ASSIGNMENT -3

Inheritance and Polymorphism

a) Implement the following class hierarchy by defining all functions of each class. Demonstrate the method overriding and access to each data member using an object of bottom most class. In addition to all operations, show the order of execution for constructors.

Class : Point

Members: xco, yco

Functions: constructors, print()

Class : Circle (inherits Point)

Members : radius

Functions: consturctors, print(), float area()

Class: Cylinder (inherits Circle)

Members : height

Functions: constructors, print, float surarea(), float volume()

CODE :-

```
import java.util.Scanner;
class Point{
   float xco;
   float yco;
   Point(){
      xco = 0;
      yco = 0;
      System.out.println("Point's Default Constructor is called!");
   }
```

```
    Point(float xco , float yco){
        this.yco = yco;
        this.xco = xco;
        System.out.println("Point's Parameterized Constructor is called!");
    }
    void print(){
        System.out.println("xco : " + xco +  " , yco : " + yco);
    }
}

class Circle extends Point{
    float radius;

    Circle(){
        radius = 0;
        System.out.println("Circle's Default Constructor is called!");
    }
    Circle(float radius){
        this.radius = radius;
        System.out.println("Circle's Parameterized Constructor is called!");
    }

    @Override
    void print(){
        System.out.println("Circle's Centre : " + xco + " , " + yco);
        System.out.println("Circle's radius : " + radius);
    }

    float area(){
```

```
        return ((float)((22 / 7) * radius * radius));
    }

}

class Cylinder extends Circle{
    float height;

    Cylinder(){
        height = 0;
        System.out.println("Cylinder's Default Constructor is called!");
    }

    Cylinder(float radius , float height){
        this.radius = radius;
        this.height = height;
        System.out.println("Cylinder's Parameterized Constructor is called!");
    }

    @Override

    void print(){
        System.out.println("Cylinder's Centre : " + xco + " , " + yco);
        System.out.println("Cylinder's radius : " + radius);
        System.out.println("Cylinder's height : " + height);
    }

    float surarea(){
        return ((float)(2 * Math.PI * radius * height) + (float)(2 * 3.14 * radius * radius));
```

```
    }

    float volume(){
        return (float)(Math.PI * radius * radius * height);
    }
}

public class Q1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Cylinder cylinder = null;
        while(true) {
            System.out.println("\nCylinder Operations Menu:");
            System.out.println("1. Create new Cylinder");
            System.out.println("2. Print Cylinder details");
            System.out.println("3. Calculate Surface Area");
            System.out.println("4. Calculate Volume");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();
            switch(choice) {
                case 1:
                    System.out.print("Enter radius: ");
                    float radius = sc.nextFloat();
                    System.out.print("Enter height: ");
                    float height = sc.nextFloat();
                    cylinder = new Cylinder(radius, height);
                    System.out.println("Cylinder created successfully!");
                    break;
```

```
case 2:
   if(cylinder != null) {
      cylinder.print();
   }
   else {
      System.out.println("Please create a cylinder first!");
   }
   break;

case 3:
   if(cylinder != null) {
      System.out.println("Surface Area: " + cylinder.surarea());
   }
   else {
      System.out.println("Please create a cylinder first!");
   }
   break;

case 4:
   if(cylinder != null) {
      System.out.println("Volume: " + cylinder.volume());
   }
   else {
      System.out.println("Please create a cylinder first!");
   }
   break;

case 5:
```

```
                System.out.println("Exiting program...");
                sc.close();
                return;

            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
  }
}
```

---

b) Write a java program to implement the following class hierarchy. Show the use of dynamic method dispatch. In addition to all operations, show the order of execution for constructors and demostrate the use of super keyword. Verify the balance and limit while performing deposit and withdraw method in respective classes.

