1. Create arrayList, add the integer elements in arrayList using asList().Remove the duplicate values and return a arrayList containing unique values. Implement the logic inside removeDuplicates() method. Test the functionalities using the main () method of the Tester class.

**Sample Input and Output**---------10, 20, 10, 15,40,15,40 --- 10,20,15,40

import java.util.\*;  
  
public class Main {  
  
 // Function to remove duplicates from an ArrayList  
 public static <T> ArrayList<T> removeDuplicates(ArrayList<T> list)  
 {  
  
 // Create a new ArrayList  
 ArrayList<T> newList = new ArrayList<T>();  
  
 // Traverse through the first list  
 for (T element : list) {  
  
 // If this element is not present in newList  
 // then add it  
 if (!newList.contains(element)) {  
  
 newList.add(element);  
 }  
 }  
  
 // return the new list  
 return newList;  
 }  
  
 // Driver code  
 public static void main(String args[])  
 {  
  
 // Get the ArrayList with duplicate values  
 ArrayList<Integer>  
 list = new ArrayList<>(  
 Arrays  
 .*asList*(10, 20, 10, 15, 40, 15, 40));  
  
 // Print the Arraylist  
 System.*out*.println("ArrayList with duplicates: "  
 + list);  
  
 // Remove duplicates  
 ArrayList<Integer>  
 newList = *removeDuplicates*(list);  
  
 // Print the ArrayList with duplicates removed  
 System.*out*.println("ArrayList with duplicates removed: "  
 + newList);  
 }  
}

1. Given two lists, concatenate the second list in reverse order to the end of the first list and return the concatenated list. Implement the logic inside concatenateLists() method.

listOne = Hello 102 200.8 25

listTwo = 150 40.8 welcome A

**output:** Hello 102 200.8 25 A welcome 40.8 150

import java.util.\*;  
  
public class Main {  
 public static List concatenateList(List listOne,List listTwo) {  
 List new\_list= new ArrayList<>();  
 for(int i =0;i<listOne.size();i++)  
 new\_list.add(listOne.get(i));  
 for(int i=listTwo.size()-1;i>=0;i--){  
 new\_list.add(listTwo.get(i));  
 }  
 return new\_list;  
  
  
 }  
 public static void main(String[] args) {  
 List listOne= new ArrayList<>(Arrays.*asList*("Hello",102,200.8,25));  
 List listTwo= new ArrayList<>(Arrays.*asList*(150,"Welcome","A"));  
 List listthree = *concatenateList*(listOne,listTwo);  
  
 for(int i=0;i<listthree.size();i++)  
 System.*out*.print(listthree.get(i)+" ");  
  
 }  
}

1. Find and return the average salary, number of salaries greater than the average salary and number of salaries lesser than the average salary from the salary array passed to the findDetails() method. Method should return a double array containing average salary in index position 0, number of salaries greater than the average salary in index position 1 and number of salaries lesser than the average salary in index position 2. Implement the logic inside findDetails() method. Test the functionalities using the main method of the Tester class. **sample Input:** {23500.0 , 25080.0 , 28760.0 , 22340.0 , 19890.0}

**output:** Average salary: 23914.0

Number of salaries greater than the average salary: 2.0

Number of salaries lesser than the average salary: 3.0

import java.util.\*;  
public class Main {  
 public static double[] lab(){  
 int s1 = 23500;  
 int s2 = 25080;  
 int s3 = 28760;  
 int s4 = 22340;  
 int s5 = 19890;  
 int [] salaries = {s1,s2,s3,s4,s5};  
 int sum = 0;  
 for(int i=0; i< salaries.length;i++){  
 sum = sum +salaries[i];  
 }  
 double average = sum/ (salaries.length+1);  
 //first value  
  
 int lower\_than\_avg = 0;  
 int higher\_than\_avg = 0;  
 for(int i=0; i< salaries.length;i++){  
 if (salaries[i]<average){  
 lower\_than\_avg = lower\_than\_avg + 1;  
 } else if (salaries[i]>average) {  
 higher\_than\_avg = higher\_than\_avg + 1;  
 }  
 }  
 double [] return\_arr = {average,lower\_than\_avg,higher\_than\_avg};  
 return return\_arr;  
 }  
  
  
 public static void main(String[] args) {  
 double [] return\_values = *lab*();  
  
 System.*out*.println("Average Salary : "+return\_values[0]);  
 System.*out*.println("Number of salaries greater than the average salary : "+return\_values[1]);  
 System.*out*.println("Number of salaries smaller than the average salary: "+return\_values[2]);  
  
 }  
}

1. Write a Java Program to-
   1. Check that given number is Armstrong or not

public class Main {  
  
 public static void main(String[] args) {  
  
 int number = 371, originalNumber, remainder, result = 0;  
  
 originalNumber = number;  
  
 while (originalNumber != 0)  
 {  
 remainder = originalNumber % 10;  
 result += Math.*pow*(remainder, 3);  
 originalNumber /= 10;  
 }  
  
 if(result == number)  
 System.*out*.println(number + " is an Armstrong number.");  
 else  
 System.*out*.println(number + " is not an Armstrong number.");  
 }  
}

* 1. Check that given number is palindrome or not

public class Main1 {  
 public static void main(String[] args) {  
  
 int num = 343, reversedNum = 0, remainder;  
  
 // store the number to originalNum  
 int originalNum = num;  
  
 // get the reverse of originalNum  
 // store it in variable  
 while (num != 0) {  
 remainder = num % 10;  
 reversedNum = reversedNum \* 10 + remainder;  
 num /= 10;  
 }  
  
 // check if reversedNum and originalNum are equal  
 if (originalNum == reversedNum) {  
 System.*out*.println(originalNum + " is Palindrome.");  
 }  
 else {  
 System.*out*.println(originalNum + " is not Palindrome.");  
 }  
 }  
}

* 1. Check that given number is odd or even

import java.util.\*;  
public class Main2 {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter a number: ");  
 int a = sc.nextInt();  
 if(a%2==0) {  
 System.*out*.println("Given number is even");  
 }  
 else {  
 System.*out*.println("Given number is odd");  
 }  
  
 }  
}

* 1. Print reverse of a number

public class Main3 {  
 public static void main(String[] args) {  
  
 int num = 9564, reversed = 0;  
  
 System.*out*.println("Given Number: " + num);  
  
 // run loop until num becomes 0  
 while(num != 0) {  
  
 // get last digit from num  
 int digit = num % 10;  
 reversed = reversed \* 10 + digit;  
  
 // remove the last digit from num  
 num /= 10;  
 }  
  
 System.*out*.println("Reverse of Number: " + reversed);  
 }  
}

1. An educational institution provides stipends for post-graduate students every year. For calculating the stipend, the institution has fixed a base amount of $100 which is provided to all the students. The students who perform exceptionally well during the academics get an extra amount based on their performance. You need to help the institution in developing an application for calculating the stipend by implementing the class using info given below. **Note:** STIPEND is a final variable.

**calculateTotalStipend():-**Calculate and return the total stipend amount based on the aggregate marks of the student using the below table. Implement the getter and setter methods appropriately.

i) Aggregate marks>=85 and <90 then bonus stipend amt is $10

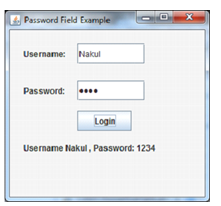
ii) Aggregate marks>=90 and <95 then bonus stipend amt is $15

iii) Aggregate marks>=95 and <100 then bonus stipend amt is $20

**sample input:** student ID:1212 aggregate marks :93 **output:** the final stipend of 1212 is $115.0

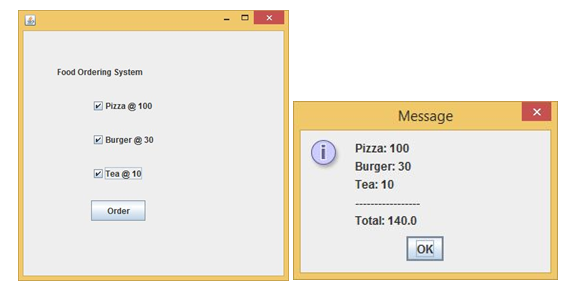
public class Main {  
 private int STIPEND = 100 ;  
 private int studentId;  
 private int aggregateMarks;  
 double calculateTotalStipend;  
  
 public Main (){  
 this.STIPEND = STIPEND;  
 this.studentId = studentId;  
 this.aggregateMarks = aggregateMarks;  
 }  
  
