**OBJECT ORIENTED PROGRAMMING LAB**

**Submitted by:**

**FATHIMA NOVRIN**

**TKM20MCA-2017**

**COURSE OUTCOME-1**

**PROGRAM-1**

**AIM:** Define a class ‘product’ with data members pcode, pname and price. Create 3 objects of the class and find the product having the lowest price.

**ALGORITHM:**

Step1-Create a class “product”

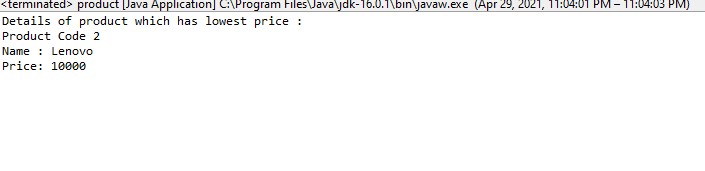
Step2-Initialize product code,name,price.

Step3-Write function to display product details

Step4-Write function to find the lowest price among products

|  |  |
| --- | --- |
| **PROGRAM CODE** | public class product {    int pcode , price;  String pname;    public static void main(String [] args)  { //objects are created from product class  product obj1 = new product();  product obj2 = new product();  product obj3 = new product();  //values are given to the objects  obj1.pcode = 1;  obj1.pname = "HP";  obj1.price = 20000;    obj2.pcode = 2;  obj2.pname = "Lenovo";  obj2.price = 10000;    obj3.pcode = 3;  obj3.pname = "Asus";  obj3.price = 50000;    System.out.println("Details of product which has lowest price : ");    if(obj1.price < obj2.price && obj1.price < obj3.price)  {  System.out.println("Product Code : "+ obj1.pcode+ "\n" + "Name : "+ obj1.pname+ "\n" + "Price: " + obj1.price );  }  else if (obj2.price < obj1.price && obj2.price < obj3.price)  {  System.out.println("Product Code "+ obj2.pcode+ "\n" + "Name : "+ obj2.pname+ "\n" + "Price: " + obj2.price );  }  else  {  System.out.println("Product Code : "+ obj3.pcode+ "\n" + "Name "+ obj3.pname+ "\n" + "Price : " + obj3.price );  }  }  } |

**OUTPUT:**



**PROGRAM-2**

**AIM:** Read 2 matrices from the console and perform matrix addition.

**ALGORITHM:**

Step1-Create a class matrix

Step2- initialize the variables

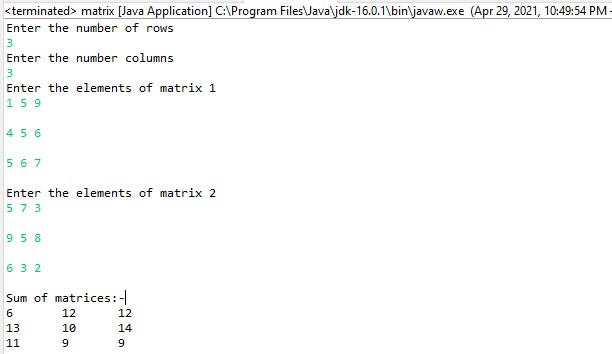
Step3- Input the rows and columns from user

Step4-Input elements in the 2 matrices

Step5-find the sum of 2 matrices

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java.util.Scanner;  public class matrix  {  public static void main(String args[])  {  int row, col,i,j;  Scanner in = new Scanner(System.in);  System.out.println("Enter the number of rows");  row = in.nextInt();  System.out.println("Enter the number columns");  col = in.nextInt();  int a[][] = new int[row][col];  int b[][] = new int[row][col];  int tot[][] = new int[row][col];  System.out.println("Enter the elements of matrix 1");  for ( i= 0 ; i < row ; i++ )  {  for ( j= 0 ; j < col ;j++ )  a[i][j] = in.nextInt();  System.out.println();  }  System.out.println("Enter the elements of matrix 2");  for ( i= 0 ; i < row ; i++ )  {  for ( j= 0 ; j < col ;j++ )  b[i][j] = in.nextInt();  System.out.println();  }  for ( i= 0 ; i < row ; i++ )  for ( j= 0 ; j < col ;j++ )  tot[i][j] = a[i][j] + b[i][j] ;  System.out.println("Sum of matrices:-");  for ( i= 0 ; i < row ; i++ )  {  for ( j= 0 ; j < col ;j++ )  System.out.print(tot[i][j]+"\t");  System.out.println();  }  }    } |

