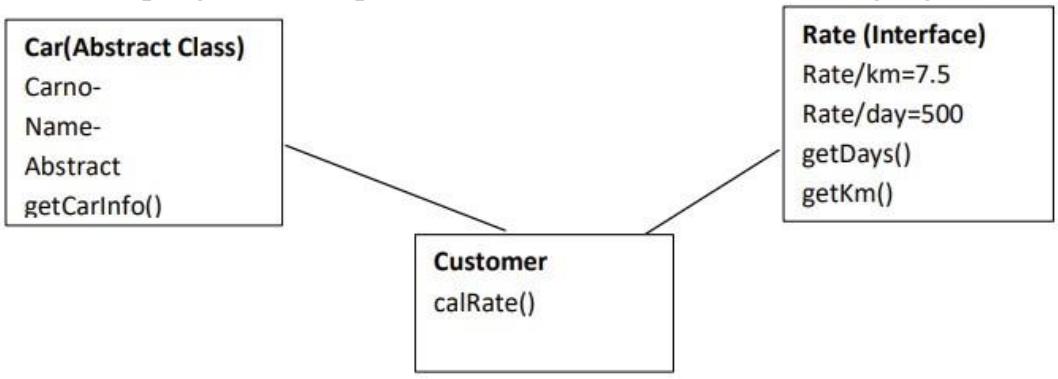
PRACTICAL - 3

Inheritance and Interfaces

Q-1) Write a program to implement inheritance from following figure.



Code:

```
using System;

namespace    Practical3

{

   class Customer{

      protected    string    name;

      protected string address;

      public  Customer(string  name,  string  address){

         this.name = name;

         this.address  =  address;

      }

      public virtual void DisplayInfo(){

         Console.WriteLine("Customer Name: " + name);

         Console.WriteLine("Customer  Address:  "  +  address);

      }

   }

   class Car : Customer{

      protected string carName;
```

PRACTICAL - 3

```
    public Car(string name, string address, string carName) : base(name, address){
        this.carName = carName;
    }
    public override void DisplayInfo()
    {
        base.DisplayInfo();
        Console.WriteLine("Car Name: " + carName);
    }
}
class Km : Car
{
    protected int kilometers;
    public Km(string name, string address, string carName, int kilometers) : base(name, address, carName){
        this.kilometers = kilometers;
    }
    public override void DisplayInfo(){
        base.DisplayInfo();
        Console.WriteLine("Kilometers Driven: " + kilometers + " km");
    }
}
class Days : Car{
    protected int rentalDays;
    public Days(string name, string address, string carName, int rentalDays) : base(name, address, carName)
    {
        this.rentalDays = rentalDays;
```

PRACTICAL - 3

```
    }

    public override void DisplayInfo()

    {

        base.DisplayInfo();

        Console.WriteLine("Rental Days: " + rentalDays + " days");

    }

}

class TotalAmount : Km{

    private readonly int rentalDays;

    private double ratePerKm;

    private double ratePerDay;

    public TotalAmount(string name, string address, string carName, int kilometers, int rentalDays, double ratePerKm, double ratePerDay)

        : base(name, address, carName, kilometers)

    {

        this.rentalDays = rentalDays;

    this.ratePerKm = ratePerKm;

    this.ratePerDay = ratePerDay;

    }

    public override void DisplayInfo()

    {

        base.DisplayInfo();

        double totalAmount = kilometers * ratePerKm + rentalDays * ratePerDay;

        Console.WriteLine("Total Amount: " + totalAmount);

    }

}
```

PRACTICAL - 3

```
class Class1 {

    static void Main(){

        string customerName = "jaymin valaki";

         string customerAddress = "Nadiad";

        string carName = "BMW";

        int  kilometersDriven  =  100;

        int rentalDays = 5;

        double  ratePerKm  =  0.25;

        double ratePerDay = 50;

        TotalAmount totalAmountObj = new TotalAmount(customerName, customerAddress, carName, kilometersDriven, rentalDays, ratePerKm, ratePerDay);

        totalAmountObj.DisplayInfo();

        Console.ReadKey();

    }

  }

}
```

Output:

Customer Name: jaymin valaki

Customer Address: Nadiad

Car Name: BMW

Kilometers Driven: 100 km

Rental Days: 5 days

Total Amount: 225

PRACTICAL - 3

Q-2) Write a program to implement single inheritance from following figure. Accept and display data for one table.