  
 public int getstudentId() {  
 return studentId;  
 }  
 public void setStudentId(int studentId) {  
 this.studentId = studentId ;  
 }  
 public int getSTIPEND(){  
 return STIPEND;  
 }  
 public void setSTIPEND(int STIPEND) {  
 this.STIPEND = STIPEND;  
 }  
  
 public double calculateTotalStipend() {  
 if(getaggregateMarks() >= 85 && getaggregateMarks() < 90) {  
 return STIPEND +10;  
 }  
 else if(getaggregateMarks() >= 90 && getaggregateMarks() < 95) {  
 return STIPEND +15;  
 }  
 else if(getaggregateMarks() >= 95 && getaggregateMarks() < 100) {  
 return STIPEND +20;  
 }  
 else return STIPEND ;  
 }  
  
 public int getaggregateMarks() {  
 return aggregateMarks;  
 }  
  
 public void setaggregateMarks(int aggregateMarks) {  
 this.aggregateMarks = aggregateMarks;  
 }  
}  
  
class Tester {  
  
 public static void main(String[] args) {  
 Main student1 = new Main() ;  
 student1.setStudentId(1212);  
 student1.setaggregateMarks(93);  
  
 double totalStipend = student1.calculateTotalStipend();  
 System.*out*.println("The final stipend of " + student1.getstudentId()+" is $" + totalStipend);  
  
  
 }  
}

1. Create Java GUI using Swing. When click on login button, it shows the credential in label as shown in the figure.



package Main;  
import javax.swing.\*;  
import java.awt.event.\*;  
public class Main {  
 public static void main(String[] args) {  
 JFrame f=new JFrame("User login");  
  
 final JButton b=new JButton("Login");  
 b.setBounds(160,100,100, 30);  
 final JTextField f1 = new JTextField();  
 final JPasswordField f2 = new JPasswordField(16);  
 f1.setBounds(160, 10, 200, 20);  
 f2.setBounds(160,60,200,20);  
 JLabel l1 = new JLabel("Username");  
 JLabel l2 = new JLabel("Password");  
 final JLabel l3 = new JLabel(" ");  
 l1.setBounds(10, 10, 100, 14);  
 l2.setBounds(10, 60, 100, 14);  
 l3.setBounds(20, 150, 400, 14);  
 f.add(b);  
 f.add(f1);//adding button in JFrame  
 f.add(l1);  
 f.add(l2);  
 f.add(f2);  
 f.add(l3);  
 b.addActionListener(new ActionListener(){  
 public void actionPerformed(ActionEvent e){  
 l3.setText("Username: " + f1.getText() + " Password: " + f2.getText());  
 }  
 });  
 f.setSize(400,300);  
 f.setLayout(null);  
 f.setVisible(true);  
 }  
}

1. Create Java GUI using Swing. When click on order button, it shows order details with bill amount in message box as shown in the figure.



import javax.swing.\*;  
import java.awt.event.\*;  
public class ques7 extends JFrame implements ActionListener{  
 JLabel l;  
 JCheckBox cb1,cb2,cb3;  
 JButton b;  
 ques7(){  
 l=new JLabel("Food Ordering System");  
 l.setBounds(50,50,300,20);  
 cb1=new JCheckBox("Pizza @ 100");  
 cb1.setBounds(100,100,150,20);  
 cb2=new JCheckBox("Burger @ 30");  
 cb2.setBounds(100,150,150,20);  
 cb3=new JCheckBox("Tea @ 10");  
 cb3.setBounds(100,200,150,20);  
 b=new JButton("Order");  
 b.setBounds(100,250,80,30);  
 b.addActionListener(this);  
 add(l);add(cb1);add(cb2);add(cb3);add(b);  
 setSize(400,400);  
 setLayout(null);  
 setVisible(true);  
 setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);  
 }  
 public void actionPerformed(ActionEvent e){  
 float amount=0;  
 String msg="";  
 if(cb1.isSelected()){  
 amount+=100;  
 msg="Pizza: 100\n";  
 }  
 if(cb2.isSelected()){  
 amount+=30;  
 msg+="Burger: 30\n";  
 }  
 if(cb3.isSelected()){  
 amount+=10;  
 msg+="Tea: 10\n";  
 }  
 msg+="-----------------\n";  
 JOptionPane.*showMessageDialog*(this,msg+"Total: "+amount);  
 }  
 public static void main(String[] args) {  
 new ques7();  
 }  
}

1. Write a Java Program to count the number of words in a string using HashMap.

**Input**: Enter String: " This this is is done by Saket Saket ";

**Output:** {Saket=2, by=1, this=1, This=1, is=2, done=1}

import java.util.\*;  
public class Main {  
 public static void main(String[] args)  
 {  
  
 // Declaring the String  
 String str = "This this is is done by Saket Saket";  
 // Declaring a HashMap of <String, Integer>  
 Map<String, Integer> hashMap = new HashMap<>();  
  
 // Splitting the words of string  
 // and storing them in the array.  
 String[] words = str.split(" ");  
  
 for (String word : words) {  
  
 // Asking whether the HashMap contains the  
 // key or not. Will return null if not.  
 Integer integer = hashMap.get(word);  
  
 if (integer == null)  
 // Storing the word as key and its  
 // occurrence as value in the HashMap.  
 hashMap.put(word, 1);  
  
 else {  
 // Incrementing the value if the word  
 // is already present in the HashMap.  
 hashMap.put(word, integer + 1);  
 }  
 }  
 System.*out*.println(hashMap);  
 }  
}

1. a) Write a program to take the input array element and convert all the elements into next prime numbers and display the changed array.

import java.util.\*;

public class NextPrime

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int a[]= {2,4,6,8,10};

int i,c,j,c1,z;

c=0;

for(i=0;i<5;i++)

{

for(j=1;j<=a[i];j++)

{

if(a[i]%j==0)

{

c++;

}

}

if(c==2)

{

System.out.println(a[i]);

}

else

{

z=1;

for(int l=1;l<=z;l++)

{

int n1=a[i]+z;

c1=0;

while(c1<=2)

{

for(int k=1;k<=n1;k++)

{

if(n1%k==0)

{

c1++;

}

}

if(c1==2)

{

System.out.println(n1+" ");

break;

}

else

{

z++;

}

}

}

}

c=0;

}

}

}

b) Write a java program that takes two positive integers as command-line arguments and prints true if either evenly divides the other.

import java.util.\*;  
public class Main2 {  
 Scanner sc = new Scanner(System.*in*);  
 int a,b;  
 public void divide(){  
 System.*out*.println("Enter two numbers ");  
 a = sc.nextInt();  
 b = sc.nextInt();  
 if(a%b==0 || b%a==0){  
 System.*out*.println("TRUE");  
 }  
 else{  
 System.*out*.println("FALSE");  
 }  
 }  
 public static void main(String[] args) {  
 Main2 ob = new Main2();  
 ob.divide();  
 }  
}

10. Write a Java Program to iterate ArrayList using for-loop, iterator, and advance for-loop. Insert 3 Array List. **Input** : 20 30 40

**Output:**

**Iterator Loop:** 20 30 40 **Advanced For Loop:** 20 30 40 **for Loop:** 20 30 40

import java.util.ArrayList;  
import java.util.Iterator;  
  
public class Main {  
 public static void main(String[] args) {  
 ArrayList<Integer> a1 = new ArrayList<Integer>();  
 a1.add(20);  
 a1.add(30);  
 a1.add(40);  
 System.*out*.println("Using For Loop:");  
 for(int i=0;i< a1.size();i++){  
 System.*out*.print(a1.get(i)+" ");  
 }  
  
 System.*out*.println("\nUsing Iterator:");  
 Iterator it = a1.iterator();  
 while (it.hasNext())  
 System.*out*.print(it.next() + " ");  
  
 System.*out*.println("\nUsing Advanced For Loop:");  
 for (Integer i : a1){  
 System.*out*.print(i+" ");  
 }  
 }  
  
}

1. a) Write a program that takes three double values X0, V0, and t from the user and prints the valuehttp://thetechpoint.in/img/question/assignment26.PNG, where g is the constant 9.78033. This value is the displacement in meters after t seconds when an object is thrown straight up from initial position x0 at velocity v0 meters per second.

