

OOPS

5th Semester - 2023

➤ Assignment no : 1

➤ Problem Statement :

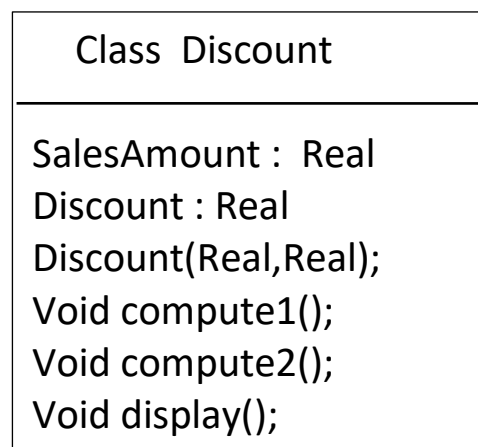
Write a java program to create a class Discount containing SalesAmount as a data member.

* SalesAmount should be given as input using appropriate input() method / Constructor defined in the class.

In the class, defined 2 method viz. compute1() and compute2(), those will compute the discount to be given using if-else constructor and ternary operator respectively following the bellow mentioned conditions :

- 1) if SalesAmount is less than INR. 10000/- , no discount given
- 2) if SalesAmount is greater equal 10000/- and less than INR 20000/- , 3% discount will be given.
- 3) if SalesAmount is greater euual INR 20000/- and less than INR 30000/- , 5% discount will be given.
- 4) if the SalesAmount is greater equal INR 40000/-, 10% discount will be given.
- 5) And display() function to display the calculated discount.

➤ Class Diagram :



➤ Source Code :

```
import java.util.Scanner;
class Discount{
    float salesamount;
    float discount;
    void putSA(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the salesAmount : ");
        salesamount = sc.nextFloat();
    }
    public void compute1(){
        if(salesamount<10000) discount = 0;
        else if(salesamount>=10000 && salesamount<20000) discount = (salesamount*3)/100;
        else if(salesamount>=20000 && salesamount<40000) discount = (salesamount*5)/100;
        else if(salesamount>=40000) discount = (salesamount*10)/100;
    }
    public void compute2(){
        discount = (salesamount<10000) ? 0 :
            (salesamount>=10000 && salesamount<20000) ? (salesamount*3)/100 :
            (salesamount>=20000 && salesamount<40000) ? (salesamount*5)/100 :
            (salesamount*10)/100;
    }
    public void display(){
        System.out.println("The discount for pursuing "+salesamount+" is : "+discount);
        System.out.println("You have to pay : "+(salesamount-discount));
    }
    public static void main(String[] args){
        System.out.println("Welcome to Hello shop!!!");
        int j=1;
        Scanner sc = new Scanner(System.in);
        do{
            Discount d1 = new Discount();
            d1.putSA();
            d1.compute1();
            d1.display();
            System.out.print("If u want to continue press 1 else anything : ");
            j=sc.nextInt();
        }while(j==1);
        sc.close();
    }
}
```

➤ Output :

Welcome to Hello shop!!!
Enter the salesAmount : 15000
The discount for pursuing 15000.0 is : 450.0
You have to pay : 14550.0
If u want to continue press 1 else anything : 1
Enter the salesAmount : 25000
The discount for pursuing 25000.0 is : 1250.0
You have to pay : 23750.0
If u want to continue press 1 else anything : 1
Enter the salesAmount : 34000
The discount for pursuing 34000.0 is : 1700.0
You have to pay : 32300.0
If u want to continue press 1 else anything : 1
Enter the salesAmount : 47000
The discount for pursuing 47000.0 is : 4700.0
You have to pay : 42300.0
If u want to continue press 1 else anything : 5

➤ Assignment no : Extra(1.1)

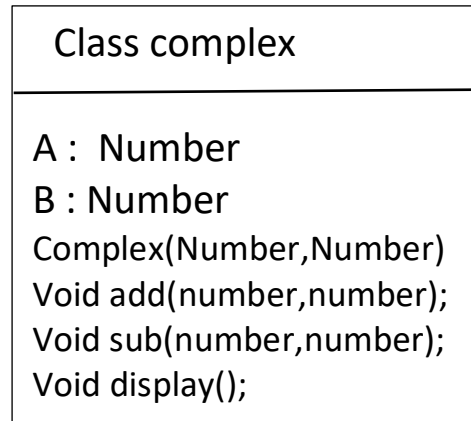
➤ Problem Statement :

Create a class named Complex containing data members a and b and member methods add(), sub(), Complex (int, int), display(). Create objects of Complex class and perform the addition & subtraction and display the result in each case.

[Hint: For addition of 2 Complex objects, use the formula: $(a_1 + ib_1) + (a_2 + ib_2) = (a_1 + a_2) + i(b_1 + b_2)$]

[N.B. Assume suitable return types of methods.]

➤ Class Diagram :



➤ Source Code :

```
import java.util.Scanner;
class complex {
    int a,b;
    complex(int A,int B){
        a = A;
        b = B;
    }
    void add(int A,int B){
        a +=A;
        b +=B;
    }
    void sub(int A,int B){
        a -=A;
        b -=B;
    }
    void display(){
        System.out.println(" "+a + "+" + b+"i");
    }
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the real and imaginary part : ");
        complex c1 = new complex(sc.nextInt(),sc.nextInt());
        System.out.print("Initially c1 : ");
        c1.display();
        System.out.print("After addition of c1 :");
        c1.add(3,8);
```

```

    c1.display();
    System.out.print("After Substuction of c1 :");
    c1.sub(2,7);
    c1.display();
    sc.close();
}
}

```

➤ Output :

Enter the real and imaginary part : 5 6

Initially c1 : 5+6i

After addition of c1 : 8+14i

After Substuction of c1 : 6+7i

➤ Assignment no : 2

➤ Problem Statement :

create a class Account containing data members Acc_No, C_Name , Contact_No as data members and a constructor for creating Account objects and show() method for displaying the data members.

create the following 3 sub-classes, namel :

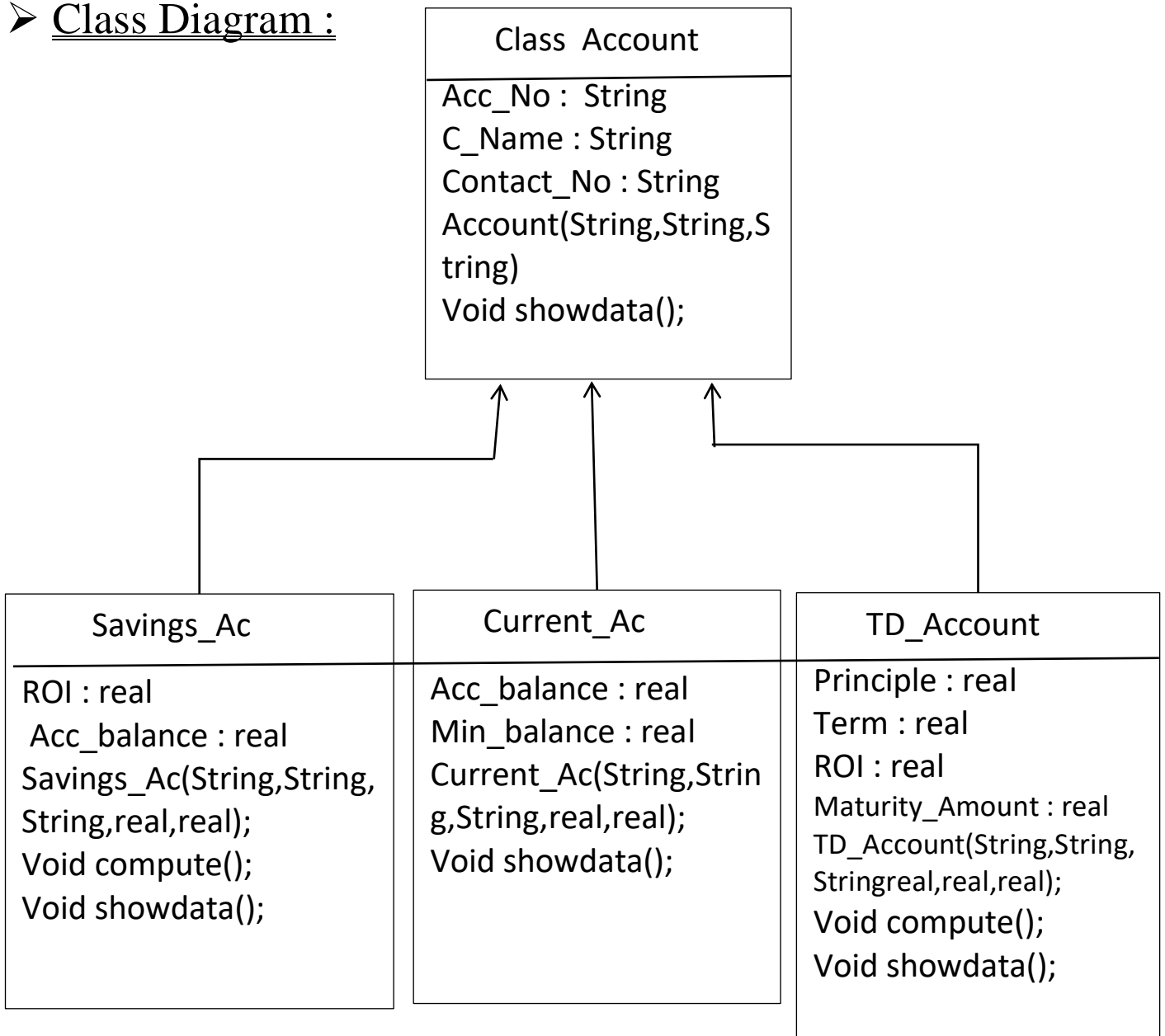
- *Savings_Ac containing specialized features viz. ROI, Acc_Balance

- *Current_Ac containing speialized features viz. Acc_Balance, Min_Balance

- *TD_Ac containing specialized features viz. Principal, Term, ROI, Maturity_Balance

Define suitable constructors in all of the above classes for creating objects and showdata() methods for displaying the data members. Also, define compute() method for different computeations in all classes such as interest calclations etc.

➤ Class Diagram :



➤ Source Code :

```
import java.lang.Math;
class Savings_Ac extends Account{
    private float ROI, Acc_Balance;
    Savings_Ac(String ANo, String Cname, String Cno, float roi, float ABalance){
        super(ANo, Cname, Cno);
        ROI = roi;
        Acc_Balance = ABalance;
    }
    public void compute(){
        System.out.println("Savings account details : ");
        System.out.println("Account Balance : "+Acc_Balance);
        Acc_Balance = Acc_Balance + (Acc_Balance*ROI)/100;
    }
}
```

```

public void showdata(){
    super.showdata();
    System.out.println("ROI : "+ROI);
    System.out.println("Updated Account Balance is : "+Acc_Balance);
}
}

class Current_Ac extends Account{
    private float Acc_Balance,Min_Balance;
    Current_Ac(String ANo,String Cname,String Cno,float ABalance,float minbal){
        super(ANo,Cname,Cno);
        Acc_Balance = ABalance;
        Min_Balance = minbal;
    }
    public void showdata(){
        System.out.println("Current Account details : ");
        super.showdata();
        System.out.println("Account Balance : "+Acc_Balance);
        System.out.println("Min balance need : "+Min_Balance);
        if(Acc_Balance<Min_Balance) System.out.println("Insufficient balance in the account.");
        else System.out.println("sufficient balance in the account.");
    }
}

class TD_Ac extends Account{
    private double Principal,Term,ROI,Maturity_Balance;
    TD_Ac(String ANo,String Cname,String Cno,double prin,double trm,double roi){
        super(ANo,Cname,Cno);
        Principal = prin;
        Term = trm;
        ROI = roi;
    }
    public void compute(){
        System.out.println("TD_Account drtails : ");
        Maturity_Balance = Principal * Math.pow(1 + ROI/100, Term);
    }
    public void showdata(){
        super.showdata();
        System.out.println("Principal amount is : "+Principal);
        System.out.println("Total term is : "+Term);
        System.out.println("Rate of interests is : "+ROI);
        System.out.println("Maturity amount is : "+Maturity_Balance);
    }
}

```

```

class Account{
    String Acc_No,C_Name,Contact_No;
    Account(String an,String cname,String cno){
        Acc_No = an;
        C_Name = cname;
        Contact_No = cno;
    }
    void showdata(){
        System.out.println("Account No : "+Acc_No);
        System.out.println("Customer name : "+C_Name);
        System.out.println("Contact No : "+Contact_No);
    }
    public static void main(String[] args){
        Savings_Ac S1 = new Savings_Ac("964827631986438","MS Dhoni","+91
9623159468",5,100000);
        S1.compute();
        S1.showdata();
        System.out.println();

        Current_Ac c1 = new Current_Ac("466491305687915","Virat Kohli","+92
9647125983",11000,10000);
        c1.showdata();
        System.out.println();

        TD_Ac t1 = new TD_Ac("656898447979434","Rohit Sharma","+1
6438109826",20000,2,10);
        t1.compute();
        t1.showdata();
    }
}

```

➤ Output :

Savings account details :
 Account Balance : 100000.0
 Account No : 964827631986438
 Customer name : MS Dhoni
 Contact No : +91 9623159468
 ROI : 5.0
 Updated Account Balance is : 105000.0

Current Account details :

Account No : 466491305687915

Customer name : Virat Kohli

Contact No : +92 9647125983

Account Balance : 11000.0

Min balance need : 10000.0

sufficient balance in the account.