Class : Account

Members: accno, name, type, balance

Methods: constructors, displayAcDetails()

Class : SavingAccount (inherits Account)

Members : minbal, withdrawlimit, interest

Methods: consturctors, displayAcDetails(), deposit(), withdraw()

Class: CurrentAccount (inherits Account)

Members : withdrawlimit, noofTxperday, interest

Methods: consturctors, displayAcDetails(), deposit(), withdraw()

CODE :-

```
import java.util.Scanner;
class Account {
```

```
    long accno;
    double balance;
    String name;
    String type;
    Account() {
        accno = 00000000000000;
        name = "Default";
        type = "Default";
        balance = 0;
    }
    Account(double balance, long accno, String name, String type) {
        this.accno = accno;
        this.balance = balance;
        this.name = name;
        this.type = type;
    }
    void displayAcDetails() {
        System.out.println("Account Details : ");
        System.out.println("Name : " + name);
        System.out.println("Account No. : " + accno);
        System.out.println("AccountType : " + type);
        System.out.println("Account Balance : " + balance);
    }
}
class SavingAccount extends Account {
    double minbal;
    double withdrawlimit;
    double interest;
    SavingAccount() {
```

```
        type = "Savings";
        minbal = 1000;
        withdrawlimit = 500;
        interest = 7.0;
    }
    SavingAccount(double minbal, double withdrawlimit, double interest, double bal, long
accno, String name, String type) {
        super(bal, accno, name, type);
        this.minbal = minbal;
        this.withdrawlimit = withdrawlimit;
        this.interest = interest;
    }
    @Override
    void displayAcDetails() {
        System.out.println("Account Details : ");
        System.out.println("Name : " + name);
        System.out.println("Account No. : " + accno);
        System.out.println("AccountType : " + type);
        System.out.println("Minimum Balance : " + minbal);
        System.out.println("Withdraw Limit: " + withdrawlimit);
        System.out.println("Interest : " + interest);
        System.out.println("Account Balance : " + balance);
    }
    void deposit(double amount) {
        balance = balance + amount;
        System.out.println("Amount Deposited Successfully!");
    }
    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient Balance");
```

```
        } else if (amount > withdrawlimit) {
            System.out.println("Entered amount is greater than withdraw limit");
        } else {
            balance -= amount;
            System.out.println("Amount Withdrawn Successfully!");
        }
    }
}
class CurrentAccount extends Account {
    double withdrawlimit;
    int noofTxperday;
    double interest;
    int trans = 0;
    CurrentAccount() {
        noofTxperday = 10;
        type = "Current";
        withdrawlimit = 500;
        interest = 7.0;
    }
    CurrentAccount(int noofTxperday, double withdrawlimit, double interest, double bal, long
accno, String name, String type) {
        super(bal, accno, name, type);
        this.noofTxperday = noofTxperday;
        this.withdrawlimit = withdrawlimit;
        this.interest = interest;
    }
    @Override
    void displayAcDetails() {
        System.out.println("Account Details : ");
        System.out.println("Name : " + name);
```

```
        System.out.println("Account No. : " + accno);
        System.out.println("AccountType : " + type);
        System.out.println("Number of Transactions per day  : " + noofTxperday);
        System.out.println("Withdraw Limit: " + withdrawlimit);
        System.out.println("Interest : " + interest);
        System.out.println("Account Balance : " + balance);
    }
    void deposit(double amount) {
        balance = balance + amount;
        System.out.println("Amount Deposited Successfully!");
    }
    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient Balance");
        }
        else if (amount > withdrawlimit) {
            System.out.println("Entered amount is greater than withdraw limit");
        }
        else {
            if (trans > noofTxperday) {
                System.out.println("Transaction limit reached");
                return;
            }
            balance -= amount;
            trans++;