Code:

```
using System;

namespace    practical3

{

   class Furniture{

      protected   string   material;

      protected int price;

      public  Furniture(string  material,  int  price)  {

         this.material = material;

         this.price = price;

      }

      public   void   AcceptData()

      {

         Console.Write("Enter Material: ");

         material = Console.ReadLine();

         Console.Write("Enter price: ");

         price   =   Convert.ToInt32(Console.ReadLine());

      }


      public   void   DisplayData()

      {

         Console.WriteLine("Material:  "  +  material);

         Console.WriteLine("Price: " + price);
```

PRACTICAL - 3

```
    }
}
class Table : Furniture{
private int height;
    private int surface_area;
    public Table(string material, int price, int height, int surface_area) : base(material, price)
    {
        this.height = height;
        this.surface_area = surface_area;
    }
    public new void AcceptData(){
        base.AcceptData();
        Console.Write("Enter height: ");
        height = Convert.ToInt32(Console.ReadLine());

        Console.Write("Enter surface area: ");
        surface_area = Convert.ToInt32(Console.ReadLine());
    }
    public new void DisplayData()
    {
        base.DisplayData();
        Console.WriteLine("Height: " + height);
        Console.WriteLine("Surface Area: " + surface_area);
    }
}
```

PRACTICAL - 3

```
  class Class2{
     static  void  Main()
     {
        Console.WriteLine("Enter  Furniture  Information:");
        Table table = new Table("", 0, 0, 0);
        table.AcceptData();
        Console.WriteLine("\nFurniture    Details:");
        table.DisplayData();
        Console.ReadKey();
     }
  }
}
```

Output:

Enter Furniture Information:

Enter Material: Wood

Enter price: 200

Enter height: 80

Enter surface area: 120

Furniture Details:

Material: Wood

Price: 200

Height: 80

Surface Area: 120

PRACTICAL - 3

Q-3) Define a class "salary" which will contain member variable Basic, TA, DA, HRA. Write a program using Constructor with default values for DA and HRA and calculate the salary of employee.

Code:

```
using System;

namespace    practical3{

   class Salary {

      private   double   Basic;

      private    double    TA;

      private    double    DA;

      private double HRA;

      public Salary(double basic, double ta, double da = 0, double hra = 0) {

         Basic = basic;

         TA = ta;

         DA = da;

      HRA = hra;

      }

      public   double   CalculateSalary()

      {

         double totalSalary = Basic + TA + DA + HRA;

         return totalSalary;

      }

   }
```

PRACTICAL - 3

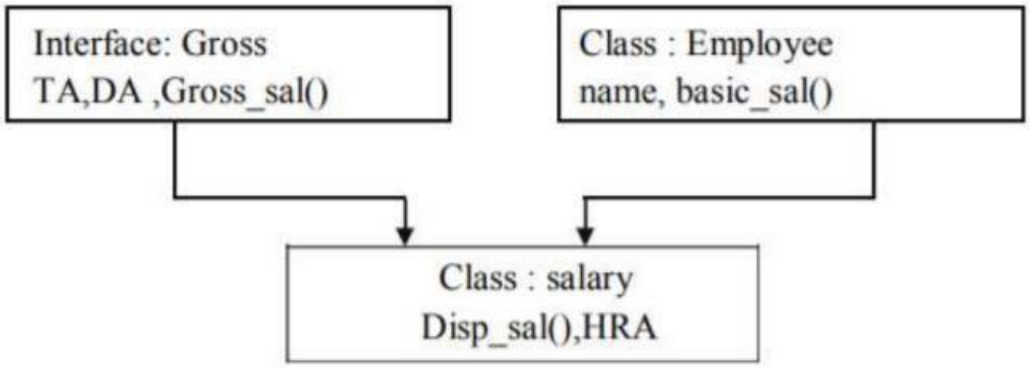
```
class Class3
{
    static  void  Main()
    {
        Salary employee1 = new Salary(45000, 7000);
        double totalSalary1 = employee1.CalculateSalary();
        Console.WriteLine("Employee 1 Total Salary: " + totalSalary1);

        Salary employee2 = new Salary(50000, 1000, 8000, 7000);
        double totalSalary2 = employee2.CalculateSalary();
        Console.WriteLine("Employee 2 Total Salary: " + totalSalary2);
        Console.ReadKey();
    }
}
}
```