TD_Account drtails :

Account No : 656898447979434

Customer name : Rohit Sharma

Contact No : +1 6438109826

Principal amount is : 20000.0

Total term is : 2.0

Rate of interests is : 10.0

Maturity amount is : 24200.000000000004

➤ Assignment no : Extra(2.1)

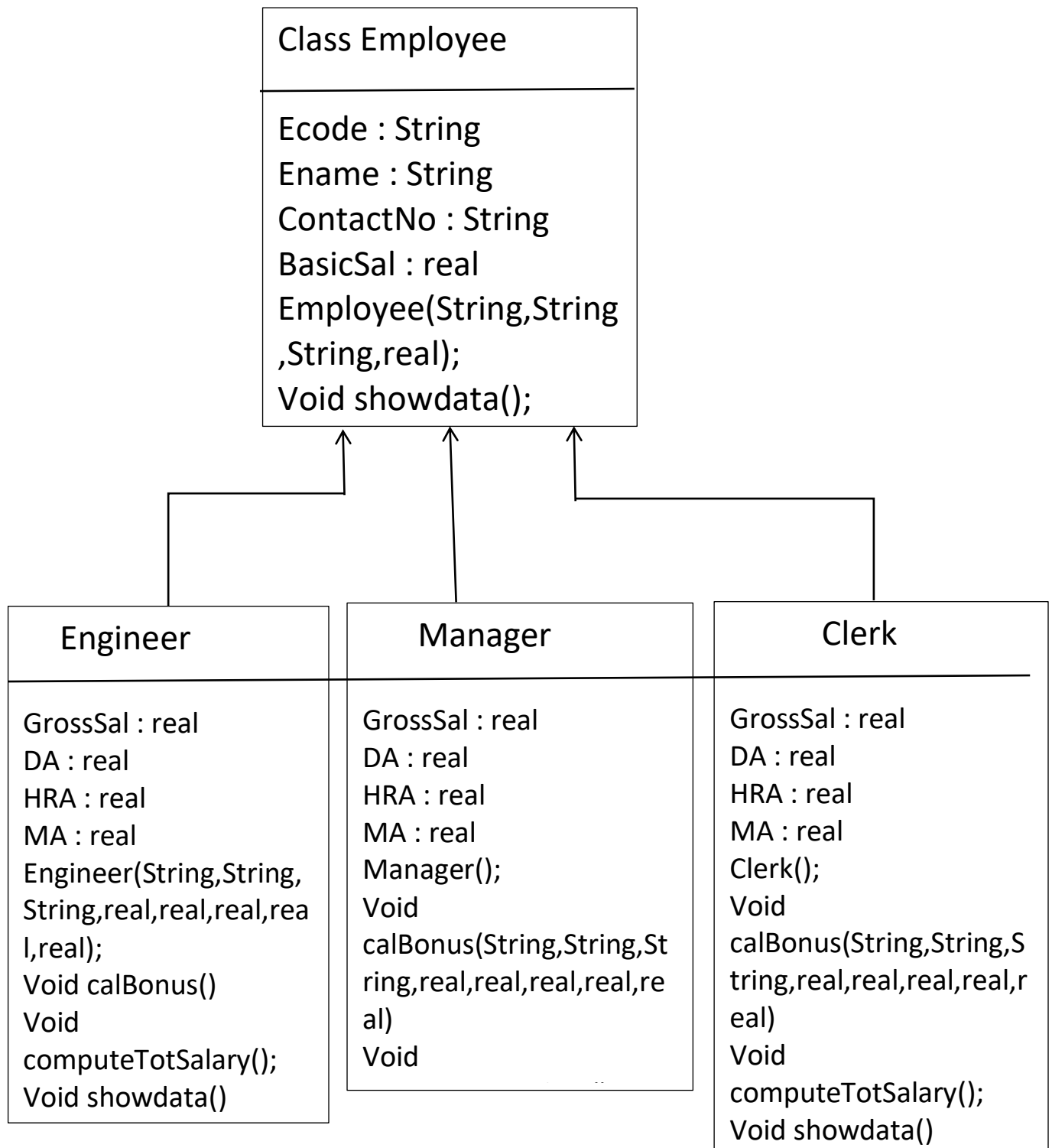
➤ Problem Statement :

Create a base class named Employee containing data members ECode, Ename, ContactNo, BasicSal, GrossSal, Dept and methods like computeTotSalary(), calcBonus(), showdata(). Different types of employees like Manager, Engineer and Clerk may have their implementations of the methods present in base class Employee. Each type of employee may have its logic in its class, e,g, if calcBonus() is present for a specific employee type, only that method would be invoked to provide hike in salary(by addition of Bonus). Use constructors in every class for creation of objects. Implement using suitable Java Code.

|N.B. 1.Assume suitable data-types of data members and return (types of methods.

2. GrossSal = BasicSal+DA+HRA+MA where DA=75% af BasicSal, HRA=15% of BasicSal, MA-10% of BasicSal)

➤ Class Diagram :



➤ Source Code :

```
class Engineer extends Employee{
    double GrossSal;
    double DA,HRA,MA;
    Engineer(String ECo,String Ena,String ConNo,String Dpt,double BaSal){
        super(ECo, Ena, ConNo, Dpt,BaSal);
    }
    void colcBonus(){
        DA = BasicSal * 0.75f;
        HRA = BasicSal * 0.15f;
        MA = BasicSal * 0.1f;
    }
    void computeTotsalary(){
        GrossSal = BasicSal+DA+HRA+MA;
    }
    void showdata(){
        super.showdata();
        System.out.println("Bonus is : " + (DA+HRA+MA));
        System.out.println("Total / Gross salary is : "+GrossSal);
    }
}

class Manager extends Employee{
    double GrossSal;
    double DA,HRA,MA;
    Manager(String ECo,String Ena,String ConNo,String Dpt,double BaSal){
        super(ECo, Ena, ConNo, Dpt,BaSal);
    }
    void colcBonus(){
        DA = BasicSal * 0.75f;
        HRA = BasicSal * 0.15f;
        MA = BasicSal * 0.1f;
    }
    void computeTotsalary(){
        GrossSal = BasicSal+DA+HRA+MA;
    }
    void showdata(){
        super.showdata();
        System.out.println("Bonus is : " + (DA+HRA+MA));
        System.out.println("Total / Gross salary is : "+GrossSal);
    }
}

class Clerk extends Employee{
    double GrossSal;
```

```

double DA,HRA,MA;
Clerk(String ECo,String Ena,String ConNo,String Dpt,double BaSal){
    super(ECo, Ena, ConNo, Dpt,BaSal);
}
void colcBonus(){
    DA = BasicSal * 0.75f;
    HRA = BasicSal * 0.15f;
    MA = BasicSal * 0.1f;
}
void computeTotsalary(){
    GrossSal = BasicSal+DA+HRA+MA;
}
void showdata(){
    super.showdata();
    System.out.println("Bonus is : " + (DA+HRA+MA));
    System.out.println("Total / Gross salary is : "+GrossSal);
}
}
class Employee {
    double BasicSal;
    String Ename,ECode,ContactNo,Dept;
    Employee(String ECo,String Ena,String ConNo,String Dpt,double BaSal){
        ECode = ECo;
        Ename =Ena;
        ContactNo = ConNo;
        Dept = Dpt;
        BasicSal = BaSal;
    }
    void showdata(){
        System.out.println("\nEmployee Ecode - " + ECode);
        System.out.println("Name : "+Ename);
        System.out.println("Contact Number : "+ContactNo);
        System.out.println("Depertment name : "+Dept);
        System.out.println("Basic salary : "+BasicSal);
    }
    public static void main(String[] args) {
        Engineer E1 = new Engineer("10000005", "Santu Mandal", "9265463092", "SDE", 95000);
        E1.colcBonus();
        E1.computeTotsalary();
        E1.showdata();

        Manager E2 = new Manager("10000009", "Joyes Pramanik", "96315893437",
"Consultant",80000);
        E2.colcBonus();
    }
}

```

```
E2.computeTotsalary();
E2.showdata();

Clerk E3 = new Clerk("10000007","Ashok Samanta", "9613558496", "File Management",
45000);
E3.colcBonus();
E3.computeTotsalary();
E3.showdata();
}
}
```

➤ Output :

Employee Ecode - 10000005
Name : Santu Mandal
Contact Number : 9265463092
Depertment name : SDE
Basic salary : 95000.0
Bonus is : 95000.00070780516
Total / Gross salary is : 190000.00070780516

Employee Ecode - 10000009
Name : Joyes Pramanik
Contact Number : 96315893437
Depertment name : Consultant
Basic salary : 80000.0
Bonus is : 80000.00059604645
Total / Gross salary is : 160000.00059604645

Employee Ecode - 10000007
Name : Ashok Samanta
Contact Number : 9613558496
Depertment name : File Management
Basic salary : 45000.0
Bonus is : 45000.00033527613
Total / Gross salary is : 90000.00033527613