            System.out.println("Amount Withdrawn Successfully!");
        }
    }
}
```

```
public class Q2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        SavingAccount savingAcc = null;
        CurrentAccount currentAcc = null;
        while (true) {
            System.out.println("\nBank Account Operations Menu:");
            System.out.println("1. Create Savings Account");
            System.out.println("2. Create Current Account");
            System.out.println("3. Display Account Details");
            System.out.println("4. Deposit Money");
            System.out.println("5. Withdraw Money");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();
            sc.nextLine();
            switch (choice) {
                case 1:
                    System.out.print("Enter Name: ");
                    String name = sc.nextLine();
                    System.out.print("Enter Account Number: ");
                    long accno = sc.nextLong();
                    System.out.print("Enter Initial Balance: ");
                    double balance = sc.nextDouble();
                    System.out.print("Enter Minimum Balance: ");
                    double minbal = sc.nextDouble();
                    System.out.print("Enter Withdraw Limit: ");
                    double withdrawLimit = sc.nextDouble();
                    System.out.print("Enter Interest Rate: ");
```

```
                double interest = sc.nextDouble();
                savingAcc = new SavingAccount(minbal, withdrawLimit, interest, balance, accno, name, "Savings");
                System.out.println("Savings Account Created Successfully!");
                break;
            case 2:
                System.out.print("Enter Name: ");
                name = sc.nextLine();
                System.out.print("Enter Account Number: ");
                accno = sc.nextLong();
                System.out.print("Enter Initial Balance: ");
                balance = sc.nextDouble();
                System.out.print("Enter Number of Transactions per Day: ");
                int noofTx = sc.nextInt();
                System.out.print("Enter Withdraw Limit: ");
                withdrawLimit = sc.nextDouble();
                System.out.print("Enter Interest Rate: ");
                interest = sc.nextDouble();
                currentAcc = new CurrentAccount(noofTx, withdrawLimit, interest, balance, accno, name, "Current");
                System.out.println("Current Account Created Successfully!");
                break;
            case 3:
                System.out.println("\n1. Savings Account\n2. Current Account");
                System.out.print("Choose account type: ");
                int type = sc.nextInt();
                if (type == 1 && savingAcc != null) {
                    savingAcc.displayAcDetails();
                } else if (type == 2 && currentAcc != null) {
                    currentAcc.displayAcDetails();
```

```
    } else {
        System.out.println("Account not created yet!");
    }
    break;
case 4:
    System.out.println("\n1. Savings Account\n2. Current Account");
    System.out.print("Choose account type: ");
    type = sc.nextInt();
    System.out.print("Enter amount to deposit: ");
    double amount = sc.nextDouble();
    if (type == 1 && savingAcc != null) {
        savingAcc.deposit(amount);
    } else if (type == 2 && currentAcc != null) {
        currentAcc.deposit(amount);
    } else {
        System.out.println("Account not created yet!");
    }
    break;
case 5:
    System.out.println("\n1. Savings Account\n2. Current Account");
    System.out.print("Choose account type: ");
    type = sc.nextInt();
    System.out.print("Enter amount to withdraw: ");
    amount = sc.nextDouble();
    if (type == 1 && savingAcc != null) {
        savingAcc.withdraw(amount);
    } else if (type == 2 && currentAcc != null) {
        currentAcc.withdraw(amount);
    } else {
```

```
                System.out.println("Account not created yet!");
            }
            break;
        case 6:
            System.out.println("Thank you for using our banking system!");
            sc.close();
            return;
        default:
            System.out.println("Invalid choice! Please try again.");
      }
    }
  }
}
```