Output:

Employee 1 Total Salary: 52000

Employee 2 Total Salary: 66000

Q-4) Program to implement the following multiple inheritance using interface.



PRACTICAL - 3

Code:

```
using      System;

class Program{

   static void Main(string[] args) {

      Salary employee1 = new Salary();

      employee1.Name = "jay";

      employee1.Basic_sal = 5000;

      employee1.HRA = 2000;

      Console.WriteLine("Employee  1  Details:");

      employee1.Disp_sal();

      Console.ReadKey();

   }

}

interface Gross{

   double TA { get; }

   double DA { get; }

   double Gross_sal(); }

class Employee : Gross{

   public string Name { get; set; }

   public double Basic_sal { get; set; }

   public double TA{

      get { return 1000; }

   }

   public double DA{
```

PRACTICAL - 3

```
        get { return 1500; }
    }
    public double Gross_sal(){
        return Basic_sal + TA + DA;
    }
}
class Salary : Employee{
    public double HRA { get; set; }
    public void Disp_sal(){
        Console.WriteLine("Employee Name: " + Name);
        Console.WriteLine("Basic Salary: " + Basic_sal);
        Console.WriteLine("TA: " + TA);
        Console.WriteLine("DA: " + DA);
        Console.WriteLine("HRA: " + HRA);
        Console.WriteLine("Gross Salary: " + Gross_sal());
    }
}
```

Output:

Employee 1 Details:

Employee Name: jay

Basic Salary: 5000

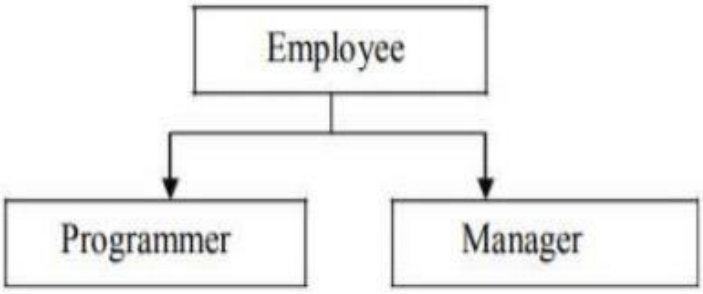
TA: 1000

DA: 1500

HRA: 2000
Gross Salary: 9500

PRACTICAL - 3

Q-5) Write a program for above class hierarchy for the Employee where the base class is Employee and derived class and Programmer and Manager. Here make display function virtual which is common for all and which will display information of Programmer and Manager interactively.



Code:

```
using      System;

class    Employee

{

   protected   string   name;

   protected int empId;

   protected string designation;

   public Employee(string name, int empId, string designation)

   {

      this.name    =    name;

      this.empId = empId;

      this.designation   =   designation;

   }

   public  virtual  void  AcceptData()

   {

      Console.Write("Enter  Employee  Name:  ");

      name = Console.ReadLine();
```

PRACTICAL - 3

```
        Console.Write("Enter Employee ID: ");
        empId = Convert.ToInt32(Console.ReadLine());
        Console.Write("Enter Designation: ");
    designation = Console.ReadLine();
    }
    public virtual void Display(){
        Console.WriteLine("Employee Name: " + name);
        Console.WriteLine("Employee ID: " + empId);
        Console.WriteLine("Designation: " + designation); }
}
class  Programmer  :  Employee
{
    private string programmingLanguage;
  public Programmer(string name, int empId, string designation, string programmingLanguage)
        : base(name, empId, designation)
    {
        this.programmingLanguage  =  programmingLanguage;
    }
    public  override  void  AcceptData(){
    base.AcceptData();
        Console.Write("Enter Programming Language: ");
    programmingLanguage = Console.ReadLine();
    }
    public override void Display(){
```