➤ Assignment no : 3

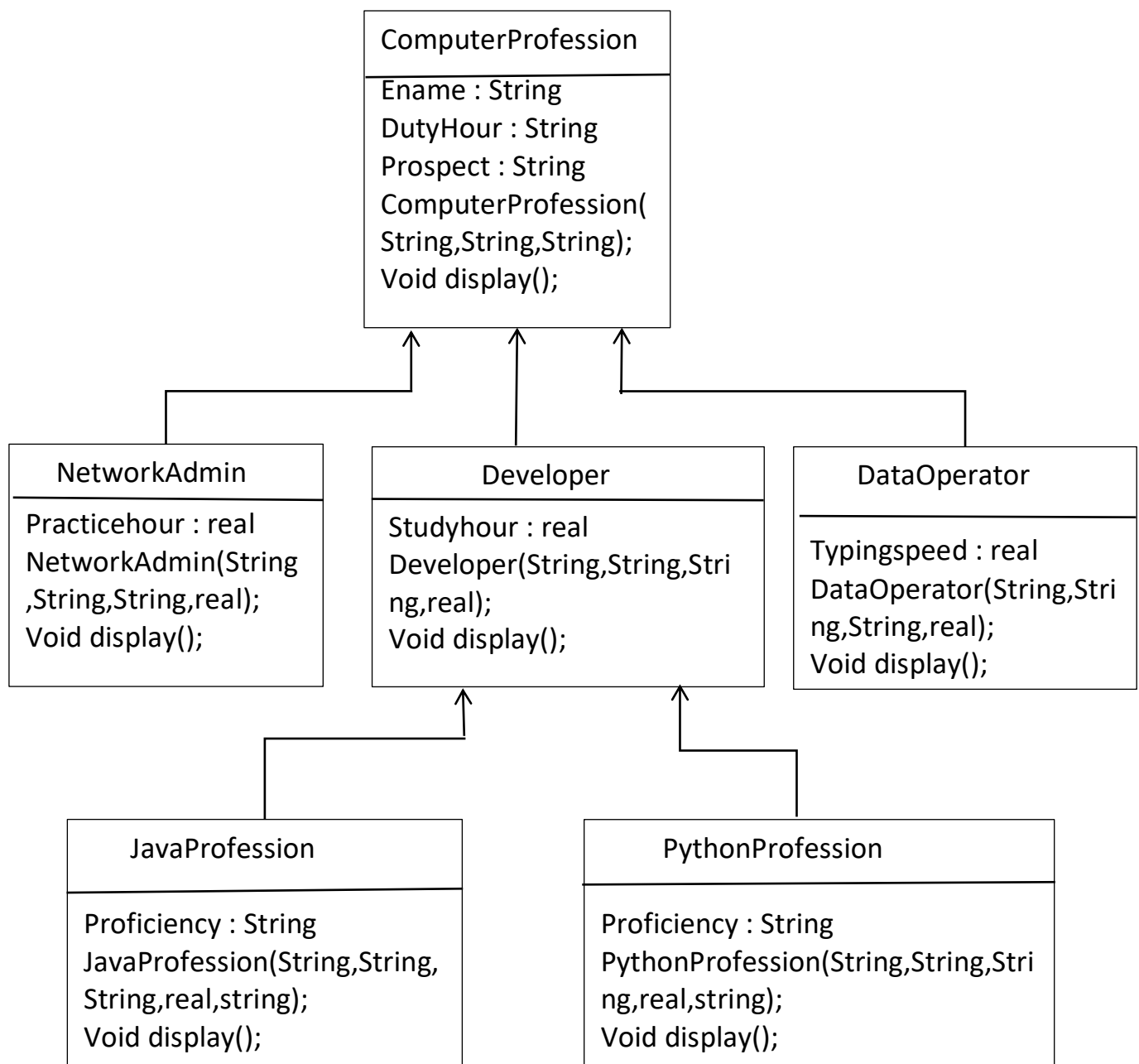
➤ Problem Statement :

Create a class ComputerProfessional containing EName, DutyHour and Prospect(Domain of values are "Excellent", "Good" and "Fair") as data members. Create 3 different child classes namely Developer, NetworkAdmin and DataOperator using ComputerProfessional as the super class. Include StudyHour as data member in Developer Class, PracticeHours as data member in NetworkAdmin Class, TypingSpeed as data member in DataOperator Class.

Create two more sub-classes of Developer Class namely JavaProfessional and PythonProfessional having ProficiencyLevel(Domain of values are "Beginner", "Intermediate" and "Pro") as specialized feature.

Use constructors to create the objects for all classes. Develop the appropriate display() methods in all of the classes to show the appropriate data members.

➤ Class Diagram :



➤ Source Code :

```
class Developer extends ComputerProfessional{
    private float studyhour;
    Developer(String name,String DHour,String pros,float SH){
        super(name,DHour,pros);
        studyhour = SH;
    }
    public void display(){
        super.display();
        System.out.println("StudyHour is "+studyhour);
    }
}

class Network extends ComputerProfessional{
    private float practicehour;
    Network(String name,String DHour,String pros,float pH){
        super(name,DHour,pros);
        practicehour = pH;
    }
    public void display(){
        super.display();
        System.out.println("Practice Hour is : "+practicehour);
        System.out.println("\n");
    }
}

class DataOperator extends ComputerProfessional{
    private float TypingSpeed;
    DataOperator(String name,String DHour,String pros,float TS){
        super(name,DHour,pros);
        TypingSpeed = TS;
    }
    public void display(){
        super.display();
        System.out.println("Typing Speed is : "+TypingSpeed);
        System.out.println("\n");
    }
}

class Javaprofession extends Developer{
    private String proficiency;
    Javaprofession(String name,String DHour,String pros,float SH,String profi){
        super(name,DHour,pros,SH);
        proficiency = profi;
    }
    public void display(){
        super.display();
```

```

        System.out.println("Proficiency level is : "+proficiency);
        System.out.println("\n");
    }
}

class Pythonprofession extends Developer{
    private String proficiency;
    Pythonprofession(String name,String DHour,String pros,float SH,String profi){
        super(name,DHour,pros,SH);
        proficiency = profi;
    }
    public void display(){
        super.display();
        System.out.println("Proficiency level is : "+proficiency);
        System.out.println("\n");
    }
}

class ComputerProfessional{
    private String Ename,DutyHour,prospect;
    ComputerProfessional(String name,String DHour,String pros){
        Ename = name;
        DutyHour = DHour;
        prospect= pros;
    }
    public void display(){
        System.out.println("The information of the employee is : ");
        System.out.println("Name : "+Ename);
        System.out.println("Duty hour : "+DutyHour);
        System.out.println("Prospective : "+prospect);
    }
    public static void main(String[] args){

        Network A1 = new Network("Arpan Hutait", "10", "fair", 3);
        A1.display();

        DataOperator A2 = new DataOperator("Sourav Das", "16", "Excellent", 40);
        A2.display();

        Javaprofession A3 = new Javaprofession("Partha Maity", "9", "Fair", 4, "Intermediate");
        A3.display();

        Pythonprofession A4 = new Pythonprofession("Jagadish Sau", "15", "Excellent", 6, "Pro");
        A4.display();
    }
}

```


➤ Output :

The information of the employee is :

Name : Arpan Hutait

Duty hour : 10

Prospective : fair

Practice Hour is : 3.0

The information of the employee is :

Name : Sourav Das

Duty hour : 16

Prospective : Excellent

Typing Speed is : 40.0

The information of the employee is :

Name : Partha Maity

Duty hour : 9

Prospective : Fair

StudyHour is 4.0

Proficiency level is : Intermediate

The information of the employee is :

Name : Jagadish Sau

Duty hour : 15

Prospective : Excellent

StudyHour is 6.0

Proficiency level is : Pro

➤ Assignment no : 4

➤ Problem Statement :

Create a class Arithmetic containing following data members

a: integer b: float c: char d: double.

and the following member functions

- Arithmetic (int, float, char, double); //Parameterized constructor
- void add(int, double); // Polymorphic add function for adding to a and d
// data members of the Object
- void add(int, float, double); //Polymorphic add function
- void add(float, int, double); //Polymorphic add function
- Arithmetic add(Arithmetic); // Polymorphic add function for adding 2 objects
- void display(); // Member function for showing the data members

Declare objects of the class Arithmetic as per requirement and invoke the member functions. Comment on the performance of all the polymorphic add() functions.

➤ Class Diagram :

Arithmetic

```
a : Number  
b : Real  
c : Character  
d : Real  
Arithmetic(Number,Real,Character,Real)  
Void add (Number, Real)  
Void add (Number, Real, Real)  
Void add (Real, Number, Real)  
Arithmetic add (Arithmetic)  
void display ();
```

➤ Source Code :

```
class Arithmetic{  
    private int a;  
    private float b;  
    private char c;  
    private double d;  
    Arithmetic(int A,float B,char C,double D) {  
        a = A;  
        b = B;  
        c = C;  
        d = D;  
    }  
    void add(int i,double k) {  
        a+=i;  
        d+=k;  
    }  
    void add(int i,float j, double k) {  
        a+=i;  
        b+=j;  
        d+=k;  
    }  
    void add(float j, int i, double k) {  
        a+=i;  
        b+=j;  
        d+=k;  
    }  
    Arithmetic add(Arithmetic p) {  
        Arithmetic Q = new Arithmetic(0, 0, 'a', 0);
```

```

        Q.a = this.a + p.a;
        Q.b = this.b + p.b;
        Q.c = this.c;
        Q.d = this.d + p.d;
        return Q;
    }
    void display() {
        System.out.println( "Interger : " + a + " , float : " + b + " ,
character : " + c + " and double : " + d );
    }
    public static void main(String[] args){
        Arithmetic A = new Arithmetic(2, 5.5f, 's', 123.45);
        Arithmetic B = new Arithmetic(5, 3.9f, 'n', 321.54);
        System.out.println( "Initially object A ==> " );
        A.display();
        System.out.println( "\nInitial object B ==> " );
        B.display();
        System.out.println( "\nAfter some operation on object A ==> " );
        A.add(2, 36.56);
        A.display();
        A.add(3, 2.6f, 2445.364);
        A.display();
        A.add(2.36f, 8, 236.75);
        A.display();
        Arithmetic C = A.add(B);
        System.out.println( "\nFinally object A , B and C ==> " );
        A.display();
        B.display();
        C.display();
    }
}

```

➤ Output :

D:\programming language\OOP - JAVA 2023\5th sem 2023 college\4th_Assignment>javac Arithmetic.java

D:\programming language\OOP - JAVA 2023\5th sem 2023 college\4th_Assignment>java Arithmetic
Initially object A ==>

Integer : 2 , float : 5.5 , character : s and double 123.45

Initial object B ==>

Integer : 5 , float : 3.9 , character : n and double 321.54

After some operation on object A ==>

Integer : 4 , float : 5.5 , character : s and double 160.01

Integer : 7 , float : 8.1 , character : s and double 2605.374

Integer : 15 , float : 10.46 , character : s and double 2842.124

Finally object A , B and C ==>

Integer : 15 , float : 10.46 , character : s and double 2842.124

Integer : 5 , float : 3.9 , character : n and double 321.54

Integer : 20 , float : 14.360001 , character : s and double 3163.6639999999998

Assignment : Extra (4.1)

➤ Problem Statement :

Create a class Time containing the data members hr(int type), min(int type), sec(double type) and member functions ----

```
Time(int,int,double),          //Parameterized constructor
showtime(),                   //Member function for showing the data members
add_time(int),                 // Polymorphic add function for adding min to that of Time object
add_time(int,int),            // Polymorphic add function for adding hr & min to that of Time
                                object
add_time(double),             // Polymorphic add function for adding sec to that of Time object
add_time(Time),               // Polymorphic add function for adding Time object to another Time
                                object
```

Create objects of Time class and perform their addition.

➤ Class Diagram :

Time

hr: Number
min : Number
sec : Real
Time (Number, Number, Real)
Void Showtime (),
Void add_time(Number)
Void Add_time(Number,Number);
Void add_time(Real)
Void add_time(Time);

➤ SourceCode :

```
class Time {
    private int hr;
    private int min;
    private double sec;
    Time(int HR,int MIN,double SEC){
        hr = HR;
        min = MIN;
        sec = SEC;
    }
    void add_time(int m){
        min+=m;
        if(min>=60){
            hr +=1;
            min%=60;
        }
    }
    void add_time(int h,int m){
        hr+=h;
        min+=m;
        if(min>=60){
            hr +=1;
            min%=60; }
    }
    void add_time(double s){
        sec+=s;
        if(s>=60){
            min +=1;
            sec%=60;
        }
    }
    Time add_time(Time k){
```

```

Time T = new Time(0,0,0);
T.sec = this.sec + k.sec;
T.min = this.min +k.min;
T.hr = this.hr + k.hr;
if(T.sec>=60){
    T.min +=1;
    T.sec%=60;
}
if(T.min>=60){
    T.hr +=1;
    T.min%=60;}
return T;
}
void showtime(){
    System.out.println("Current time is => " + hr + " : " + min + " : "+ sec); }
public static void main(String[] args){
    Time t1 = new Time(12, 5, 20.06);
    Time t2 = new Time(2, 56, 45);
    System.out.println("Initially t1 object : ");
    t1.showtime();
    System.out.println("\nInitially t2 object : ");
    t2.showtime();
    System.out.println("\nAfter some operation of t1 object : ");
    t1.add_time(50);
    t1.showtime();
    t1.add_time(2,40);
    t1.showtime();
    t1.add_time(37.3);
    t1.showtime();
    Time t3 = t1.add_time(t2);
    System.out.println("\nFinally t1 , t2 and t3 object : ");
    t1.showtime();
    t2.showtime();
    t3.showtime();
}
}