**OUTPUT:**



**PROGRAM-3**

**AIM:**Add complex numbers

**ALGORITHM:**

Step1-Create a class complex

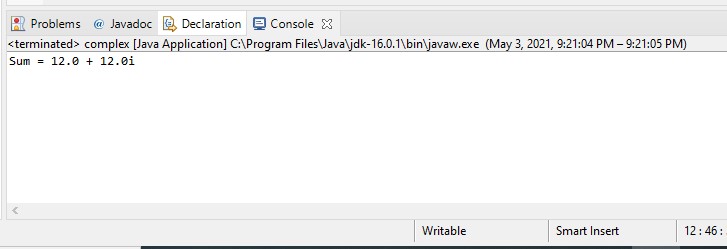
Step2- initialize the variables and objects

Step3- enter the values for imaginary and real part

Step4- add the complex numbers

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  public class complex {  double real;  double imag;  public complex(double real,double imag) {  this.real = real;  this.imag = imag;  }  public static void main(String[] args) {  complex n1 = new complex(8, 7),  n2 = new complex(4, 5.0),  temp;  temp = add(n1, n2);  System.out.printf("Sum = %.1f + %.1fi", temp.real, temp.imag);  }  public static complex add(complex n1, complex n2)  {  complex temp = new complex(0.0, 0.0);  temp.real = n1.real + n2.real;  temp.imag = n1.imag + n2.imag;  return(temp);  }  } |

**OUTPUT:**



**PROGRAM-4**

**AIM:** Read a matrix from the console and check whether it is symmetric or not.

**ALGORITHM:**

Step1 – initialize the variables and objects

Step2 – ask for the number of rows and columns of the matrix

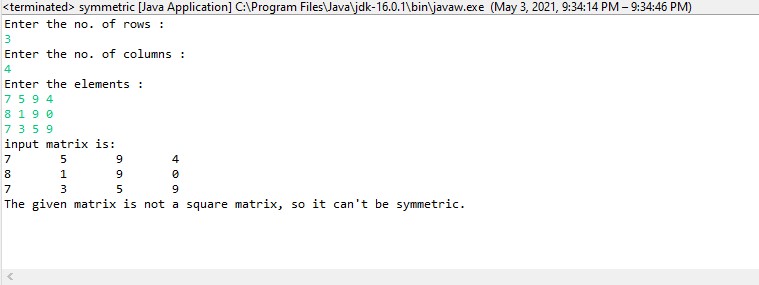
Step3 – ask for the elements in the matrix

Step 4 – find the transpose

Step5- check if the elements in the given matrix and transposed matrix is same or not i.e., symmetric or not.

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class symmetric {    public static void main(String[] args)  {  Scanner s = new Scanner(System.in);  System.out.println("Enter the no. of rows : ");  int rows = s.nextInt();  System.out.println("Enter the no. of columns : ");  int cols = s.nextInt();  int matrix[][] = new int[rows][cols];  System.out.println("Enter the elements :");    for (int i = 0; i < rows; i++)  {  for (int j = 0; j < cols; j++)  {  matrix[i][j] = s.nextInt();  }  }  System.out.println("input matrix is:");    for (int i = 0; i < rows; i++)  {  for (int j = 0; j < cols; j++)  {  System.out.print(matrix[i][j]+"\t");  }    System.out.println();  }    if(rows != cols)  {  System.out.println("The given matrix is not a square matrix, so it can't be symmetric.");  }  else  {  boolean symmetric = true;    for (int i = 0; i < rows; i++)  {  for (int j = 0; j < cols; j++)  {  if(matrix[i][j] != matrix[j][i])  {  symmetric = false;  break;  }  }  }    if(symmetric)  {  System.out.println("The given matrix is symmetric");  }  else  {  System.out.println("The given matrix is not symmetric");  }  }    }  } |

**OUTPUT:**



**PROGRAM-5**

**AIM:** Create CPU with attribute price. Create inner class Processor (no. of cores, manufacturer) and static nested class RAM (memory, manufacturer). Create an object of CPU and print information of Processor and RAM.