**Sample Output:** Enter the value of X0, V0, and t: 0 2 2

Note: The displacement in meters after t seconds when an object is thrown straight up from initial position X0 at velocity V0 meters per second is: 23.56066

import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 double x=sc.nextDouble();  
 double v=sc.nextDouble();  
 double t=sc.nextDouble();  
 double k= (double) (((9.78033)\*t\*t)/2);  
 System.*out*.println("Enter the value of x");  
 x = sc.nextInt();  
 System.*out*.println("Enter the value of v");  
 v = sc.nextInt();  
 System.*out*.println("Enter the value of t");  
 t = sc.nextInt();  
 System.*out*.println(x + v\*t + k);  
 }  
}

b) Write a program that takes two int values m and d from the command line and prints true if day d of month m is between 3/20 and 6/20, false otherwise.

import java.util.Scanner;  
  
public class Main2 {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 int m = sc.nextInt();  
 int d = sc.nextInt();  
 boolean res = (((m==3)&&(d>20&&d<=31))||((m==4)&&(d>=1&&d<=30))||((m==5)&&(d>=1&&d<=31))||((m==6)&&(d>=1&&d<20)));  
 System.*out*.println("Result=" + res);  
 }  
}

1. a) Write a program to check if the two strings entered by user are anagrams or not. Two words are said to be anagrams if the letters of one word can be rearranged to form the other word. For example, jaxa and ajax are anagrams of each other.
2. import java.util.\*;  
   public class Main {  
    public static void main(String[] args) {  
    Scanner sc =new Scanner(System.*in*);  
    System.*out*.println("Enter 1st string: ");  
    String str1 = sc.nextLine();  
    System.*out*.println("Enter 2st string: ");  
    String str2 = sc.nextLine();  
    str1 = str1.toLowerCase();  
    str2 = str2.toLowerCase();  
     
    // check if length is same  
    if(str1.length() == str2.length()) {  
     
    // convert strings to char array  
    char[] charArray1 = str1.toCharArray();  
    char[] charArray2 = str2.toCharArray();  
     
    // sort the char array  
    Arrays.*sort*(charArray1);  
    Arrays.*sort*(charArray2);  
     
    // if sorted char arrays are same  
    // then the string is anagram  
    boolean result = Arrays.*equals*(charArray1, charArray2);  
     
    if(result) {  
    System.*out*.println(str1 + " and " + str2 + " are anagram.");  
    }  
    else {  
    System.*out*.println(str1 + " and " + str2 + " are not anagram.");  
    }  
    }  
    else {  
    System.*out*.println(str1 + " and " + str2 + " are not anagram.");  
    }  
    }  
   }

b) Check whether the string is palindrome without using string methods.

import java.util.Scanner;  
public class Main2 {  
  
 public static void main(String[] args) {  
  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Please enter string to check palindrome: ");  
 String strInput = sc.nextLine();  
 char[] chString = strInput.toCharArray();  
 // storing reverse string  
 String strReverse = "";  
 // reading char by char  
 for(int a = chString.length - 1; a >= 0; a--)  
 {  
 strReverse = strReverse + chString[a];  
 }  
 System.*out*.println("Given string: " + strInput);  
 System.*out*.println("Reverse String: " + strReverse);  
 if(strInput.equals(strReverse))  
 {  
 System.*out*.println("string is palindrome.");  
 }  
 else  
 {  
 System.*out*.println("string is not palindrome.");  
 }  
 }  
}

13.Write a Java program to calculate the Factorial of an integer number using both iterative and recursive solutions.

import java.util.\*;  
public class Main {  
 private static int factRecursive(int number) {  
 // base condition  
 if (number == 1)  
 return 1;  
  
 // calculate the factorial of all number  
 return number \* *factRecursive*(number - 1);  
 }  
  
 private static int factIterative(int number) {  
  
 int factorial = 1;  
 for (int iNumber = 1; iNumber <= number; iNumber++) {  
 factorial = factorial \* iNumber;  
 }  
 return factorial;  
 }  
  
 public static void main(String[] args) {  
 try (Scanner scanner = new Scanner(System.*in*)) {  
  
 /\* Calculate factorial for input number \*/  
 System.*out*.printf(" Enter input number : ");  
 int number = scanner.nextInt();  
  
 int factNumber = *factRecursive*(number);  
 System.*out*.printf("factorial(%d) - Recursive method: %d\n",number,factNumber);  
  
 factNumber = *factIterative*(number);  
 System.*out*.printf("factorial(%d) - Iterative method: %d\n",number,factNumber);  
 }  
 }  
}

1. Write a program that reads from the user four integers representing the numerators and denominators of two fractions calculates the results of the two fractions and displays the values of the fractions sum, subtraction, multiplication and division. Display output up to two decimal places.

**Sample Input**: Enter the numerator and denominator of the first fraction: 6 4

Enter the numerator and denominator of the second fraction: 8 5

**Output**: The sum is: 3.10

The subtraction is: -0.10

The multiplication is: 2.40

The division is: 0.93

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 int n1= sc.nextInt();  
 int d1= sc.nextInt();  
 int n2= sc.nextInt();  
 int d2= sc.nextInt();  
 float a=n1\*d2;  
 float b=n2\*d1;  
 float c=d1\*d2;  
 float d=n1\*n2;  
 float e=d1\*d2;  
 float sum=(a+b)/c;  
 float diff=(a-b)/c;  
 float multi=d/e;  
 float div=a/b;  
 System.*out*.println("The sum is: "+sum);  
 System.*out*.println("The diff is: "+diff);  
 System.*out*.println("The multiplication is: "+multi);  
 System.*out*.println("The division is: "+div);  
 }  
}

1. Write a program to implement following inheritance. Accept data for 5 persons and display the name of employee having salary greater than 5000. Class Name: Person Member variables: Name, age Class Name: Employee Member variables: Designation, salary.

class Person{  
 String name;  
 int age;  
 Person(int age, String name) {  
 this.name = name;  
 this.age = age;  
 }  
}  
class Employee extends Person{  
 String designation;  
 int salary;  
 Employee(String designation, String name, int age, int salary) {  
 super(age, name);  
 this.designation = designation;  
 this.salary = salary;  
 if (salary>5000){  
 System.*out*.println(name);  
 }  
 }  
}  
public class Main{  
 public static void main (String [] args){  
 Employee emp = new Employee("Developer","Mohit",19,10000);  
 Employee emp1 = new Employee("Manager","Aryan",25,3000);  
 Employee emp2 = new Employee("Accountant","Akashdeep",26,4000);  
 Employee emp3 = new Employee("Developer","Ritesh",22,6000);  
 Employee emp4 = new Employee("Data Scientist","Sai",23,2500);  
 }  
}

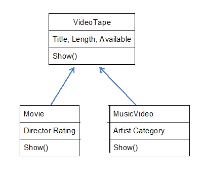
1. Implementing “Multiple Inheritance”. Create a two interfaces Account containing methods set () and display () And interface Person containing methods store () and disp(). Derive a class Customer from Person and Account. Accept the name, account number, balance and display all the information related to account along with the interest.

// Interface Account  
interface Account {  
 void set();  
 void display();  
}  
  
// Interface Person  
interface Person {  
 void store();  
 void disp();  
}  
  
// Class Customer  
public class Customer implements Person, Account {  
 private String name;  
 private int accountNumber;  
 private double balance;  
  
 // Constructor to initialize name, account number, and balance  
 public Customer(String name, int accountNumber, double balance) {  
 this.name = name;  
 this.accountNumber = accountNumber;  
 this.balance = balance;  
 }  
  
 // Method to set the account information  
 public void set() {  
 // Code to set the account information  
 }  
  
  
 public void display() {  
 //  
 }  
  
 // Method to display the account information  
 public void disp() {  
 System.*out*.println("Name: " + name);  
 System.*out*.println("Account Number: " + accountNumber);  
 System.*out*.println("Balance: " + balance);  
 // Code to calculate and display the interest  
 double p, r, t, si; // principal amount, rate, time and simple interest respectively.  
 p = balance; r = 12; t = 1;  
 si = (p\*r\*t)/100;  
 System.*out*.println("Simple interest is: "+si);  
 }  
  
 // Method to store the person's information  
 public void store() {  
 // Code to store the person's information  
 }  
  
  
 public static void main(String[] args) {  
 Customer c = new Customer("Mohit Bang", 1256789345, 100000.0);  
 c.set();  
 c.display();  
 c.store();  
 c.disp();  
 }  
  
}

1. Write a program to read 10 strings from console and then print the sorted strings on console (Use String Class).2) combine two string 3) reverse first string and display it.