```

➤ Output :

D:\programming language\OOP - JAVA 2023\5th sem 2023 college\4th_Assignment>javac Time.java

D:\programming language\OOP - JAVA 2023\5th sem 2023 college\4th_Assignment>java Time

Initially t1 object :

Current time is => 12 : 5 : 20.06

Initially t2 object :

Current time is => 2 : 56 : 45.0

After some operation of t1 object :

Current time is => 12 : 55 : 20.06

Current time is => 15 : 35 : 20.06

Current time is => 15 : 35 : 57.36

Finally t1 , t2 and t3 object :

Current time is => 15 : 35 : 57.36

Current time is => 2 : 56 : 45.0

Current time is => 18 : 32 : 42.36

Assignment : 5

➤ Problem Statement :

Create a class named Student containing S Name, Roll No, Batch, Year of Adm, Stream as datamembers. Write different types of constructors for creating Student Objects such as

Student(String, int)

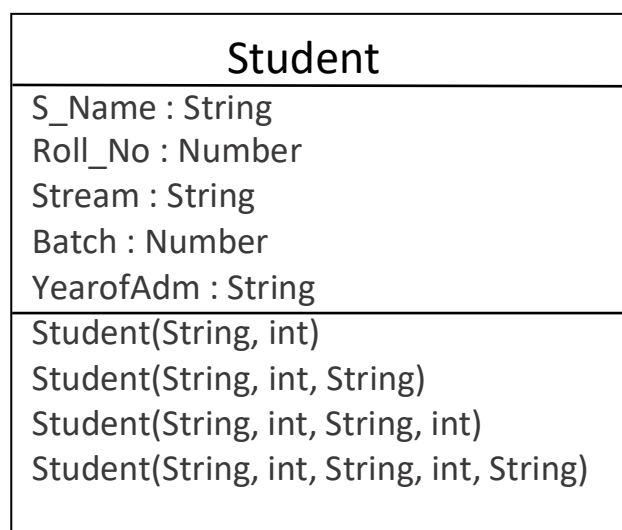
Student(String, int, String)

Student(String, int, String, int)

Student(String, int, String, int, String)

Assume default values for the data Use showdata(members for which values have not beenpassed.) method to display the data members for all objects so created.

➤ Class Diagram :



➤ Source Code :

```
class Student{
```

```

String S_Name, Batch, Stream;
int YearofAdm, Roll_No;
Student(String SN, int Roll){
    S_Name = SN;
    Roll_No = Roll;
}
Student(String SN, int Roll, String B){
    S_Name = SN;
    Roll_No = Roll;
    Batch = B;
}
Student(String SN, int Roll, String B, int Yr){
    S_Name = SN;
    Batch = B;
    Roll_No = Roll;
    YearofAdm = Yr;
}
Student(String SN, int Roll, String B, int yr, String Str){
    S_Name = SN;
    Batch = B;
    Roll_No = Roll;
    YearofAdm = yr;
    Stream = Str;
}
void show(){
    System.out.println("Student name is "+ S_Name+"\nRoll no : "+Roll_No+"\nStream : 
"+Stream+"\nBatch : "+Batch+"\nYear of Admission : "+YearofAdm+"\n\n");
}
}
class Test5 {
    public static void main(String[] args) {
        Student s1 = new Student("Tapas Maity", 45);
        Student s2 = new Student("Arpan Hutait", 5, "2024-29");
        Student s3 = new Student("Sourav Sasmal", 6, "2018-22", 2001);
        Student s4 = new Student("Rahul Patra", 3, "2022-27", 2003, "Medicine.");

        s1.show();
        s2.show();
        s3.show();
        s4.show();
    }
}

```



```
}  
}
```

➤ Output :

Student name is Tapas Maity

Roll no : 45

Stream : null

Batch : null

Year of Admission : 0

Student name is Arpan Hutait

Roll no : 5

Stream : null

Batch : 2024-29

Year of Admission : 0

Student name is Sourav Sasmal

Roll no : 6

Stream : null

Batch : 2018-22

Year of Admission : 2001

Student name is Rahul Patra

Roll no : 3

Stream : Medicine.

Batch : 2022-27

Year of Admission : 2003

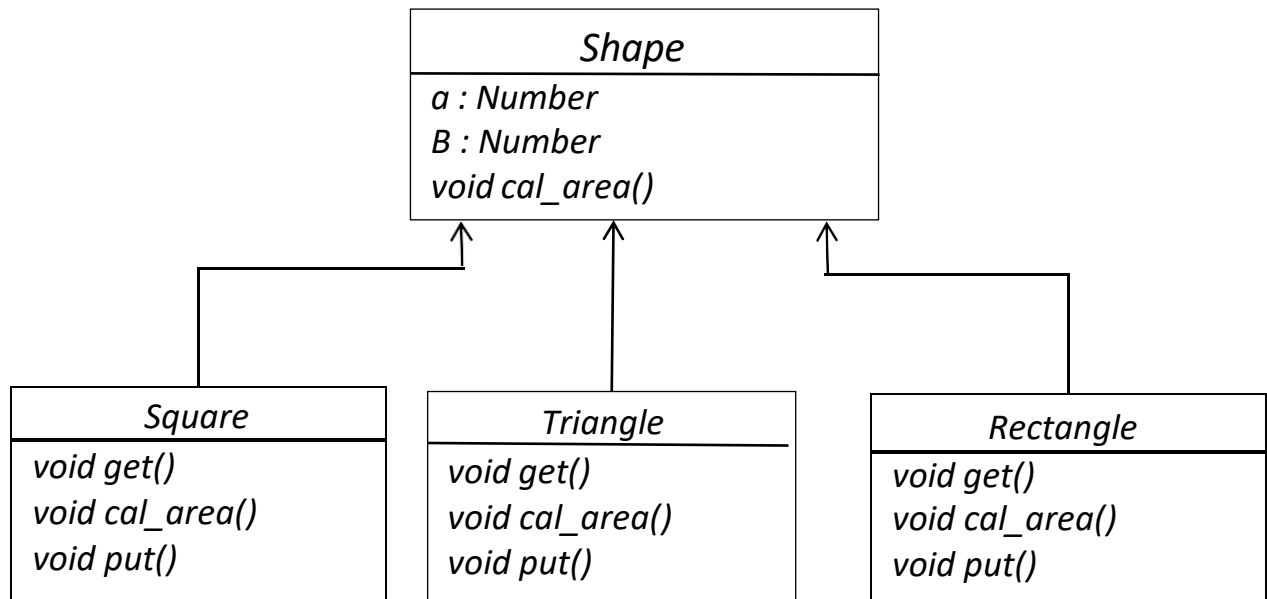
?

Assignment : 6

➤ Problem Statement :

Write a Java program to create a class Shape" and 3 other classes named Square, Rectangle and Triangle derived from it, all containing a overridden method cal_area() to calculate area of a Square or a Rectangle or a Triangle.

Assume suitable data members(2 int type data members only) and member methods(get() and put()) in all classes. Also validate the inputs.



➤ Source Code :

```
import java.util.Scanner;

abstract class shape{
    int a,b;
    void cal_area(){};
}

class square extends shape{
    void get(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the side of the square : ");
        a = sc.nextInt();
        while(a<=0){
            System.out.print("Invalid Input!!!\nEnter the side value again : ");
            a = sc.nextInt();
        }
    }
}
```

```

void cal_area(){
    System.out.print("So the area of the square is : "+a*a);
}
void put(){
    System.out.println(" , Where entered side value of the square is : "+a+"\n");
}
}

class Triangle extends shape{
    void get(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the base of the triangle : ");
        a = sc.nextInt();
        while(a<=0){
            System.out.print("Invalid Input!!!\nEnter the base value again : ");
            a = sc.nextInt();
        }
        System.out.print("Enter the height of the triangle : ");
        b = sc.nextInt();
        while(b<=0){
            System.out.print("Invalid Input!!!\nEnter the height value again : ");
            b = sc.nextInt();
        }
    }
    void cal_area(){
        System.out.print("So the area of the square is : "+0.5*a*b);
    }
    void put(){
        System.out.println(" , Where entered base is : "+a + " and the height is : "+b+"\n");
    }
}

class Rectangle extends shape{
    void get(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the length of the rectangle : ");
        a = sc.nextInt();
        while(a<=0){
            System.out.print("Invalid Input!!!\nEnter the length value again : ");

```

```

        a = sc.nextInt();
    }
    System.out.print("Enter the weidth of the rectangle : ");
    b = sc.nextInt();
    while(b<=0){
        System.out.print("Invalid Input!!!\nEnter the weidth value again : ");
        b = sc.nextInt();
    }
}

void cal_area(){
    System.out.print("So the area of the rectangle is : "+a*b);
}

void put(){
    System.out.println(" , Where entered length is : "+a + " and the weidth is : "+b+"\n");
}
}

public class Test1 {
    public static void main(String args[]){
        square s1 = new square();
        s1.get();
        s1.cal_area();
        s1.put();
        Triangle t1 = new Triangle();
        t1.get();
        t1.cal_area();
        t1.put();
        Rectangle r1 = new Rectangle();
        r1.get();
        r1.cal_area();
        r1.put();
    }
}

```

➤ Output :

Enter the side of the square : -1

Invalid Input!!!

Enter the side value again : 2

So the area of the square is : 4 , Where entered side value of the square is : 2

Enter the base of the triangle : 0

Invalid Input!!!

Enter the base value again : 3

Enter the heirgt of the triangle : -4

Invalid Input!!!

Enter the height value again : 2

So the area of the square is : 3.0 , Where entered base is : 3 and the height is : 2

Enter the length of the rectangle : 0

Invalid Input!!!

Enter the length value again : 6

Enter the weidth of the rectangle : -8

Invalid Input!!!

Enter the weidth value again : 3

So the area of the rectangle is : 18 , Where entered length is : 6 and the weidth is : 3

Assignment : Extra(6.1)

➤ Problem Statement :

Create an abstract class 'Bird' containing the following members:

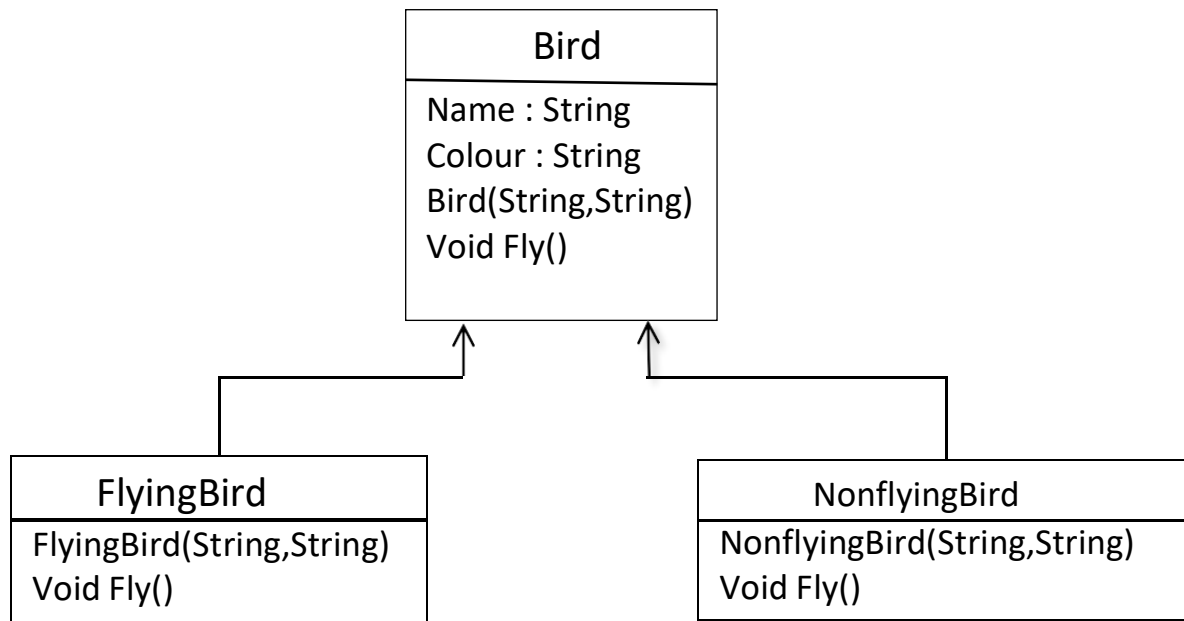
Name(String),Colour(String)

Bird(String,String)

Fly();

Create a derived class named 'FlyingBird' containing no new data members except the over-ridden Fly() method .Also create another derived class named 'NonFlying Bird' containing no new data member except the over-ridden Fly() method .Implement using a suitable java function.