**ALGORITHM:**

Step1- create a public class CPU with attribute price

Step2 – create inner class processor with number of cores, manufacture attributes

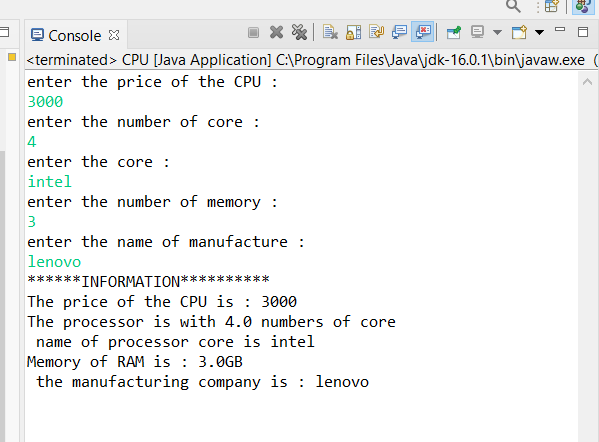
Step3 – create a static nested class RAM with memory and company attributes

Step4 – ask to enter the values

Step5 – display the information

|  |  |
| --- | --- |
| **PROGRAM CODE** | package product;  import java.util.Scanner;  import java.lang.String;  public class CPU {  int price;    public class Processor {    float num\_of\_core;  String manufacture;    void info(float a, String name) {    num\_of\_core = a;  manufacture = name;    System.out.println("The processor is with "+ num\_of\_core + " numbers of core \n name of processor core is "+manufacture);  }  }  static class RAM {    float memory;  String company;    void pinfo(float b , String com) {    memory =b;  company = com;    System.out.println("Memory of RAM is : "+ memory +"GB \n the manufacturing company is : "+company);    }    }  public static void main(String[] args) {    CPU obj = new CPU();    CPU.Processor obj1 = obj.new Processor();    CPU.RAM obj2 = new CPU.RAM();      Scanner in = new Scanner(System.in);    System.out.println("enter the price of the CPU : ");    obj.price = in.nextInt();    System.out.println("enter the number of core : ");    float a = in.nextFloat();    System.out.println("enter the core : ");    String name = in.next();        System.out.println("enter the number of memory : ");    float b = in.nextInt();    System.out.println("enter the name of manufacture : ");    String man = in.next();    System.out.println("\*\*\*\*\*\*INFORMATION\*\*\*\*\*\*\*\*\*\*");    System.out.println("The price of the CPU is : "+ obj.price);    obj1.info(a,name);    obj2.pinfo(b, man);  }  } |

**OUTPUT:**



**COURSE OUTCOME – 2**

**PROGRAM-6**

**AIM:**.Program to Sort strings

**ALGORITHM:**

Step1-Create a class sort

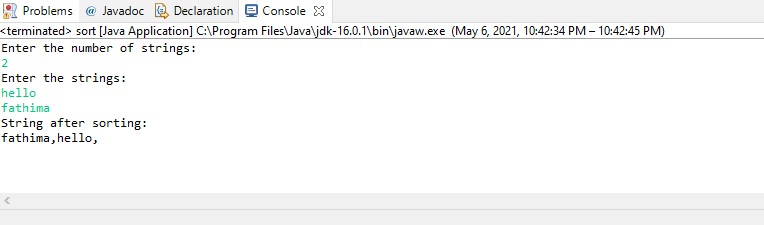
Step2-Input the number strings from user

Step3-enter the string

Step4- sort the given strings

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class sort {  public static void main(String[] args) {  int count;  String str;  Scanner sc=new Scanner(System.in);  System.out.println("Enter the number of strings: ");  count=sc.nextInt();    String str\_arr[]=new String[count];  Scanner sc1=new Scanner(System.in);  System.out.println("Enter the strings: ");  for(int i=0;i<count;i++)  {  str\_arr[i]=sc1.nextLine();  }  sc.close();  sc1.close();    for(int i=0;i<count;i++)  {  for(int j=i+1;j<count;j++)  {  if(str\_arr[i].compareTo(str\_arr[j])>0)  {  str=str\_arr[i];  str\_arr[i]=str\_arr[j];  str\_arr[j]=str;  }  }  }    System.out.println("String after sorting: ");  for(int i=0;i<count;i++)  {  System.out.print(str\_arr[i]+",");  }}} |