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 String str;  
 String arr1[]=new String[10];  
 for(int i=0;i<10;i++){  
 System.*out*.println("Enter String: "+i);  
 str=sc.next();  
 arr1[i]=str;  
 char arr[] = str.toCharArray();  
 char temp;  
 for(int a=0;a<arr.length;a++){  
 for (int b=a+1;b<arr.length;b++){  
 if (arr[b] < arr[a]) {  
 temp = arr[a];  
 arr[a] = arr[b];  
 arr[b] = temp;  
 }  
 }  
 }  
 System.*out*.println(arr);  
 }  
 System.*out*.println(arr1[0]+" "+arr1[1]);  
 String nstr="";  
 char ch;  
 for (int i=0; i<arr1[0].length(); i++)  
 {  
 ch= arr1[0].charAt(i);  
 nstr= ch+nstr;  
 }  
 System.*out*.println("Reversed First String: "+ nstr);  
  
 }  
}

1. Write a program, to implement the following hierarchy. Displays information of each class the rectangle represents the classes. The classes Movie and MusicVideo inherit all the members of the class VideoTape.



class VideoTape {  
 private String title;  
 private int length;  
  
 public VideoTape(String title, int length) {  
 this.title = title;  
 this.length = length;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public int getLength() {  
 return length;  
 }  
  
 @Override  
 public String toString() {  
 return "VideoTape: " + title + " (" + length + " minutes)";  
 }  
}  
  
class Movie extends VideoTape {  
 private String rating;  
  
 public Movie(String title, int length, String rating) {  
 super(title, length);  
 this.rating = rating;  
 }  
  
 public String getRating() {  
 return rating;  
 }  
  
 @Override  
 public String toString() {  
 return "Movie: " + getTitle() + " (" + getLength() + " minutes, rated " + rating + ")";  
 }  
}  
  
class MusicVideo extends VideoTape {  
 private String artist;  
  
 public MusicVideo(String title, int length, String artist) {  
 super(title, length);  
 this.artist = artist;  
 }  
  
 public String getArtist() {  
 return artist;  
 }  
  
 @Override  
 public String toString() {  
 return "MusicVideo: " + getTitle() + " (" + getLength() + " minutes, by " + artist + ")";  
 }  
  
  
 public static void main(String[] args) {  
  
 // Test the classes  
 VideoTape tape1 = new VideoTape("The Secret Life of Pets", 90);  
 System.*out*.println(tape1.toString()); // prints "VideoTape: The Secret Life of Pets (90 minutes)"  
  
 Movie movie1 = new Movie("Jurassic Park", 127, "4.5\*");  
 System.*out*.println(movie1.toString()); // prints "Movie: Jurassic Park (127 minutes, rated 4.5\*)"  
  
 MusicVideo musicVideo1 = new MusicVideo("Roar", 3, "Katy Perry");  
 System.*out*.println(musicVideo1.toString()); // prints "MusicVideo: Roar (3 minutes, by Katy Perry)"  
  
 }  
}

1. Create two arrayLists ,add the String elements in arrayList using add().
   1. Add one arraylist into the other from index 1 using appropriate method.
   2. Print the two added list.
   3. Print the index of "Are".
   4. Remove the 4th element from arraylist1
   5. Replace 4 element of arraylist2 as "U"

Test the functionalities using the main() method of the Tester class.

**Sample Input: str1 - ("**Hello", "How", "Are", "You") **str2 - ("**Hi" , "How", "Are" ,You") **Sample Output:** [Hello, Hi, How, Are, You, How, Are, You]

import java.util.\*;  
public class Main {  
 public static void main(String[] args) {  
 ArrayList<String> str1=new ArrayList<>();  
 str1.add("Hello");  
 str1.add("How");  
 str1.add("Are");  
 str1.add("You");  
  
 ArrayList<String> str2=new ArrayList<>();  
 str2.add("Hi");  
 str2.add("How");  
 str2.add("Are");  
 str2.add("You");  
  
 ArrayList<String> str3=new ArrayList<>();  
 str3.addAll(str1);  
 str3.addAll(1,str2);  
 System.*out*.println(str3);  
  
 System.*out*.println("The index of Are:"+str3.indexOf("Are"));  
  
 str1.remove(3);  
 System.*out*.println(str1);  
  
 str2.set(3,"U");  
 System.*out*.println(str2);  
 }  
}

1. Create a new class Order in the Java project with the instance variables and methods mentioned below. **orderId: int , orderedFoods: String, totalPrice: double, status: String , calculateTotalPrice(int unitPrice): double**. Calculate the total price by applying a service charge of 5% on the food item ordered and store it in the instance variable totalPrice. Return the calculated total price. Create an object of the Order class, initialize the instance variables using parameterized constructor, invoke the calculateTotalPrice() method and display the values of the instance variables in the main() method of the Tester class. Use static block to print status "Ordered".

**Sample Output: Order Details:**

Order Id: 101

Ordered Food: Burger

Order status: Ordered

Total Price: 35

class Order1{  
 private int orderId;  
 private String orderedFoods;  
 private double totalPrice;  
 private String status;  
  
 public Order1(int orderId, String orderedFoods) {  
 this.orderId = orderId;  
 this.orderedFoods = orderedFoods;  
 }  
  
 public int getOrderId() {  
 return orderId;  
 }  
  
 public String getOrderedFoods() {  
 return orderedFoods;  
 }  
  
 double calculateTotalPrice(int unitPrice){  
 totalPrice = (unitPrice+(0.05\*unitPrice));  
 return totalPrice;  
 }  
 static {  
 System.*out*.println("Order Status: Ordered");  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 System.*out*.println("Order Details");  
 Order1 order1=new Order1(101,"Burger");  
 System.*out*.println("Order ID:" + order1.getOrderId());  
 System.*out*.println("Ordered Food: "+order1.getOrderedFoods());  
 System.*out*.println("Total Price "+order1.calculateTotalPrice(33));  
  
 }  
}

1. Create GUI using Swing. As shown in the figure. Write a program to print the text entered in the text field-1 into text field-2 after button click. Display entered text in label below button also.

Button

values print here

TextField-1

TextField-2

import javax.swing.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class Main {  
 public static void main(String[] args) {  
 JFrame f=new JFrame("Swing");  
 final JButton b=new JButton("Click");  
 b.setBounds(50,100,100, 30);  
 final JTextField f1 = new JTextField();  
 final JTextField f2 = new JTextField();  
 f1.setBounds(50, 10, 200, 20);  
 f2.setBounds(50,60,200,20);  
 final JLabel l3 = new JLabel(" ");  
 l3.setBounds(20, 150, 400, 14);  
 f.add(b);  
 f.add(f1);//adding button in JFrame  
 f.add(f2);  
 f.add(l3);  
 b.addActionListener(new ActionListener(){  
 public void actionPerformed(ActionEvent e){  
 l3.setText(f1.getText() + " "+ f2.getText());  
 }  
 });  
 f.setSize(400,300);  
 f.setLayout(null);  
 f.setVisible(true);  
 }  
}

1. Write a program to handle the custom exception when the number entered by user through keyboard is odd. Use throw and throws keywords. Exception name is OddNumberException. If the number is odd print message "Number is Odd" else print "Number is Even".

class OddNumberException extends Exception{  
 @Override  
 public String getMessage(){  
 return "Number is Odd";  
 }  
}  
  
public class Main {  
 public static void main(String[] args) throws OddNumberException{  
 Scanner sc=new Scanner(System.*in*);  
 OddNumberException oddNumberException = new OddNumberException();  
 System.*out*.println("Enter a number");  
 int a= sc.nextInt();  
 if(a%2!=0){  
 try {  
 throw new OddNumberException();  
 }  
 catch (Exception e){  
 System.*out*.println(oddNumberException.getMessage());  
 }  
 }  
 else {  
 System.*out*.println("Number is Even");  
 }  
 }  
}

1. Calculate and return the sum of all the even numbers present in the numbers array passed to the method calculateSumOfEvenNumbers. Implement the logic inside calculateSumOfEvenNumbers() method. Test the functionalities using the main() method of the Tester class.

**Sample Input :** {68,79,86,99,23,2,41,100} **Sample Output**: 256

**Sample Input :** {1,2,3,4,5,6,7,8,9,10} **Sample Output:** 30

public class Main {  
 static int calculateSumOfEvenNumbers(int array[]){  
 int sum=0;  
 for(int i=0;i<array.length;i++){  
 if(array[i]%2==0){  
 sum=sum+array[i];  
 }  
 }  
 return sum;  
  
 }  
 public static void main(String[] args) {  
 int arr[]={1,2,3,4,5,6,7,8,9,10};  
 System.*out*.println(*calculateSumOfEvenNumbers*(arr));  
  
 }  
}

1. Two classes - Camera and DigitalCamera are provided to you. DigitalCamera extends Camera class. Both classes have their parameterized constructors.