➤ Class Diagram :



➤ Source Code :

```
class FlyingBird extends Bird{
    FlyingBird(String Na,String co){
        super(Na, co);
    }
    void Fly(){
        System.out.println("Hello It's a "+Name+" a flying bird of colour "+colour);
    }
}

class NonFlyingBird extends Bird{
    NonFlyingBird(String Na,String co){
        super(Na, co);
    }
    void Fly(){
        System.out.println("Hello It's a "+Name+" a Nonflying bird of colour "+colour+"");
    }
}

abstract class Bird {
    String Name,colour;
    Bird(String N,String c){
        Name = N;
        colour = c;
    }
    abstract void Fly();
    public static void main(String args[]){
        FlyingBird f1 = new FlyingBird("Parrot", "green");
```

```

    f1.Fly();
    NonFlyingBird f2 = new NonFlyingBird("Hean", "White");
    f2.Fly();
}
}

```

➤ Output :

Hello It's a Parrot a flying bird of colour green

Hello It's a Hean a Nonflying bird of colour White

Assignment : 7

➤ Problem Statement :

- Convert the following primitive data types into corresponding wrapper objects
int, char, float, double.
- Store null values in the int and float variables.
- Store the Wrapper Objects into primitive types.
- Considering an array of characters, check whether individual characters are of lowercase , uppercase, a digit or a whitespace character.

➤ Source Code :

```

class Main {
    public static void main(String[] args) {
        int i1 = 354;
        char c1 = 'J';
        float f1 = 9845.9562f;
        double d1 = 641.62;

        //a) Autoboxing ---> Primitive data type to Wrapper class
        Integer I2 = Integer.valueOf(i1);
        Character C2 = Character.valueOf(c1);
        Float F2 = Float.valueOf(f1);
        Double D2 = Double.valueOf(d1);
        System.out.println("Primitive to wrapper class conversion:");
        System.out.println("i2 = " + I2 + " , C2 = " + C2 + " , F2 = " + F2 + " , D2 = " + D2);

        //b) store null values in the int and float variables ---> not possible
    }
}

```

```

//int a = null;
//float b = null;
Integer A = null;
Float B = null;
System.out.println("\nWrapper object : A = "+A+" , B = "+B);

//c) Unboxing : Store Wrapper objects into primitive types
Integer I3 = 98;
Character C3 = 'A';
Float F3 = 164.64F;
Double D3 = 359.5644;

int i2 = I3.intValue();
char c2 = C3.charValue();
float f2 = F3.floatValue();
double d2 = D3.doubleValue();
System.out.println("\nWrapper class to Primitive conversion:");
System.out.println("i2 = " + i2 + " , C2 = " + c2 + " , f2 = " + f2 + " , d2 = " + d2);

//d) lowercase, uppercase, a digit or a whitespace character from char array
char[] arr = {'P','a','5',' ','@'};
System.out.println("\nIndividual characters are:");
for(int i=0;i<arr.length;i++){
    if(arr[i]>='A' && arr[i]<='Z'){
        System.out.println(arr[i] + " is uppercase.");
    }
    else if(arr[i]>='a' && arr[i]<='z'){
        System.out.println(arr[i] + " is lowercase.");
    }
    else if(arr[i]>=0 && arr[i]<=9){
        System.out.println(arr[i] + " is digit.");
    }
    else if(arr[i]==' '){
        System.out.println(arr[i] + " is whitespace.");
    }
    else{
        System.out.println(arr[i] + " is special character.");
    }
}
}
}

```

➤ Output :

Primitive to wrapper class conversion:

i2 = 354 , C2 = J , F2 = 9845.956 , D2 = 641.62

Wrapper object : A = null , B = null

Wrapper class to Primitive conversion:

i2 = 98 , C2 = A , f2 = 164.64 , d2 = 359.5644

Individual characters are:

P is uppercase.

a is lowercase.

5 is special character.

 is whitespace.

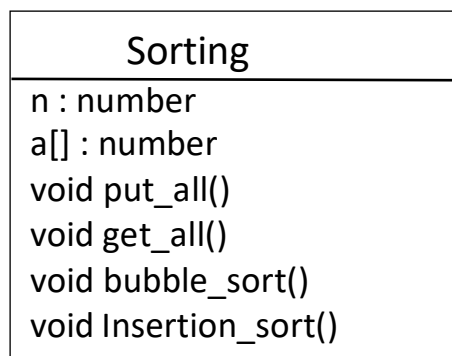
@ is special character.

Assignment : 8

➤ Problem Statement :

Implement Bubble sort/ Insertion sort by taking user input inside an array of integers. [Note: Use Object Oriented features only]

➤ Class Diagram :



➤ Source Code :

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

```

class sorting{
    int n;
    int a[] = new int[10];
    Scanner sc = new Scanner(System.in);

    void put_all(){
        System.out.print("\nEnter the number of elements : ");
        n = sc.nextInt();
        System.out.print("Enter "+n+ " elements : " );
        for(int i = 0;i<n;i++) a[i] = sc.nextInt();
    }
    void get_all(){
        System.out.print("Current elements are : ");
        for(int i=0;i<n;i++) System.out.print(a[i]+" ");
    }
    void Bubble_sort(){
        for(int i = 0;i<n-1;i++){
            for(int j=0;j<n-i-1;j++){
                if(a[j]>a[j+1]){
                    int t = a[j];
                    a[j]= a[j+1];
                    a[j+1] = t;
                }
            }
        }
    }
    void Insertion_sort(){
        for(int i = 1;i<n;i++){
            int t = a[i];
            int j = i-1;
            while(j>=0 && a[j]>t){
                a[j+1] = a[j];
                j--;
            }
            a[j+1] = t;
        }
    }
}

public class Test8 {
    public static void main(String[] args) {
        sorting s1 = new sorting();
    }
}

```

```
s1.put_all();
s1.Bubble_sort();
System.out.println("Using bubble sort -- ");
s1.get_all();

sorting s2 = new sorting();
s2.put_all();
s2.Insertion_sort();
System.out.println("Using insertion sort -- ");
s2.get_all();
}
}
```

➤ Output :

Enter the number of elements : 5

Enter 5 elements : 31 65 45 90 15

Using bubble sort --

Current elements are : 15 31 45 65 90

Enter the number of elements : 5

Enter 5 elements : 10 19 15 9 21

Using insertion sort --

Current elements are : 9 10 15 19 21

Assignment : 9

➤ Problem Statement :

Multiply two matrices by taking user input in two 2D Arrays of integers. Check whether the dimensions of matrices conform to the rule of Matrix Multiplication.

[Note: Use Object Oriented features only]

➤ Source Code :

```
import java.util.Scanner;
class Matrix{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("For first matrix:");
        System.out.print("Enter the number of rows: ");
        int rows1 = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols1 = scanner.nextInt();
        int[][] matrix1 = new int[rows1][cols1];
        System.out.println("Enter the elements:");
        inputMatrix(matrix1, scanner);

        System.out.println("\nFor second matrix: ");
        System.out.print("Enter the number of rows: ");
        int rows2 = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols2 = scanner.nextInt();
        int[][] matrix2 = new int[rows2][cols2];
        System.out.println("Enter the elements:");
        inputMatrix(matrix2, scanner);

        if (cols1 != rows2) {
            System.out.println("\nMatrix multiplication is not possible.");
        }
        else {
            int[][] resultMatrix = multiplyMatrices(matrix1, matrix2);
            System.out.println("\nResultant Matrix after multiplication:");
            displayMatrix(resultMatrix);
        }
        scanner.close();
    }
    private static void inputMatrix(int[][] matrix, Scanner scanner){
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[0].length; j++) {
                System.out.print("Enter element at position (" + (i + 1) + ", " + (j + 1) + "): ");
```

```

        matrix[i][j] = scanner.nextInt();
    }
}
private static int[][] multiplyMatrices(int[][] matrix1, int[][] matrix2) {
    int rows1 = matrix1.length;
    int cols1 = matrix1[0].length;
    int cols2 = matrix2[0].length;
    int[][] resultMatrix = new int[rows1][cols2];
    for (int i = 0; i < rows1; i++) {
        for (int j = 0; j < cols2; j++) {
            resultMatrix[i][j] = 0;
            for (int k = 0; k < cols1; k++) {
                resultMatrix[i][j] += matrix1[i][k] * matrix2[k][j];
            }
        }
    }
    return resultMatrix;
}
private static void displayMatrix(int[][] matrix) {
    for (int[] row : matrix) {
        for (int value : row) {
            System.out.print(value + " ");
        }
        System.out.println();
    }
}
}

```

➤ Output :

For first matrix:

Enter the number of rows: 3

Enter the number of columns: 2

Enter the elements:

Enter element at position (1,1): 2

Enter element at position (1,2): 6

Enter element at position (2,1): 8

Enter element at position (2,2): 1

Enter element at position (3,1): 5

Enter element at position (3,2): 0

For second matrix:

Enter the number of rows: 2

Enter the number of columns: 3

Enter the elements:

Enter element at position (1,1): 6

Enter element at position (1,2): 5

Enter element at position (1,3): 1

Enter element at position (2,1): 2

Enter element at position (2,2): 0

Enter element at position (2,3): 9

Resultant Matrix after multiplication:

24 10 56

50 40 17

30 25 5

Assignment : 10

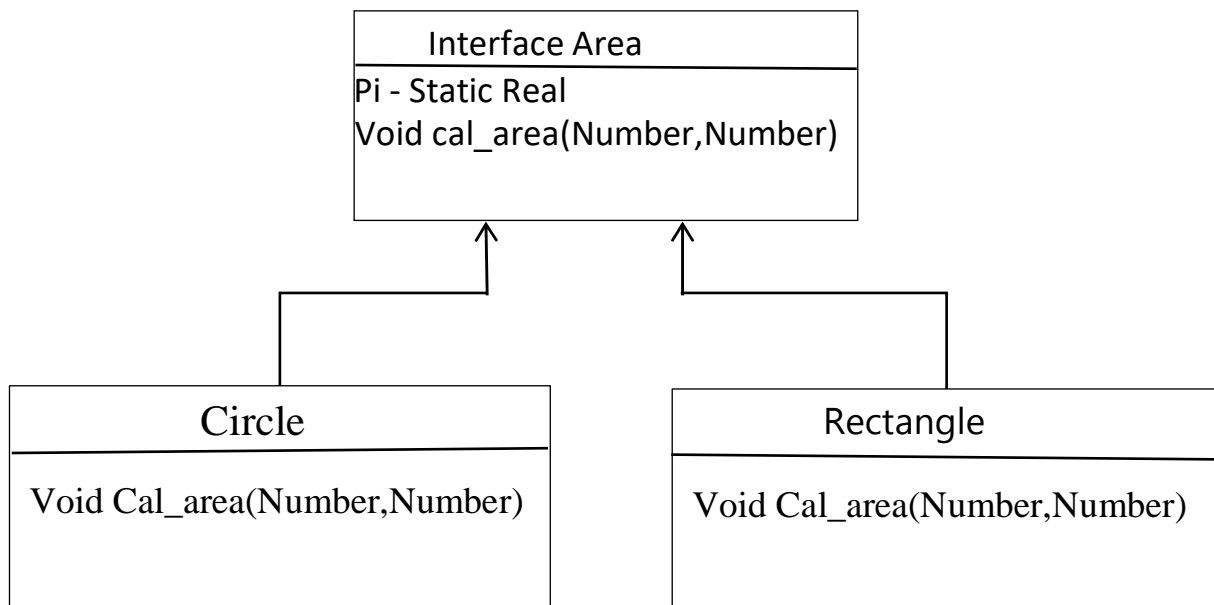
➤ Problem Statement :

- a) Create an interface named Area containing a final static variable, pi-3.14F and a method named cal area (float dim1, float dim2).

b) Create 2 classes named Circle and Rectangle to implement the Area interface (that is, define respective cal area() method).