**OUTPUT:**



**PROGRAM-7**

**AIM:** Search an element in an array.

**ALGORITHM:**

Step1-Create a class search

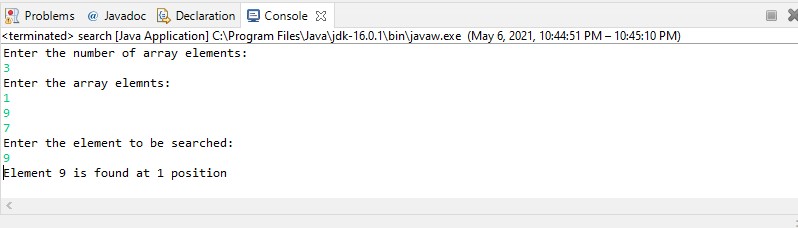
Step2- enter array of elements

Step3- search for an element

Step4- find the element with its index numberr

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class search {  public static void main(String[] args) {  int n,key,flag=0;  Scanner sc=new Scanner(System.in);  System.out.println("Enter the number of array elements: ");  n=sc.nextInt();  int[] arr=new int[n];  Scanner sc1=new Scanner(System.in);  System.out.println("Enter the array elemnts: ");  for(int i=0;i<n;i++)  {  arr[i]=sc1.nextInt();  }    System.out.println("Enter the element to be searched: ");  Scanner sc2=new Scanner(System.in);  key=sc2.nextInt();    for(int i=0;i<n;i++)  {  if(arr[i]==key)  {  System.out.println("Element "+key+" is found at "+i+ " position");  flag=1;  break;  }  }  if(flag==0)  {  System.out.println("Element"+key+"not found");  }  }  } |

**OUTPUT:**



**PROGRAM-8**

**AIM:** Perform string manipulations

**ALGORITHM:**

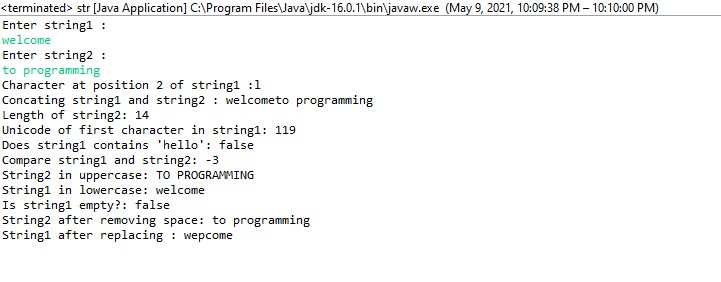
Step1-Create a class str

Step2-enter the strings

Step3-perform string operations

Step4- note the corresponding result.

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class str {  public static void main(String[] args) {  Scanner sc=new Scanner(System.in);  String str1,str2;  System.out.println("Enter string1 : ");  str1=sc.nextLine();  System.out.println("Enter string2 : ");  str2=sc.nextLine();  System.out.println("Character at position 2 of string1 :"+str1.charAt(2));  System.out.println("Concating string1 and string2 : "+str1.concat(str2));  System.out.println("Length of string2: "+str2.length());  System.out.println("Unicode of first character in string1: "+str1.codePointAt(0));  System.out.println("Does string1 contains 'hello': "+str1.contentEquals("hello"));  System.out.println("Compare string1 and string2: "+str2.compareTo(str1));  System.out.println("String2 in uppercase: "+str2.toUpperCase());  System.out.println("String1 in lowercase: "+str1.toLowerCase());  System.out.println("Is string1 empty?: "+str1.isEmpty());  System.out.println("String2 after removing space: "+str2.trim());  System.out.println("String1 after replacing : "+str1.replace('l', 'p'));  }  } |

**OUTPUT:**

**PROGRAM-9**

**AIM:** Program to create a class for Employee having attributes eNo, eName eSalary. Read n employ information and Search for an employee given eNo, using the concept of Array of Objects.