Camera Class: private String brand; private double cost;

DigitalCamera Class: private int memory;

Create a instance of child class and display the output as shown below.

**Sample Output:** Nikon, 100$, 16GB

class Camera{  
 private String brand;  
 private double cost;  
  
 public Camera(String brand, double cost) {  
 this.brand = brand;  
 this.cost = cost;  
 }  
  
}  
  
class DigitalCamera extends Camera{  
 private int memory;  
  
 public DigitalCamera(String brand, double cost, int memory) {  
 super(brand, cost);  
 this.memory = memory;  
 System.*out*.println("The brand is: "+brand);  
 System.*out*.println("The cost is: "+cost+"$");  
 System.*out*.println("The memory is: "+memory+"GB");  
 }  
}  
  
  
public class Main {  
 public static void main(String[] args) {  
 DigitalCamera digitalCamera = new DigitalCamera("Nikon",100,16);  
 }  
}

1. Create Calculator GUI using Swing. Add buttons for numbers 0-9 , operators +, -, x, /, =, AC. Display output of addition and subtraction operations in textbox.

Calculator

12 + 13 =25

7

9

6

-

+

0

CC

=

x

1

/

2

4

3

8

5

import java.awt.event.\*;  
import javax.swing.\*;  
class calculator extends JFrame implements ActionListener {  
 // create a frame  
 static JFrame *f*;  
  
 // create a textfield  
 static JTextField *l*,*l2*;  
  
 // store operator and operands  
 String s0, s1, s2;  
  
 // default constructor  
 calculator()  
 {  
 s0 = s1 = s2 = "";  
 }  
  
 // main function  
 public static void main(String args[])  
 {  
 // create a frame  
 *f* = new JFrame("calculator");  
 *l2* = new JTextField("Calculator");  
  
  
  
 // create a object of class  
 calculator c = new calculator();  
  
 // create a textfield  
 *l* = new JTextField(16);  
  
 // set the textfield to non editable  
 *l*.setEditable(false);  
  
 // create number buttons and some operators  
 JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq, beq1;  
  
 // create number buttons  
 b0 = new JButton("0");  
 b1 = new JButton("1");  
 b2 = new JButton("2");  
 b3 = new JButton("3");  
 b4 = new JButton("4");  
 b5 = new JButton("5");  
 b6 = new JButton("6");  
 b7 = new JButton("7");  
 b8 = new JButton("8");  
 b9 = new JButton("9");  
  
 // equals button  
 beq1 = new JButton("=");  
  
 // create operator buttons  
 ba = new JButton("+");  
 bs = new JButton("-");  
 bd = new JButton("/");  
 bm = new JButton("\*");  
 beq = new JButton("C");  
  
 // create . button  
 be = new JButton(".");  
 // create a panel  
 JPanel p = new JPanel();  
  
 // add action listeners  
 bm.addActionListener(c);  
 bd.addActionListener(c);  
 bs.addActionListener(c);  
 ba.addActionListener(c);  
 b9.addActionListener(c);  
 b8.addActionListener(c);  
 b7.addActionListener(c);  
 b6.addActionListener(c);  
 b5.addActionListener(c);  
 b4.addActionListener(c);  
 b3.addActionListener(c);  
 b2.addActionListener(c);  
 b1.addActionListener(c);  
 b0.addActionListener(c);  
 be.addActionListener(c);  
 beq.addActionListener(c);  
 beq1.addActionListener(c);  
  
  
 // add elements to panel  
 p.add(*l2*);  
 p.add(*l*);  
 p.add(b1);  
 p.add(b2);  
 p.add(b3);  
 p.add(b4);  
 p.add(b5);  
 p.add(b6);  
 p.add(b7);  
 p.add(b8);  
 p.add(b9);  
 p.add(b0);  
 p.add(ba);  
 p.add(bs);  
 p.add(bd);  
 p.add(bm);  
 p.add(beq);  
 p.add(beq1);  
  
 // set Background of panel  
  
  
 // add panel to frame  
 *f*.add(p);  
  
 *f*.setSize(200, 250);  
 *f*.setVisible(true);  
 *f*.setLayout(null);  
 }  
 public void actionPerformed(ActionEvent e)  
 {  
 String s = e.getActionCommand();  
  
 // if the value is a number  
 if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.') {  
 // if operand is present then add to second no  
 if (!s1.equals(""))  
 s2 = s2 + s;  
 else  
 s0 = s0 + s;  
  
 // set the value of text  
 *l*.setText(s0 + s1 + s2);  
 }  
 else if (s.charAt(0) == 'C') {  
 // clear the one letter  
 s0 = s1 = s2 = "";  
  
 // set the value of text  
 *l*.setText(s0 + s1 + s2);  
 }  
 else if (s.charAt(0) == '=') {  
 double te;  
 // store the value in 1st  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2));  
 else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2));  
 else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2));  
 else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2));  
  
 // set the value of text  
 *l*.setText(s0 + s1 + s2 + "=" + te);  
  
 // convert it to string  
 s0 = Double.*toString*(te);  
  
 s1 = s2 = "";  
 }  
 else {  
 // if there was no operand  
 if (s1.equals("") || s2.equals(""))  
 s1 = s;  
 // else evaluate  
 else {  
 double te;  
  
 // store the value in 1st  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2));  
 else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2));  
 else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2));  
 else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2));  
  
 // convert it to string  
 s0 = Double.*toString*(te);  
 // place the operator  
 s1 = s;  
 // make the operand bla  
 s2 = "";  
 }  
 // set the value of text  
 *l*.setText(s0 + s1 + s2);  
 }  
 }  
}

1. 1) Write a Java program to find the maximum and minimum value of an array

public class Main {  
 public static void main(String[] args) {  
 int array[]= {1, 2, 3, 4, 5};  
 int max = 0;  
  
 for(int i=0; i<array.length; i++ ) {  
 if(array[i]>max) {  
 max = array[i];  
 }  
 }  
 int min = array[0];  
  
 for(int i=0; i<array.length; i++ ) {  
 if(array[i]<min) {  
 min = array[i];  
 }  
 }  
 System.*out*.println("Maximum value: "+max);  
 System.*out*.println("Minimum value: "+min);  
 }  
}

2) Write a Java program to find the second largest element in an array.

import java.util.Arrays;  
  
public class Main3 {  
 public static void main(String[] args) {  
 int[] arr = {5, 2, 7, 1, 8, 3};  
  
 // Sort the array in non-descending order  
 Arrays.*sort*(arr);  
  
 // Return the second last element of the sorted array  
 System.*out*.println("Second largest element: " + arr[arr.length - 2]);  
 }  
}

3) Take 10 integer inputs from user and store them in an array. Now, copy all the elements in another array but in reverse order.

import java.util.Scanner;  
  
public class Main2 {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 int arr[] =new int[10];  
 System.*out*.println("Enter 10 numbers");  
 for(int i=0;i<10;i++){  
 arr[i]=sc.nextInt();  
 }  
  
 int arr1[]=new int[10];  
 for (int j=0;j<10;j++){  
 arr1[j]=arr[10-j-1];  
 }  
  
 for(int k=0;k<10;k++){  
 System.*out*.print(arr1[k]);  
 }  
  
 }  
}

1. 1) Find Subarray with given sum

**Input:** arr[] = {1, 4, 20, 3, 10, 5}, sum = 33

**Output:** Sum found between indexes 2 and 4

**Explanation:** Sum of elements between indices 2 and 4 is 20 + 3 + 10 = 33

**Input:** arr[] = {1, 4}, sum = 0

**Output:** No subarray found

**Explanation:** There is no subarray with 0 sum

public class SubarraySum {  
 /\* Returns true if the there is a  
subarray of arr[] with a sum equal to  
 'sum' otherwise returns false.  
Also, prints the result \*/  
 void subArraySum(int arr[], int n, int sum)  
 {  
 // Pick a starting point  
 for (int i = 0; i < n; i++) {  
 int currentSum = arr[i];  
  
 if (currentSum == sum) {  
 System.*out*.println("Sum found at indexe "  
 + i);  
 return;  
 }  
 else {  
 // Try all subarrays starting with 'i'  
 for (int j = i + 1; j < n; j++) {  
 currentSum += arr[j];  
  
 if (currentSum == sum) {  
 System.*out*.println(  
 "Sum found between indexes " + i  
 + " and " + j);  
 return;  
 }  
 }  
 }  
 }  
 System.*out*.println("No subarray found");  
 return;  
 }  
  
 public static void main(String[] args)  
 {  
 SubarraySum arraysum = new SubarraySum();  
 int arr[] = {1, 4, 20, 3, 10, 5 };  
 int n = arr.length;  
 int sum = 33;  
 arraysum.subArraySum(arr, n, sum);  
 }  
}