[Hint: - Use an interface object to refer to Circle object and Rectangle object to display the respective areas]

➤ Class Diagram :



➤ Source Code :

```
interface Area{
    static float pi = 3.14F;
    void cal_area(int dim1,int dim2);
}
class circle implements Area{
    public void cal_area(int dim1,int dim2){
        System.out.println("The area of the circle is : "+2*pi*dim1);
    }
}
class Rectangle implements Area{
    public void cal_area(int dim1,int dim2){
        System.out.println("The area of the rectangle is : "+dim1*dim2);
    }
}
public class Test10 {
```

```
public static void main(String[] args) {  
    circle c1 = new circle();  
    c1.cal_area(2, 0);  
  
    Rectangle r1 = new Rectangle();  
    r1.cal_area(3, 6);  
}  
}
```

➤ Output :

The area of the circle is : 12.56

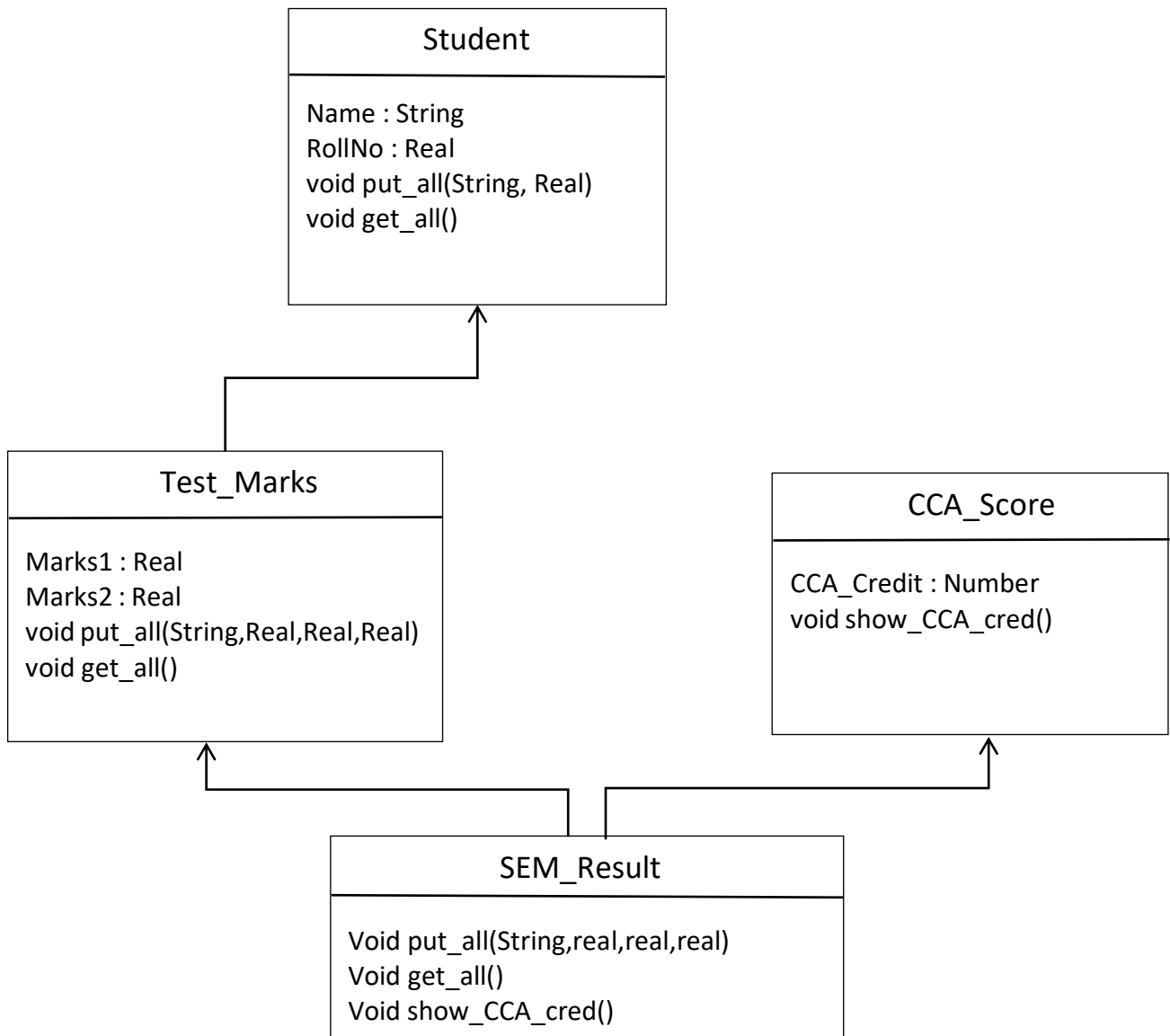
The area of the rectangle is : 18

Assignment : 11

➤ Problem Statement :

Create a class Student containing Name & Roll No as data members. Create a sub-class Test_marks containing Marks1, Marks2 as data members. Create an Interface CCA_Score containing CCA_credit data member and show cca cred) method. Create a sub-class Sem Result that inherits from Test marks and implements CCA_Score interface. Define switable get data() and put data() methods in different classes. Create objects of Sem Result class to display all details of 2 students.

➤ Class Diagram :



➤ Source Code :

```
class Student{
    String Name;
    double rollno;
    void put_all(String na,double roll){
        Name = na;
        rollno = roll;
    }
    void get_all(){
        System.out.println("The Student name is : "+Name);
        System.out.println("Roll no : "+rollno);
    }
}

class Test_Marks extends Student{
```

```

float marks1,marks2;
void put_all(String na,double roll,float m1,float m2){
    super.put_all(na, roll);
    marks1 = m1;
    marks2 = m2;
}
void get_all(){
    super.get_all();
    System.out.println("Marks1 = "+marks1);
    System.out.println("Marks2 =" +marks2);
}
}
interface CCA_Score{
    static int CCA_credit = 5;
    void show_CCA_cred();
}
class sem_result extends Test_Marks implements CCA_Score{
    void put_all(String na,double roll,float m1,float m2){
        super.put_all(na, roll, m1, m2);
    }
    void get_all(){
        super.get_all();
        show_CCA_cred();
    }
    public void show_CCA_cred(){
        System.out.println("CCA_credit is : "+CCA_credit);
    }
}
public class Test11 {
    public static void main(String[] args) {
        sem_result s1 = new sem_result();
        sem_result s2 = new sem_result();
        s1.put_all("Jagadish Sau", 55, 100,99);
        s1.get_all();
        s2.put_all("Soumyajit Nath", 54, 95, 102);
        s2.get_all();

    }
}

```

➤ Output :

The Student name is : Jagadish Sau

Roll no : 55.0

Marks1 = 100.0

Marks2 =99.0

CCA_credit is : 5

The Student name is : Soumyajit Nath

Roll no : 54.0

Marks1 = 95.0

Marks2 =102.0

CCA_credit is : 5

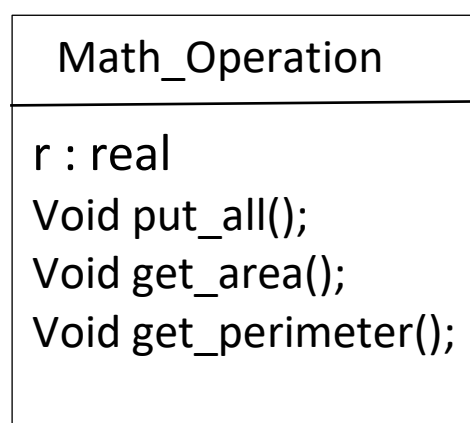
Assignment : 12

➤ Problem Statement :

Write a Java program to import Math Class defined within java.lang system package and use the value of PI defined there for finding the arca and perimeter of a circle by accepting radius as input through suitable methods defined in a user defined Class named Math_operations.

Declare object of the class Math_operations and invoke the member functions as per requirement.

➤ Class Diagram :



➤ Source Code :

```
import java.lang.Math;
```

```

import java.util.Scanner;
public class Math_operation{
    double r;
    void put_all(){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the radius of the circle : ");
        r = sc.nextDouble();
        sc.close();
    }
    void get_area(){
        System.out.println("The area of the circle of radius "+r+" is "+Math.PI*r*r);
    }
    void get_perimeter(){
        System.out.println("The perimeter of the circle of radius "+r+" is "+2*Math.PI*r);
    }
    public static void main(String[] args) {
        Math_operation r1 = new Math_operation();
        r1.put_all();
        r1.get_area();
        r1.get_perimeter();
    }
}

```

➤ Output :

Enter the radius of the circle : 5

The area of the circle of radius 5.0 is 78.53981633974483

The perimeter of the circle of radius 5.0 is 31.41592653589793

Assignment : 13

➤ Problem Statement :

Modify Assignment 11 as follows:

Design a package to contain the class Student and another package to contain the interface CCA_Score. Implement the same problem using these packages.

➤ Source Code and Output :

Subarna 13_th_assignment main ~13 33ms tree /f

pwsh 52 01:09:00

Folder PATH listing for volume New Volume

Volume serial number is 06A3-C67A

```
D:.\
├── Test13.java
├── CCA
│   └── CCA_score.java
└── Student
    └── Student.java
```

Subarna 13_th_assignment main ~13 36ms cat CCA/CCA_score.java

pwsh 52 01:09:19

package CCA;

```
public interface CCA_score{
    int CCA_credit = 5;
    void show_CCA_cred();
}
```

Subarna 13_th_assignment main ~13 7ms cat Student/Student.java

pwsh 52 01:09:27

package Student;

```
public class Student {
    String Name;
    double rollno;
    public void put_all(String na,double roll){
        Name = na;
        rollno = roll;
    }
    public void get_all(){
        System.out.println("The Student name is : "+Name);
        System.out.println("Roll no : "+rollno);
    }
}
```

Subarna 13_th_assignment main ~13 224ms cat Test13.java

```
import CCA.CCA_score;
import Student.Student;
```

```
class Test_Marks extends Student{
    float marks1,marks2;
    public void put_all(String na,double roll,float m1,float m2){
        super.put_all(na, roll);
        marks1 = m1;
        marks2 = m2;
    }
    public void get_all(){
        super.get_all();
        System.out.println("Marks1 = "+marks1);
        System.out.println("Marks2 ="+marks2);
    }
}
```

```
class sem_result extends Test_Marks implements CCA_score{
    public void put_all(String na,double roll,float m1,float m2){
        super.put_all(na, roll, m1, m2);
    }
    public void get_all(){
        super.get_all();
        show_CCA_cred();
    }
    public void show_CCA_cred(){
        System.out.println("CCA_credit is : "+CCA_credit);
    }
}
```

```
public class Test13 {
    public static void main(String[] args) {
        sem_result s1 = new sem_result();
        sem_result s2 = new sem_result();

        s1.put_all("Subarna Mandal", 42, 94,99);
        s1.get_all();

        s2.put_all("Soumyajit Nath", 54, 95, 102);
        s2.get_all();
    }
}
```

```
Subarna 13_th_assignment @main ~13 1.025s javac Test13.java
Subarna 13_th_assignment @main ~13 743ms tree /f
Folder PATH listing for volume New Volume
Volume serial number is 06A3-C67A
D:.
├── sem_result.class
├── Test13.class
├── Test13.java
├── Test_Marks.class
├── CCA
│   ├── CCA_score.class
│   └── CCA_score.java
└── Student
    ├── Student.class
    └── Student.java

Subarna 13_th_assignment @main ~13 21ms java Test13
The Student name is : Subarna Mandal
Roll no : 42.0
Marks1 = 94.0
Marks2 =99.0
CCA_credit is : 5
The Student name is : Soumyajit Nath
Roll no : 54.0
Marks1 = 95.0
Marks2 =102.0
CCA_credit is : 5
```

Assignment : 14

➤ Problem Statement :

Show with a suitable Java program that when we import any package, only the objects of public classes contained within the imported package can be created and used directly but the non-public classes contained within the package are hidden and cannot be used.