**ALGORITHM:**

Step1- create a class named Employee

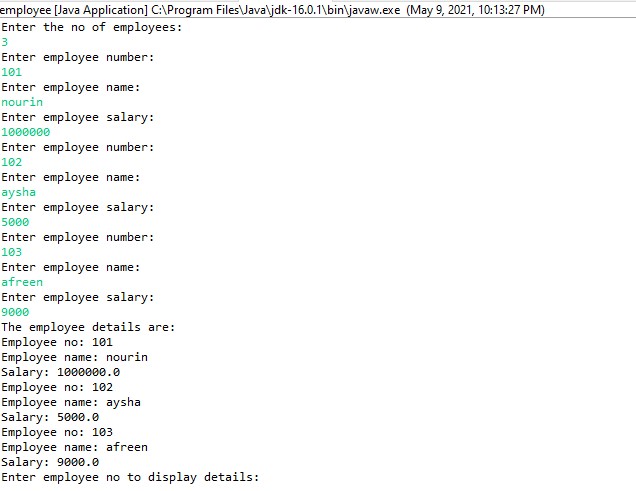
Step2- create attributes eNo,eName,eSalary

Step3- take n employ information

Step4- search and find employee using the eNo.

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class employee {  int eno;  String ename;  double esalary;    void getinfo()  {  Scanner sc=new Scanner(System.in);  System.out.println("Enter employee number: ");  eno=sc.nextInt();  Scanner sc1=new Scanner(System.in);  System.out.println("Enter employee name: ");  ename=sc1.nextLine();  Scanner sc2=new Scanner(System.in);  System.out.println("Enter employee salary: ");  esalary=sc2.nextDouble();  }  void display()  {  System.out.println("Employee no: "+eno);  System.out.println("Employee name: "+ename);  System.out.println("Salary: "+esalary);    }  public static void main(String args[])  {  int n;  Scanner sc3=new Scanner(System.in);  System.out.println("Enter the no of employees: ");  n=sc3.nextInt();  employee e[]=new employee[n];    for(int i=0;i<n;i++)  {  e[i]=new employee();  e[i].getinfo();  }  System.out.println("The employee details are:");  for(int i=0;i<n;i++)  {  e[i].display();  }    int no,flag=0;  Scanner sc4=new Scanner(System.in);  System.out.println("Enter employee no to display details: ");  no=sc4.nextInt();    for(int i=0;i<n;i++)  {  if(no==e[i].eno)  {  e[i].display();  flag=1;  break;  }  }  if(flag==0)  {  System.out.println("No such employee");  }  }    } |

**OUTPUT:**



**COURSE OUTCOME-3**

**PROGRAM-10**

**AIM**: Area of different shapes using overloaded functions

**ALGORITHM:**

Step 1: Start

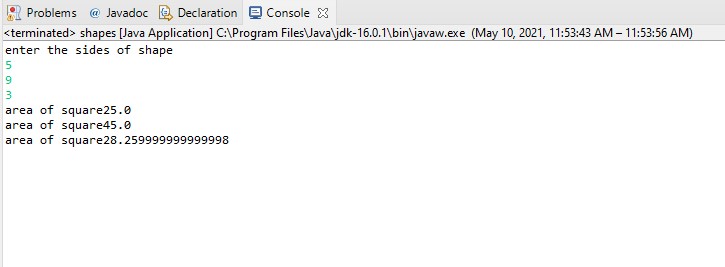
Step 2: Define the main class

Step 3: Define methods with the same method name that performs the area operation for each shape

Step 4: Display the areas of each shapes

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  public class shapes {  float area(float a)  {  float sqrarea=a\*a;  System.out.println("area of square"+sqrarea);  return sqrarea;  }  float area(float a,float b)  {  float rctarea=a\*b;  System.out.println("area of square"+rctarea);  return rctarea;  }  double area(double r)  {  double carea=3.14\*r\*r;  System.out.println("area of square"+carea);  return carea;  }  public static void main(String[] arg)  {  shapes obj=new shapes();  Scanner s=new Scanner(System.in);  System.out.println("enter the sides of shape");  float a=s.nextInt();  float b=s.nextInt();  double r=s.nextInt();  obj.area(a);  obj.area(a,b);  obj.area(r);    }    } |

**OUTPUT:**



**PROGRAM-11**

**AIM:** Create a class ‘Employee’ with data members Empid, Name, Salary, Address and constructors to initialize the data members. Create another class ‘Teacher’ that inherit the properties of class employee and contain its own data members department, Subjects taught and constructors to initialize these data members and also include display function to display all the data members. Use array of objects to display details of N teachers

**ALGORITHM:**

Step 1: Start

Step 2: create class “employee” with the provided data members and define the constructors