---

c) Write a program in java to implement the abstract class and override the methods of the abstract class in the provided derived classes.

Class : Shape2D

Members: type

Abstract methods: area(), perimeter()

Class: Circle (inherits Shape2D)

Members: center, radius

Methods: area(), perimeter(), print()

Class: Triangle (inherits Shape2D)

Members: base, height

Methods: area(), perimeter(), print()

CODE :-

```
import java.util.Scanner;
```

```
abstract class Shape2D {
    String type;

    Shape2D(String type) {
        this.type = type;
    }
    abstract double area();
    abstract double perimeter();
}
class Circle extends Shape2D {
    Point center;
    double radius;
    Circle(Point center, double radius) {
        super("Circle");
        this.center = center;
        this.radius = radius;
    }
    @Override
    double area() {
        return Math.PI * radius * radius;
    }
    @Override
    double perimeter() {
        return 2 * Math.PI * radius;
    }
    void print() {
        System.out.println("Circle Details:");
        System.out.println("Type: " + type);
        System.out.println("Center: (" + center.x + "," + center.y + ")");
```

```
        System.out.println("Radius: " + radius);
        System.out.println("Area: " + area());
        System.out.println("Perimeter: " + perimeter());
    }
}
class Triangle extends Shape2D {
    double base;
    double height;
    double side1, side2;
    Triangle(double base, double height, double side1, double side2) {
        super("Triangle");
        this.base = base;
        this.height = height;
        this.side1 = side1;
        this.side2 = side2;
    }
    @Override
    double area() {
        return 0.5 * base * height;
    }
    @Override
    double perimeter() {
        return base + side1 + side2;
    }
    void print() {
        System.out.println("Triangle Details:");
        System.out.println("Type: " + type);
        System.out.println("Base: " + base);
        System.out.println("Height: " + height);
```

```
        System.out.println("Side 1: " + side1);
        System.out.println("Side 2: " + side2);
        System.out.println("Area: " + area());
        System.out.println("Perimeter: " + perimeter());
    }
}
class Point {
    double x, y;
    Point(double x, double y) {
        this.x = x;
        this.y = y;
    }
}
public class Q3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Circle circle = null;
        Triangle triangle = null;
        while(true) {
            System.out.println("\nShape Operations Menu:");
            System.out.println("1. Create Circle");
            System.out.println("2. Create Triangle");
            System.out.println("3. Calculate Area");
            System.out.println("4. Calculate Perimeter");
            System.out.println("5. Print Shape Details");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();
            switch(choice) {
```

```
case 1:
    System.out.println("Enter Circle Details:");
    System.out.print("Enter center x-coordinate: ");
    double x = sc.nextDouble();
    System.out.print("Enter center y-coordinate: ");
    double y = sc.nextDouble();
    System.out.print("Enter radius: ");
    double radius = sc.nextDouble();
    circle = new Circle(new Point(x, y), radius);
    System.out.println("Circle created successfully!");
    break;
case 2:
    System.out.println("Enter Triangle Details:");
    System.out.print("Enter base: ");
    double base = sc.nextDouble();
    System.out.print("Enter height: ");
    double height = sc.nextDouble();
    System.out.print("Enter side 1: ");
    double side1 = sc.nextDouble();
    System.out.print("Enter side 2: ");
    double side2 = sc.nextDouble();
    triangle = new Triangle(base, height, side1, side2);
    System.out.println("Triangle created successfully!");
    break;
case 3:
    System.out.println("\n1. Circle\n2. Triangle");
    System.out.print("Choose shape type: ");
    int type = sc.nextInt();
    if (type == 1 && circle != null) {
```

```
            System.out.println("Circle Area: " + circle.area());
        } else if (type == 2 && triangle != null) {
            System.out.println("Triangle Area: " + triangle.area());
        } else {
            System.out.println("Shape not created yet!");
        }
        break;
    case 4:
        System.out.println("\n1. Circle\n2. Triangle");
        System.out.print("Choose shape type: ");
        type = sc.nextInt();
        if (type == 1 && circle != null) {
            System.out.println("Circle Perimeter: " + circle.perimeter());
        } else if (type == 2 && triangle != null) {
            System.out.println("Triangle Perimeter: " + triangle.perimeter());
        } else {
            System.out.println("Shape not created yet!");
        }
        break;
    case 5:
        System.out.println("\n1. Circle\n2. Triangle");
        System.out.print("Choose shape type: ");
        type = sc.nextInt();
        if (type == 1 && circle != null) {
            circle.print();
        } else if (type == 2 && triangle != null) {
            triangle.print();
        } else {
            System.out.println("Shape not created yet!");
```

```
                }
                break;
            case 6:
                System.out.println("Exiting program...");
                sc.close();
                return;

            default:
                System.out.println("Invalid choice! Please try again.");
        }
    }
  }
}
```