➤ Source Code & Output in Terminal:

```
Subarna 14th_Assignment /main # 212 ~4 -2 12ms tree /f
Folder PATH listing for volume New Volume
Volume serial number is 06A3-C67A
D:..
├── Main.java
└── Subarna
    ├── Usepublic.java
    └── withoutusePublic.java

Subarna 14th_Assignment /main # 212 ~4 -2 36ms cat Subarna/UsePublic.java
package Subarna;

public class Usepublic {
    public void display(){
        System.out.println("This is the display function of public class.");
    }
}

Subarna 14th_Assignment /main # 212 ~4 -2 8ms cat Subarna/withoutusePublic.java
package Subarna;

class withoutusePublic {
    public void display(){
        System.out.println("This is the display function of Non public class.");
    }
}

Subarna 14th_Assignment /main # 212 ~4 -2 46ms cat Main.java
import Subarna.*;
public class Main {
    public static void main(String[] args) {
        Usepublic up1 = new Usepublic();
        up1.display();

        withoutusePublic wup1 = new withoutusePublic();
        wup1.display();
    }
}

Subarna 14th_Assignment /main # 212 ~4 -2 49ms javac Main.java
Main.java:7: error: cannot find symbol
    withoutusePublic wup1 = new withoutusePublic();
    ^
symbol:   class withoutusePublic
location: class Main
Main.java:7: error: cannot find symbol
    withoutusePublic wup1 = new withoutusePublic();
    ^
symbol:   class withoutusePublic
location: class Main
2 errors

Subarna 14th_Assignment /main # 212 ~4 -2 694ms cat Main.java
import Subarna.*;
public class Main {
    public static void main(String[] args) {
        Usepublic up1 = new Usepublic();
        up1.display();

        //withoutusePublic wup1 = new withoutusePublic();
        //wup1.display();
    }
}

Subarna 14th_Assignment /main # 212 ~4 -2 9ms javac Main.java
Subarna 14th_Assignment /main # 212 ~4 -2 568ms tree /f
Folder PATH listing for volume New Volume
Volume serial number is 06A3-C67A
D:..
├── Main.class
├── Main.java
└── Subarna
    ├── Usepublic.class
    ├── Usepublic.java
    └── withoutusePublic.java

Subarna 14th_Assignment /main # 212 ~4 -2 41ms java Main
This is the display function of public class.

Subarna 14th_Assignment /main # 212 ~4 -2 185ms
```

➤ Discussion :

In Java, when a package is imported, only the public classes within that package can be directly used. Non-public classes within the same package are hidden and cannot be accessed from external packages. This promotes encapsulation and helps control the visibility of classes, ensuring that only intended interfaces are accessible outside the package.

Assignment : 15

➤ Problem Statement :

Write a Java program for adding a new class to an existing package. Also, implement two public classes in the Java program using packages.

➤ Source Code :

```
Subarna ➤ 15th_Assignment ➤ main # 284ms 🔥 tree /f 94% 11:25:30
Folder PATH listing for volume New Volume
Volume serial number is 06A3-C67A
D:.
├── Main.java
└── Subarna
    ├── existing.java
    └── Newadded.java

Subarna ➤ 15th_Assignment ➤ main # 81ms 🔥 cat Subarna/existing.java 94% 11:25:54
package Subarna;

public class existing {
    public void display(){
        System.out.println("I am an existing class!!!");
    }
}

Subarna ➤ 15th_Assignment ➤ main # 53ms 🔥 cat Subarna/Newadded.java 94% 11:26:06
package Subarna;

public class Newadded{
    public void display(){
        System.out.println("I am a new class!!!");
    }
}

Subarna ➤ 15th_Assignment ➤ main # 10ms 🔥 cat Main.java 94% 11:26:24
import Subarna.*;

public class Main {
    public static void main(String[] args) {
        existing e1 = new existing();
        e1.display();

        Newadded n1 = new Newadded();
        n1.display();
    }
}

Subarna ➤ 15th_Assignment ➤ main # 13ms 🔥 javac Main.java 94% 11:26:31
Subarna ➤ 15th_Assignment ➤ main # 3.565s 🔥 tree /f 94% 11:26:45
Folder PATH listing for volume New Volume
Volume serial number is 06A3-C67A
D:.
├── Main.class
├── Main.java
└── Subarna
    ├── existing.class
    ├── existing.java
    ├── Newadded.class
    └── Newadded.java

Subarna ➤ 15th_Assignment ➤ main # 50ms 🔥 java Main
I am an existing class!!!
I am a new class!!!

Subarna ➤ 15th_Assignment ➤ main # 305ms 🔥
```


➤ Discussion :

The provided Java program illustrates the creation of classes within a package (Subarna). The newly added class, “Newadded”, along with existing classes “existing”, showcase the use of packages for organizing and encapsulating related code. The compilation and execution process involves ensuring the correct package structure. The program output demonstrates the distinct functionality of each class, highlighting the benefits of modularization and code organization in Java.

Assignment : 16

➤ Problem Statement :

Write a java program to show the following built-in-exceptions

- i) ArithmeticException
- ii) ArrayIndexOutOfBoundsException
- iii) NullPointerException
- iv) NumberFormatException

➤ Source Code :

```
public class Exception {  
    public static void main(String[] args) {  
        // ArithmeticException  
        try {  
            int result = 5 / 0; // Division by zero  
        } catch (ArithmeticException e) {  
            System.out.println("ArithmeticException: " + e.getMessage());  
        }  
        // ArrayIndexOutOfBoundsException  
        try {  
            int[] arr = {1, 2, 3};  
            int element = arr[5]; // Accessing an index beyond the array size  
        } catch (ArrayIndexOutOfBoundsException e) {
```

```

        System.out.println("ArrayIndexOutOfBoundsException: " +
e.getMessage());
    }
    // NullPointerException
    try {
        String str = null;
        int length = str.length(); // Attempting to invoke a method on a null object
    }

    catch (NullPointerException e) {
        System.out.println("NullPointerException: " + e.getMessage());
    }
    // NumberFormatException
    try {
        String strNumber = "abc";
        int parsedNumber = Integer.parseInt(strNumber); // Parsing a non-
numeric string
    } catch (NumberFormatException e) {
        System.out.println("NumberFormatException: " + e.getMessage());
    }
}
}

```

```

Subarna 16th_Assignment ? ?main ?/? ?16 ~4 -2 ? 24ms ? dir

Directory: D:\programming language\OOP - JAVA 2023\5th sem 2023 college\16th_Assignment

Mode                LastWriteTime         Length Name
----                -
-a---             23-11-2023    12:44           1248 Exception.java

Subarna 16th_Assignment ? ?main ?/? ?16 ~4 -2 ? 15ms ? javac Exception.java
Subarna 16th_Assignment ? ?main ?/? ?16 ~4 -2 ? 707ms ? dir

Directory: D:\programming language\OOP - JAVA 2023\5th sem 2023 college\16th_Assignment

Mode                LastWriteTime         Length Name
----                -
-a---             23-11-2023    12:46           1599 Exception.class
-a---             23-11-2023    12:44           1248 Exception.java

Subarna 16th_Assignment ? ?main ?/? ?16 ~4 -2 ? 10ms ? java Exception
ArithmeticException: / by zero
ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 3
NullPointerException: Cannot invoke "String.length()" because "<local1>" is null
NumberFormatException: For input string: "abc"

```

➤ Discussion :

The Java program demonstrates intentional triggering and handling of four common exceptions:

ArithmeticException:

- Triggered by attempting division by zero.
- Caught and a relevant message is printed.

ArrayIndexOutOfBoundsException:

- Triggered by accessing an index beyond the array's size.
- Caught and a descriptive message is printed.

NullPointerException:

- Triggered by invoking a method on a null object.
- Caught with a message indicating the nature of the exception.

NumberFormatException:

- Triggered by attempting to parse a non-numeric string into an integer.
- Caught and a message is printed.

The program uses `try-catch` blocks to handle exceptions, promoting controlled error handling and enhancing the overall robustness of the code.

Assignment : 17

➤ Problem Statement :

Create a java Applet to display your name at location (20,50) of a window.

➤ Source Code :

```
import java.applet.Applet;
import java.awt.*;
public class applet1 extends Applet {
    public void paint (Graphics g){
        int x = 20;
        int y = 50;
```

```
g.setColor(Color.BLACK);
g.setFont(new Font("Arial",Font.BOLD,16));
g.drawString("SUBARNA MANDAL", x, y);
}
}
/*
* <applet code="applet1.class" width="1000" height="1000"></applet>
*/
```

➤ Output :



➤ Discussion :

The provided Java applet code named 'applet1', extends the 'Applet' class to draw the text "SUBARNA MANDAL" at coordinates (20, 50) in a specified font and color. The paint method is overridden to define the rendering behavior.

Assignment : 18

➤ Problem Statement :

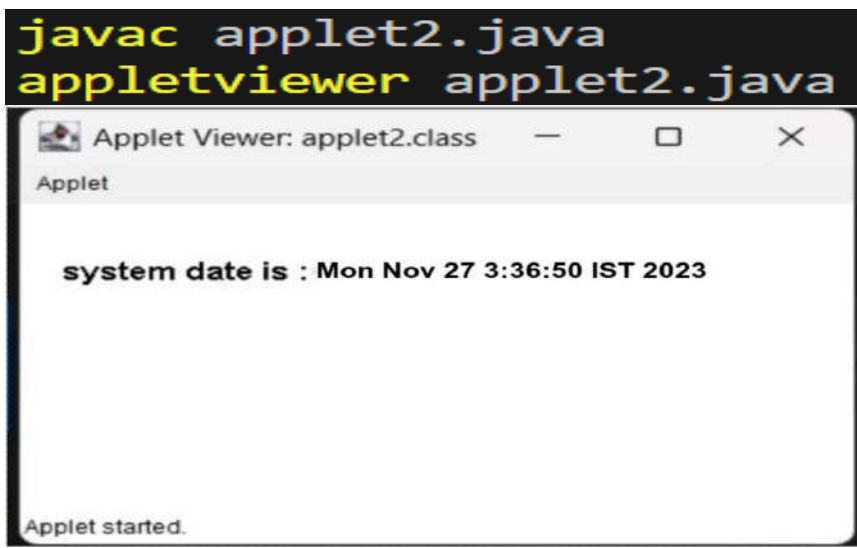
Create and run applet to display the system date on your window.

➤ Source Code :

```
import java.applet.Applet;
import java.awt.*;
```

```
import java.util.Date;
public class applet2 extends Applet {
    public void paint(Graphics g){
        Date currentdate = new Date();
        String date = currentdate.toString();
        g.setColor(getBackground().BLACK);
        g.setFont(new Font("Arial",Font.BOLD,16));
        g.drawString("System time : "+date,20,50);
    }
}
/*
 * <applet code="applet1.class" width="1000" height="1000"></applet>
 */
```

➤ Output :



➤ Discussion :

The provided Java applet code, named 'applet2', extends the Applet class to display the current system time on the applet window. The 'paint' method is overridden to set the rendering behavior. The applet retrieves the current date using the Date class and renders it on the window using specified font and color.

Assignment : 19

➤ Problem Statement :

Create a JAVA Applet that accepts 2 numbers from user and display their sum.