2) Count number of occurrences (or frequency) in a sorted array

**Input:** arr[] = {1, 1, 2, 2, 2, 2, 3,}, x = 1

**Output:** 2

**Input:** arr[] = {1, 1, 2, 2, 2, 2, 3,}, x = 4

**Output:** -1 // 4 doesn't occur in arr[]

// Java program to count occurrences  
// of an element  
  
class Main  
{  
 // Returns number of times x occurs in arr[0..n-1]  
 static int countOccurrences(int arr[], int n, int x)  
 {  
 int res = 0;  
 for (int i=0; i<n; i++)  
 if (x == arr[i])  
 res++;  
 return res;  
 }  
  
 public static void main(String args[])  
 {  
 int arr[] = {1, 1, 2, 2, 2, 2, 3 };  
 int n = arr.length;  
 int x = 1;  
 System.*out*.println(*countOccurrences*(arr, n, x));  
 }  
}

1. Create a GUI that has two buttons on it. When clicked, each button creates a new window. The first button creates a window that has a single button on it. Pressing that button will change the window’s background color back and forth between red and green. The second button creates a window that has two buttons. Pressing the first button of these two buttons will change the window’s background color to black. Pressing the second button will change the background color to white. Make sure that your windows will just dispose of themselves when they are closed.

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
import javax.swing.border.\*;  
public class Main  
{  
 public static void main(String[] args) {  
 MultiWindow window1 = new MultiWindow();  
 window1.setVisible(true);  
 }  
}  
  
class MultiWindow extends JFrame implements ActionListener {  
  
 public static final int *WIDTH* = 400;  
 public static final int *HEIGHT* = 300;  
 public static final int *FLOAT\_WIDTH* = 500;  
 public static final int *FLOAT\_HEIGHT* = 150;  
  
 private JButton floatOneButton;  
  
 private JButton floatTwoButton;  
  
 */\*\*  
  
 \* Creates a new instance of MultiWindow  
  
 \*/* public MultiWindow() {  
  
 super();  
  
// This window is sized so that it will not be hidden by the floating window  
  
// that is created  
  
 setSize(*WIDTH*, *HEIGHT*);  
  
 setTitle("Multiple windows Demo");  
  
 setLayout(new BorderLayout());  
  
 floatOneButton = new JButton("Create Window Type 1");  
  
 floatTwoButton = new JButton("Create Window Type 2");  
  
 floatOneButton.addActionListener(this);  
  
 floatTwoButton.addActionListener(this);  
  
 getContentPane().add(floatOneButton, BorderLayout.*EAST*);  
  
 getContentPane().add(floatTwoButton, BorderLayout.*WEST*);  
  
//setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);  
  
// WindowDestroyer listener = new WindowDestroyer();  
//  
// addWindowListener(listener);  
  
 }  
  
 public void actionPerformed(ActionEvent e) {  
  
 if (e.getSource()==floatOneButton) {  
  
 RedGreenWindow x = new RedGreenWindow();  
  
 x.setVisible(true);  
  
 } else if (e.getSource()==floatTwoButton) {  
  
 BlackWhiteWindow x = new BlackWhiteWindow();  
  
 x.setVisible(true);  
  
 } else  
  
 System.*out*.println("Error in interface.");  
  
 }  
  
 private class FloatingWindowHandler extends WindowAdapter {  
  
 private JFrame myWindow;  
  
 public FloatingWindowHandler(JFrame aWindow){  
  
 myWindow = aWindow;  
  
 }  
  
 public void windowClosing(WindowEvent e) {  
  
 myWindow.dispose();  
  
 }  
  
 }  
  
// The first kind of window  
  
 private class RedGreenWindow extends JFrame  
  
 implements ActionListener {  
  
 private JButton colorButton;  
  
 private boolean isRed = true;  
  
 public RedGreenWindow(){  
  
 super();  
  
 setSize(*FLOAT\_WIDTH*, *FLOAT\_HEIGHT*);  
  
 setTitle("A Red/Green Window");  
  
 setLayout(new BorderLayout());  
  
 setForeground(Color.*RED*);  
  
 colorButton = new JButton("Change");  
  
 colorButton.addActionListener(this);  
  
 add(colorButton, BorderLayout.*WEST*);  
  
 FloatingWindowHandler listener = new FloatingWindowHandler(this);  
  
 addWindowListener(listener);  
  
 }  
  
 public void actionPerformed(ActionEvent e) {  
  
 if (e.getActionCommand().equals("Change")) {  
  
//System.out.println("Change button pressed");  
  
//System.out.println("Change applied to " + this);  
  
 if(isRed){  
  
 this.getContentPane().setBackground(Color.*GREEN*);  
  
 isRed = false;  
  
 } else {  
  
 this.getContentPane().setBackground(Color.*RED*);  
  
 isRed = true;  
  
 }  
  
 this.setForeground(Color.*RED*);  
  
 } else  
  
 System.*out*.println("Error in red/green events");  
  
 }  
  
 }  
  
// The second kind of window  
  
 private class BlackWhiteWindow extends JFrame  
  
 implements ActionListener {  
  
 private JButton blackButton;  
  
 private JButton whiteButton;  
  
 public BlackWhiteWindow(){  
  
 super();  
  
 setSize(*FLOAT\_WIDTH*, *FLOAT\_HEIGHT*);  
  
 setTitle("A Black/White Window");  
  
 setLayout(new BorderLayout());  
  
 setForeground(Color.*GRAY*);  
  
 blackButton = new JButton("Black");  
  
 blackButton.addActionListener(this);  
  
 add(blackButton, BorderLayout.*NORTH*);  
  
 whiteButton = new JButton("White");  
  
 whiteButton.addActionListener(this);  
  
 add(whiteButton, BorderLayout.*SOUTH*);  
  
 FloatingWindowHandler listener = new FloatingWindowHandler(this);  
  
 addWindowListener(listener);  
  
 }  
  
 public void actionPerformed(ActionEvent e) {  
  
 if (e.getActionCommand().equals("White")) {  
  
//System.out.println("White button pressed");  
  
//System.out.println("Change applied to " + this);  
  
 this.getContentPane().setBackground(Color.*WHITE*);  
  
 } else if (e.getActionCommand().equals("Black")) {  
  
//System.out.println("White button pressed");  
  
//System.out.println("Change applied to " + this);  
  
 this.getContentPane().setBackground(Color.*BLACK*);  
 } else  
 System.*out*.println("Error in Black/White events");  
 }  
  
 }  
  
}

1. Create an abstract class 'Bank' with an abstract method 'getBalance'. $100, $150 and $200 are deposited in banks A, B and C respectively. 'BankA', 'BankB' and 'BankC' are subclasses of class 'Bank', each having a method named 'getBalance'. Call this method by creating an object of each of the three classes.

abstract class Bank{  
 abstract void getBalance();  
}  
  
class BankA extends Bank{  
 public void getBalance(){  
 System.*out*.println("The balance in Bank A is 100$");  
 }  
}  
  
class BankB extends Bank{  
 public void getBalance(){  
 System.*out*.println("The balance in Bank B is 150$");  
 }  
}  
  
class BankC extends Bank{  
 public void getBalance(){  
 System.*out*.println("The balance in Bank C is 200$");  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 BankA bankA = new BankA();  
 bankA.getBalance();  
  
 BankB bankB=new BankB();  
 bankB.getBalance();  
  
 BankC bankC = new BankC();  
 bankC.getBalance();  
 }  
}

1. All the banks operating in India are controlled by RBI. RBI has set a well defined guideline (e.g. minimum interest rate, minimum balance allowed, maximum withdrawal limit etc) which all banks must follow. For example, suppose RBI has set minimum interest rate applicable to a saving bank account to be 4% annually; however, banks are free to use 4% interest rate or to set any rates above it.

Write a JAVA program to implement bank functionality in the above scenario and demonstrate the dynamic polymorphism concept. **Note:** Create few classes namely Customer, Account, RBI (Base Class) and few derived classes (SBI, ICICI, PNB etc). Assume and implement required member variables and functions in each class.

**Hint:** Class Customer {

//Personal Details ... // Few functions ...

}

Class Account {

// Account Detail ... // Few functions ...