PRACTICAL - 3

```
        base.Display();
        Console.WriteLine("Programming Language: " + programmingLanguage);
    }
}
class Manager : Employee{
    private string department;
    public Manager(string name, int empId, string designation, string department)
        : base(name, empId, designation)
    {
        this.department = department;
    }
    public override void AcceptData(){
    base.AcceptData();
        Console.Write("Enter Department: ");
    department = Console.ReadLine();
    }
    public override void Display(){
    base.Display();
        Console.WriteLine("Department: " + department);
    }
}
class Program{
    static void Main(){
        Console.WriteLine("Enter Programmer Information:");
        Programmer programmer = new Programmer("", 0, "", "");
```

```
        programmer.AcceptData();
        Console.WriteLine("\nEnter Manager Information:");
        Manager manager = new Manager("", 0, "", "");
        manager.AcceptData();
        Console.WriteLine("\nProgrammer    Details:");
        programmer.Display();
        Console.WriteLine("\nManager    Details:");
        manager.Display();
        Console.ReadKey();
    }
}
```

Output:

Programmer Details:

Employee Name: Jaymin

Employee ID: 67

Designation: Developer

Programming Language: Android

Manager Details:

Employee Name: karan

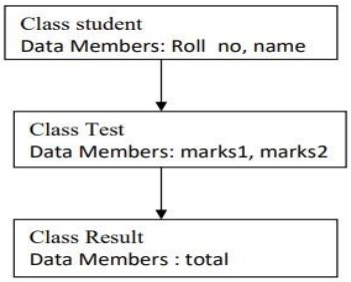
Employee ID: 82

Designation: Manager

Department: Sales

Q-6) Write a program to implement multilevel inheritance from following figure. Accept and display data for one student

PRACTICAL - 3



Code:

```
using    System;

class Student{

   protected int Roll_no;

   protected string Name;

   public void AcceptData(){

      Console.Write("Enter Roll Number: ");

      Roll_no  =  Convert.ToInt32(Console.ReadLine());

      Console.Write("Enter Name: ");

      Name  =  Console.ReadLine();

   }

   public void DisplayData(){

      Console.WriteLine("Roll Number: " + Roll_no);

      Console.WriteLine("Name: " + Name);

   }

}

class Test : Student{

   protected  int  marks1;

   protected int marks2;

   public void AcceptMarks(){
```

PRACTICAL - 3

```
        Console.Write("Enter Marks 1: ");
        marks1  =  Convert.ToInt32(Console.ReadLine());
        Console.Write("Enter Marks 2: ");
        marks2  =  Convert.ToInt32(Console.ReadLine());
    }
    public void DisplayMarks(){
        Console.WriteLine("Marks 1: " + marks1);
        Console.WriteLine("Marks 2: " + marks2); }
}
class  Result  :  Test{
    private int total;
    public  void  CalculateTotal(){
        total = marks1 + marks2; }
    public void DisplayResult(){
        Console.WriteLine("Total  Marks:  "  +  total);
    }
}
class Program{
    static void Main(){
        Result result = new Result();
        Console.WriteLine("Enter  Student  Information:");
        result.AcceptData();
        Console.WriteLine("\nEnter Marks Information:");
```

PRACTICAL - 3

```
        result.AcceptMarks();
        result.CalculateTotal();
        Console.WriteLine("\nStudent    Details:");
        result.DisplayData();
        Console.WriteLine("\nMarks    Details:");
        result.DisplayMarks();
        Console.WriteLine("\nResult:");
        result.DisplayResult();
        Console.ReadKey();
    }
}
```

Output:

Enter Roll Number: 67

Enter Name: JAY

Enter Marks Information:

Enter Marks 1: 85

Enter Marks 2: 78

Student Details:

Roll Number: 67

Name: JAY

Marks Details:

Marks 1: 85

Marks 2: 78

Result: Total Marks: 163