➤ Source Code :

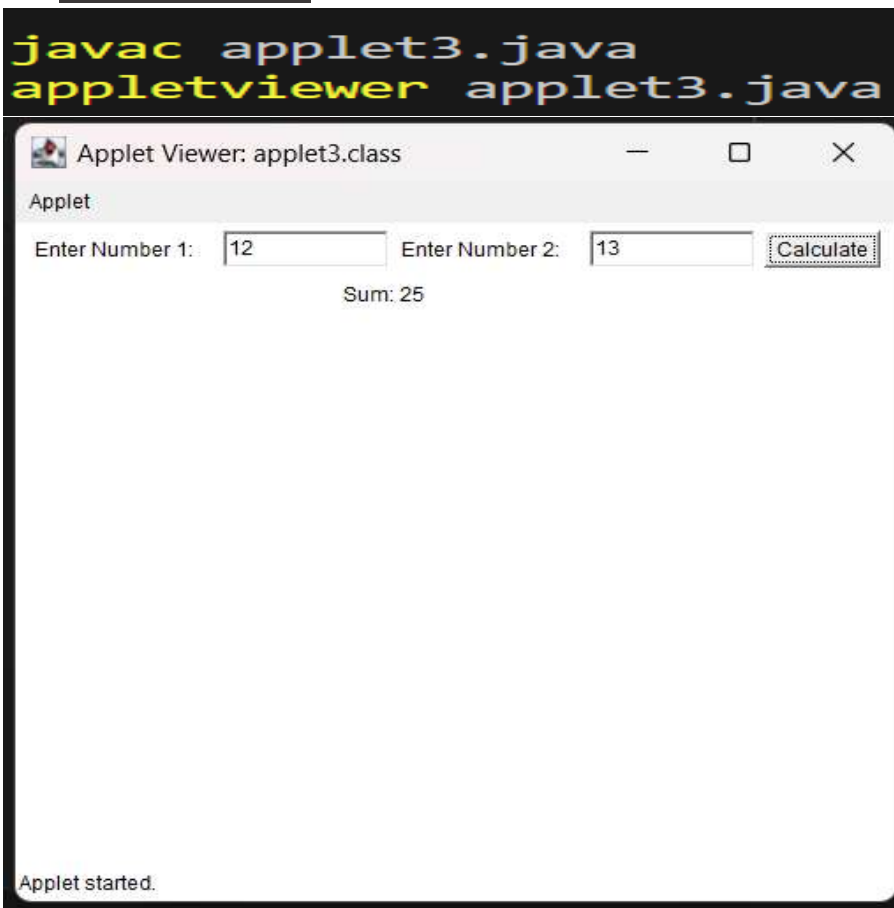
```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.ActionListener;
public class applet3 extends Applet implements ActionListener {
    TextField num1Field, num2Field;
    Button calculateButton;
    Label resultLabel;
    public void init() {
        // Create text fields for input
        num1Field = new TextField(10);
        num2Field = new TextField(10);
        // Create a button to trigger calculation
        calculateButton = new Button("Calculate");
        calculateButton.addActionListener(this);
        // Create a label to display the result
        resultLabel = new Label("Sum will appear here");
        // Add components to the applet
        add(new Label("Enter Number 1:"));
        add(num1Field);
        add(new Label("Enter Number 2:"));
        add(num2Field);
        add(calculateButton);
        add(resultLabel);
    }
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == calculateButton) {
            // Get the numbers entered by the user
            int num1 = Integer.parseInt(num1Field.getText());
```

```

    int num2 = Integer.parseInt(num2Field.getText());
    // Calculate the sum
    int sum = num1 + num2;
    // Display the result
    resultLabel.setText("Sum: " + sum);
}
}
}
/*
 * <applet code = "applet3.class" width = "400" height = "400" ></applet>
 */

```

➤ Output :



➤ Discussion :

The Java applet code, `applet3`, creates a basic interactive applet for calculating the sum of two numbers. It extends the `Applet` class, initializes user interface components in the `init` method, and implements the `ActionListener` interface for event handling. Users can input numbers, click the "Calculate" button, and view the sum.

Assignment : 20

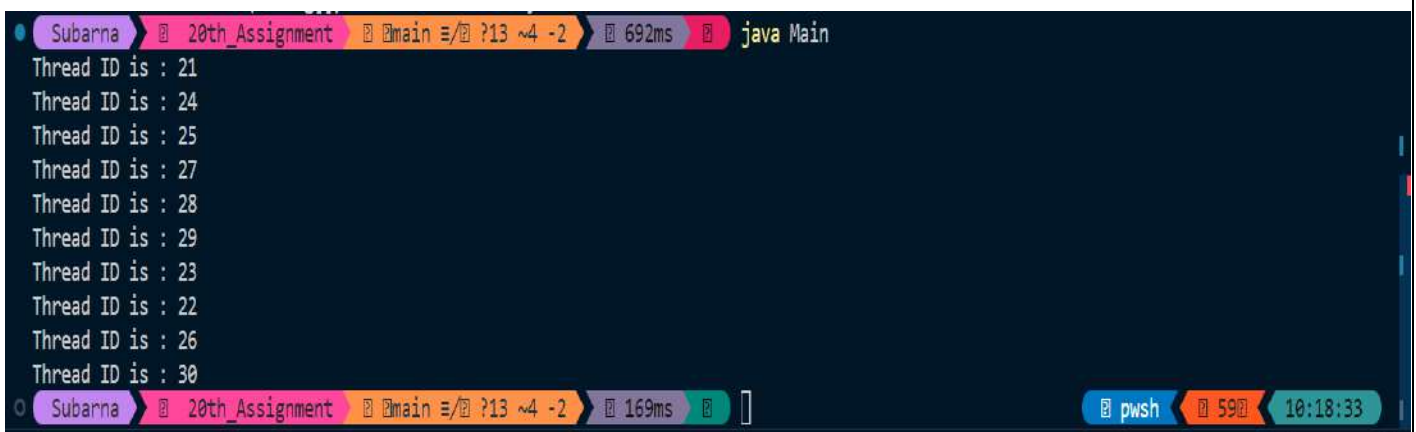
➤ Problem Statement :

Write a Java program to create 20 threads and print their Ids. [Hint: Inherit Thread Class]

➤ Source Code :

```
class MyThread extends Thread {  
    public void run() {  
        System.out.println("Thread ID is : " + Thread.currentThread().getId());  
    }  
}  
class Main {  
    public static void main(String[] args) {  
        for(int i=1; i<=10; i++) {  
            Thread thread = new MyThread();  
            thread.start();  
        }  
    }  
}
```

➤ Output:



```
Subarna 20th_Assignment main 13 ~4 -2 692ms java Main  
Thread ID is : 21  
Thread ID is : 24  
Thread ID is : 25  
Thread ID is : 27  
Thread ID is : 28  
Thread ID is : 29  
Thread ID is : 23  
Thread ID is : 22  
Thread ID is : 26  
Thread ID is : 30  
Subarna 20th_Assignment main 13 ~4 -2 169ms pwsh 59 10:18:33
```

➤ Discussion :

The Java program creates 10 threads, each printing its unique ID. It defines a class MyThread that extends Thread and overrides the run method. The main class creates 10 instances of MyThread and starts each thread. The output displays the thread IDs, showcasing concurrent execution.

Assignment : 21

➤ Problem Statement :

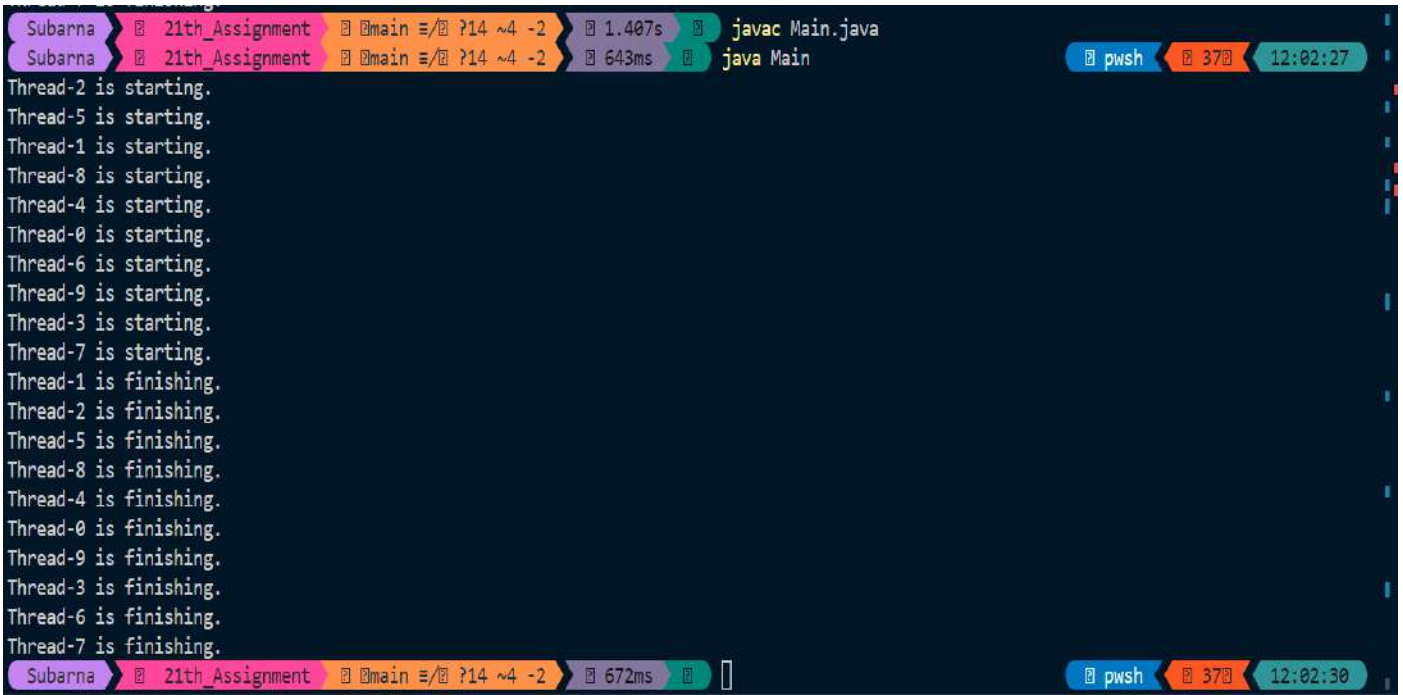
Write a Java program to create 10 threads and print their names implementing Runnable interface. Use sleep() method to control working of threads. Use appropriate exception handling techniques.

➤ Source Code :

```
class MyRunnable implements Runnable{
    public void run(){
        try{
            System.out.println(Thread.currentThread().getName() + " is starting.");
            Thread.sleep(500);
            System.out.println(Thread.currentThread().getName() + " is finishing.");
        }catch (InterruptedException e) {
            System.err.println("Thread execution interrupted: " + e.getMessage());
        }
    }
}

public class Main{
    public static void main(String[] args) {
        for (int i=0; i<10; i++) {
            MyRunnable my = new MyRunnable();
            Thread thread = new Thread(my);
            thread.start();
        }
    }
}
```

➤ Output:



```
Subarna 21th_Assignment main ?14 ~4 -2 1.407s javac Main.java
Subarna 21th_Assignment main ?14 ~4 -2 643ms java Main
pwsh 37 12:02:27
Thread-2 is starting.
Thread-5 is starting.
Thread-1 is starting.
Thread-8 is starting.
Thread-4 is starting.
Thread-0 is starting.
Thread-6 is starting.
Thread-9 is starting.
Thread-3 is starting.
Thread-7 is starting.
Thread-1 is finishing.
Thread-2 is finishing.
Thread-5 is finishing.
Thread-8 is finishing.
Thread-4 is finishing.
Thread-0 is finishing.
Thread-9 is finishing.
Thread-3 is finishing.
Thread-6 is finishing.
Thread-7 is finishing.
Subarna 21th_Assignment main ?14 ~4 -2 672ms
pwsh 37 12:02:30
```

➤ Discussion :

The Java program utilizes the Runnable interface to create 10 threads. Each thread, represented by the MyRunnable class, prints its name and simulates work using the sleep() method. Exception handling is implemented to address potential interruptions during the sleep. The program demonstrates how to achieve concurrency and control thread behavior in a concise manner using the Runnable interface in Java.