title: The Great Gatsby  Author: F. Scott Fitzgerald  Year of Release: 1925  Input:  MediaItem\* item2 = new Movie("Inception", 2010, "Christopher Nolan");  item2->display();  Output:  Title: Inception  Director: Christopher Nolan  Year of Release: 2010 |