}

Class RBI { Customer c; //hasA relationship Account a;

//hasA relationship ..

Public double GetInterestRate() { }

Public double GetWithdrawalLimit() { }

}

Class SBI: public RBI {

//Use RBI functionality or define own functionality.

}

Class ICICI: public RBI { //Use RBI functionality or define own functionality. }

**package** Lab;

**import** java.util.\*;

**class** Customer

{

**private** String name,age,address;

}

**class** Account

{

String acc,branchType;

}

**class** RBI {

**int** withLimit, n;

**double** A, P, r;

**float** t;

**long** avg, tot, mb;

**int** setWithdrawalLimit() {

Scanner sc = **new** Scanner(System.***in***);

String actype = sc.nextLine();

String brtype = sc.nextLine();

**if** (actype.equals("SAVINGS") && brtype.equals("METRO")) {

withLimit = 5000;

} **else** {

withLimit = 4000;

}

**return** withLimit;

}

**static** **class** SBI **extends** RBI { // Child class1

**double** setInterestRate() {

A = P \* (1 + r / n) + n \* t;

**return** A;

}

**long** setMAB() {

mb = (avg + tot) / 30;

**return** mb;

}

}

**static** **class** ICICI **extends** RBI { // Child class2

**double** setInterestRate() {

A = (P \* r) / n + t;

**return** A;

}

**int** setWithdrawalLimit() {

withLimit = 7000;

**return** withLimit;

}

}

**public** **static** **class** bank30 {

**public** **static** **void** main(String args[]) {

SBI sbi = **new** SBI();

ICICI icici = **new** ICICI();

sbi.P = 10.433;

sbi.r = 7.42;

sbi.n = 5;

sbi.t = 7;

**double** sbiInt = sbi.setInterestRate();

System.***out***.println("SBI interest rate is:" + sbiInt);

**int** iciciWL = icici.setWithdrawalLimit();

System.***out***.println("ICICI withdraw limit is:" + iciciWL);

}

}

}

**LAB ASSIGNMENT 01**

**Problem Statement:**

A VIT Melange committee is conducting auditions to admit interested candidates. You need to implement a Participant class for the dance club based on the class diagram and description given below.

|  |
| --- |
| **C-Participant** |
| * counter: int * registrationId: String * name: String * contactNumber: long * branch: String |
| * **C**Participant(name:String,contactNumber:long,branch:String) * getRegistrationId(): String * getCounter(): int * setCounter(counter:int):void * getName():String * setName(name:String):void * getContactNumber():long * setContactNumber(contactNumber:long):void * getBranch():String * setBranch(branch:String):void |

**Method Description**

**Partcipant(String name, long contactNumber, String branch)**

* Initialize the name, contactNumber and branch instance variables appropriately with the values passed to the constructor.
* Generate the registrationId using the static variable counter.
* The value of registrationId should start from 'D1001' and the numerical part should be incremented by 1 for the subsequent values.
* Initialize the counter in static block.

Implement the appropriate getter and setter methods.

Test the functionalities using the provided Tester class. Create two or more Participant objects and validate that the values of the member variables are proper.

**Sample Input and Output**

**For constructor**

**Input**

For first Participant object

|  |  |
| --- | --- |
| **Parameters** | **Values** |
| name | Rohit |
| contactNumber | 1234567889 |
| branch | Computer |

For second Participant object

|  |  |
| --- | --- |
| **Parameters** | **Values** |
| name | Sayli |
| contactNumber | 1988612300 |
| branch | Mechanical |

**Expected Output**

|  |
| --- |
| Hi Rohit! Your registration id is D1001  Hi Sayli! Your registration id is D1002 |

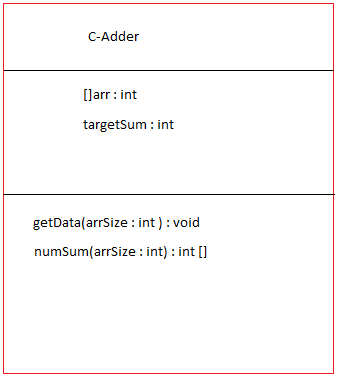
Add your Code Here

class Participant{  
 static int *counter*;  
 String RegistrationID;  
 String name;  
 long ContactNo;  
 String Branch;  
  
 static{  
 *counter* = 1001;  
 }  
  
 Participant(String name, long ContactNo, String Branch){  
 this.name = name;  
 this.ContactNo = ContactNo;  
 this.Branch = Branch;  
 this.RegistrationID = "D"+*counter*;  
 Participant.*counter*++;  
 }  
  
 public String getRegID(){  
 return RegistrationID;  
 }  
  
 public void setRegID(String RegistrationID){  
 this.RegistrationID = RegistrationID;  
 }  
  
 public static int getCounter(){  
 return *counter*;  
 }  
  
 public static void setCounter(int counter){  
 Participant.*counter* = counter;  
 }  
  
 public String getName(){  
 return name;  
 }  
  
 public void setName(String name){  
 this.name = name;  
 }  
  
 public long getCont(){  
 return ContactNo;  
 }  
  
 public void setCont(long ContactNO){  
 this.ContactNo = ContactNo;  
 }  
  
 public String getBranch(){  
 return Branch;  
 }  
  
 public void setBranch(String Branch){  
 this.Branch = Branch;  
 }  
  
  
 public static void main(String [] args){  
 Participant student1 = new Participant("Mohit!", 1234567889, "Mechanical");  
 Participant student2 = new Participant("Harry!", 1988612300, "Computer");  
  
 System.*out*.println("Hi "+student1.getName()+" your registration ID is "+student1.getRegID());  
 System.*out*.println("Hi "+student2.getName()+" your registration ID is "+student2.getRegID());  
 }  
  
}

**LAB ASSIGNMENT 02**

**Problem Statement:**

There is a class Adder which has two data members of type 1D int array and int variable. It has two functions: getdata and numsum. Function getdata accepts non-empty array of distinct integers from user in 1D int array data member and a targetsum in another data member. The function numsum adds any two elements from an input array which is equal to targetsum and return an array of resulting two elements, in any order. If no two numbers sum up to the target sum, the function should return an empty array. Note that the target sum is to be obtained by summing two different integers in the array; you can’t add a single integer to itself in order to obtain the target sum. You can assume that there will be at most one pair of numbers summing up to the target sum. Use constructor. Use extra variables if needed.



**Sample Input and Output**

**Test Case 1**

|  |  |  |
| --- | --- | --- |
| **Input Parameters** | **Values** | **Expected Output** |
| 1D Array | [3,5,-4,8,11,1,-1,7] | [8,7] |
| targetsum | 15 |

**Test Case 2**

|  |  |  |
| --- | --- | --- |
| **Input Parameters** | **Values** | **Expected Output** |
| 1D Array | [3,5,-4,8,11,1,-1,6] | [ ] |
| targetsum | 15 |

Add your Code Here

import java.util.Scanner;  
class assignment2 {  
 int size,k=1; int a[];  
  
 int targetSum;  
  
 public static void main(String[] args) { Scanner sc = new Scanner(System.*in*);  
  
 assignment2 demo = new assignment2(); System.*out*.println("Enter size of array"); demo.size = Integer.*parseInt*(sc.nextLine());  
  
 System.*out*.println("Enter array"); demo.a = new int[demo.size];  
 for (int i = 0; i < demo.size; i++) {  
 demo.a[i] = Integer.*parseInt*(sc.nextLine());  
 }  
 System.*out*.println("Enter targeted sum"); demo.targetSum = Integer.*parseInt*(sc.nextLine()); for (int i = 0; i < demo.size; i++) {  
 for (int j = i+1; j < demo.size; j++) {  
 if (demo.a[i] + demo.a[j] == demo.targetSum) { demo.k= 0;  
 System.*out*.println("(" + demo.a[i] + ", " + demo.a[j] + ")");  
 }  
 if (demo.k != 0 && i == demo.size-2 && i== demo.size-1) { System.*out*.println("No such combination");  
 }  
 }  
  
 }  
 }  
}

**LAB ASSIGNMENT 03**

**Problem Statement:**

Calculate area of triangle, square & circle using function overloading. Function parameter accept from user. Create Base Class **Shape** and Derived Classes **Triangle, Square, Circle** respectively. Implement **getInputs()** Method for accepting inputs, and Overload **setArea()** method for calculating area of respective shapes.

Use Class **Tester** for creating objects.

**Sample Input and Output**

|  |  |  |
| --- | --- | --- |
| **Sample Input/Parameter for Triangle** | **Values** | **Expected Output** |
| Height (H) | 50 | 2500 |
| Base (B) | 100 |

|  |  |  |
| --- | --- | --- |
| **Sample Input/Parameter for Circle** | **Values** | **Expected Output** |
| π (Pie) | 3.14 | 7853.98 |
| Radius (R) | 50 |

|  |  |  |
| --- | --- | --- |
| **Sample Input/Parameter for Square** | **Values** | **Expected Output** |
| Side (S) | 15 | 225 |

**Add your code here**

import java.util.Scanner;  
  
public class Main { double are;  
  
 void area(int r, Double pi) { are = pi \* r \* r;  
 }  
  
 void area(int base, int height) { are = 0.5 \* base \* height;  
 }  
  
 void area(int side) { are = side \* side;  
 }  
  
 public static void main(String[] args) { Scanner sc = new Scanner(System.*in*);  
  
 System.*out*.println("Press 1 for finding area of circle"); System.*out*.println("Press 2 for finding area of triangle"); System.*out*.println("Press 3 for finding area of square"); int key = Integer.*parseInt*(sc.nextLine());  
 switch (key) { case 1:  
  
 System.*out*.println("Enter radius of circle"); int radius = Integer.*parseInt*(sc.nextLine()); circle one = new circle(radius);  
 break; case 2:  
 System.*out*.println("Enter base and height of triangle"); int base = Integer.*parseInt*(sc.nextLine());  
 int height = Integer.*parseInt*(sc.nextLine()); triangle two = new triangle(base, height); break;  
  
 case 3:  
  
 System.*out*.println("Enter side of square"); int side = Integer.*parseInt*(sc.nextLine()); square three = new square(side);  
 break;  
  
 default:  
 break;  
 }  
 }  
  
}  
  
  
class circle extends Main { circle(int radius) {  
 area(radius, 3.14);  
 System.*out*.println("Area of circle is " + are);  
}  
}  
  
class triangle extends Main { triangle(int base, int height) {  
 area(base, height);  
 System.*out*.println("Area of triangle is " + are);  
}  
}  
  
class square extends Main { square(int side) {  
 area(side);  
 System.*out*.println("Area of circle is " + are);  
}  
}

**LAB ASSIGNMENT 04**

**Problem Statement:**

Write a program for following exception; develop a suitable scenario in which the following exceptions occur:

* 1. divide by zero
  2. Array index out of bounds exception
  3. Null pointer Exception

**Develop a menu driven program for handle the above listed exception.**

**Sample Input and Output**

|  |  |
| --- | --- |
| **Sample Input/Parameter** | **Expected Output** |
| Try to divide a number by zero | You shouldn’t divide a number by zero. |
| Try to access the array index which does not exist. | OOPs!!!Array Index 7 out of bounds for length 6. |
| Try to find the length of String in method (pass parameter string as null) | Null Pointer Exception arises!! |

**Add your code here**

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 int a[]= new int[5];  
 String s = null;  
 System.*out*.println("Hello Everyone");  
 System.*out*.println("Enter the Exception");  
 System.*out*.println("choose 1:Arithmetic 2:Array 3:String");  
 Scanner input = new Scanner(System.*in*);  
 int i = input.nextInt();  
 switch(i){  
 case 1:{  
 try {  
 System.*out*.println(10/0);  
 }  
 catch (ArithmeticException ae) {  
 System.*out*.println(ae.getMessage());  
 }  
 finally {  
 System.*out*.println("The Code Ends");  
 }  
 break;  
 }  
 case 2:{  
 try {  
 System.*out*.println(a[10]);  
 }  
 catch (ArrayIndexOutOfBoundsException b) {  
 System.*out*.println(b.getMessage());  
 }  
 finally {  
 System.*out*.println("The Code Ends");  
 }  
 break;  
 }  
 case 3:{  
 try {  
 System.*out*.println(s.length());  
 }  
 catch (NullPointerException sb) {  
 System.*out*.println(sb.getMessage());  
 }  
 finally {  
 System.*out*.println("The Code Ends");  
 }  
 break;  
  
 }  
 }  
  
 }  
}

**LAB ASSIGNMENT 05**

**Problem Statement:**

A fashion E-commerce company keeps a track of all the orders using an **ArrayList** and a class Order. Implement class **Order** and retrieve and return the list of items present in all the orders. Implement the logic inside **getItems()** method. Consider true and false value given in input for cash on delivery option.

Test the functionalities using the **main()** method of the **Tester** class.

**Sample Input and Output**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
| orders=[Order(101,itemNames=[Jeans, Shirt, Belt],true), Order(102,itemNames=[Tie,Shirt],true),Order(103,itemNames=[Tshirt,Socks,Tie],true) | [Jeans,Shirt,Belt,Tie,Shirt,Tshirt,Socks,Tie] |
| orders=[Order(311,itemNames=[Sportswear, Dumbbell],true), Order(102,itemNames=[, Jeans],true),Order(103,itemNames=[Smartwatch,Fitnessband,Joggers],true) | Sportswear,Dumbbell,Smartwatch,Fitnessband,Joggers] |

**CODE :-**

import java.util.List;  
import java.util.ArrayList;  
  
class Order {  
 //private  
 int orderId;  
 List<String> itemNames;  
 boolean cashOnDelivery;  
  
 // public int getOrderId() {  
// return orderId;  
// }  
// public void setOrderId(int orderId) {  
// this.orderId = orderId;  
// }  
// public List<String> getItemNames() {  
// return itemNames;  
// }  
// public void setItemNames(List<String> itemNames) {  
// this.itemNames = itemNames;  
// }  
// public boolean isCashOnDelivery() {  
// return cashOnDelivery;  
// }  
// public void setCashOnDelivery(boolean cashOnDelivery) {  
// this.cashOnDelivery = cashOnDelivery;  
// }  
 public Order(int orderId, List<String> itemNames, boolean cashOnDelivery) {  
  
 this.orderId = orderId;  
 this.itemNames = itemNames;  
 this.cashOnDelivery = cashOnDelivery;  
 }  
}  
  
class Tester {  
 public static List<String> getItems(List<Order> orders) {  
  
 List<String> items = new ArrayList<String>();  
  
 for( Order order:orders){  
  
 items.addAll(order.itemNames);  
 }  
 return items;  
 }  
  
 public static void main(String[] args) {  
 List<Order> orders = new ArrayList<Order>();  
  
 List<String> items1 = new ArrayList<String>();  
 items1.add("Jeans");  
 items1.add("Shirt");  
 items1.add("Belt");  
 //orders.add(new Order(101, items1, true));  
  
 List<String> items2 = new ArrayList<String>();  
 items2.add("Tie");  
 items2.add("Shirt");  
 //orders.add(new Order(102, items2, true));  
  
 List<String> items3 = new ArrayList<String>();  
 items3.add("Tshirt");  
 items3.add("Socks");  
 items3.add("Tie");  
 //orders.add(new Order(103, items3, true));  
  
  
 Order o1=new Order(101, items1, true);  
 Order o2=new Order(102, items2, true);  
 Order o3=new Order(103, items3, true);  
 orders.add(o1);  
 orders.add(o2);  
 orders.add(o3);  
  
 List<String> items = *getItems*(orders);  
 System.*out*.println(items);  
 }  
}

**LAB ASSIGNMENT 06**

**Problem Statement:**

Implement various operations like add, update and delete the data using JDBC Connectivity.

**Code:**

import java.sql.\*;  
import java.util.\*;  
public class Main {  
  
 public static void main(String[] args) {  
  
 String url="jdbc:mysql://localhost:3307/vit";  
 String user="root";  
 String pwd=" ";  
  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.println("Enter rollNo: ");  
 int rno = sc.nextInt();  
  
 System.*out*.println("Enter Sname: ");  
 String sname = sc.next();  
  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection con=DriverManager.*getConnection*(url,user,pwd);  
 Statement st=con.createStatement();  
  
 String sqlInsert="insert into CSA1 values('"+rno+"','"+sname+"')";  
 st.executeUpdate(sqlInsert);  
 System.*out*.println("record inserted successfully");  
  
  
 String sql="select \* from CSA1";  
 ResultSet rs= st.executeQuery(sql);  
 while(rs.next()) {  
 System.*out*.println("RollNo: "+rs.getInt(1));  
 System.*out*.println("Sname: "+rs.getString(2));  
  
 }  
  
  
  
 con.close();  
 }  
 catch(Exception ex) {  
 System.*out*.println(ex);  
 }  
  
 }  
  
}