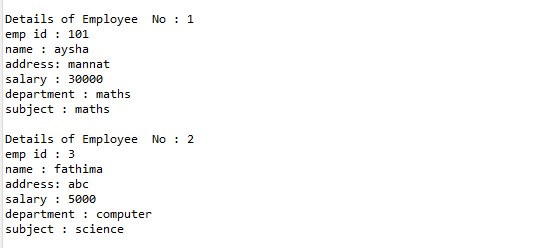
Step 3: create another class “Teachers” that performs inheritance of employee class and define constructors for the same

Step 4: create an array of objects in the corresponding class

Step 5: Display the details for the number of teachers provided

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  class Employees  {  int empid,salary;  String name,address;  public Employees()  {  Scanner inp = new Scanner(System.in);  System.out.println(" Enter emp id : ");  empid = inp.nextInt();  System.out.println("Enter name : ");  name = inp.next();  System.out.println("Enter salary : ");  salary = inp.nextInt();  System.out.println("Enter address : ");  address = inp.next();    }  }  class Teachers extends Employees  {  String dept,sub;  public Teachers()  {  Scanner inp = new Scanner(System.in);  System.out.println("Enter Department : ");  dept = inp.next();  System.out.println("Enter Subject : ");  sub = inp.next();  }    public void display()  {  System.out.println("emp id : " + empid);  System.out.println("name : " + name);  System.out.println("address: " + address);  System.out.println("salary : " + salary);  System.out.println("department : " + dept);  System.out.println("subject : " + sub);    }  }  public class singleinheritance  {  public static void main(String [] args)  {  int no;  Scanner inp2 = new Scanner(System.in);  System.out.println("Enter no of teachers : ");    no = inp2.nextInt();  Teachers obj[] = new Teachers[no];    for(int i=0;i<no;i++)  {  obj[i] = new Teachers();    }      for(int i=0;i<no;i++)  {  System.out.println("\nDetails of Employee No : " + (i+1));  obj[i].display();  }    }  } |

**OUTPUT:**



**PROGRAM-12**

**AIM:** Create a class ‘Person’ with data members Name, Gender, Address, Age and a constructor to initialize the data members and another class ‘Employee’ that inherits the properties of class Person and also contains its own data members like Empid, Company\_name, Qualification, Salary and its own constructor. Create another class ‘Teacher’ that inherits the properties of class Employee and contains its own data members like Subject, Department, Teacherid and also contain constructors and methods to display the data members. Use array of objects to display details of N teachers.

**ALGORITHM:**

Step 1: Start

Step 2: create class “person” with the provided data members and define the constructors

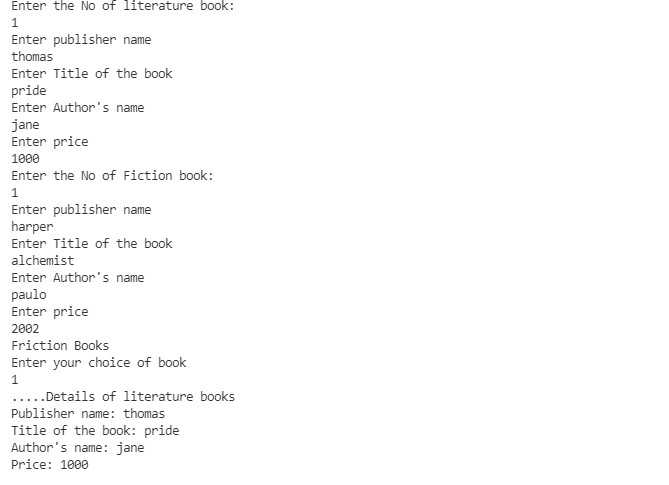
Step 3: create another class “employee” that performs inheritance of person class and another class “teacher” that further inherits the properties of its former class

Step 4: create an array of objects in the corresponding class

Step 5: Display the details of the specific teacher

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java.util.Scanner;  public class Person{  String name;  String gender;  String address;  int age;    public Person()  {  Scanner s=new Scanner(System.in);  System.out.println("Enter name: ");  name=s.next();  System.out.println("Enter gender: ");  gender=s.next();  System.out.println("Enter address: ");  address=s.next();  System.out.println("Enter age: ");  age=s.nextInt();  }  }  class Employees extends Person {  int Empid;  String Company\_name;  String qualification;  double salary;    public Employees()  {  Scanner s1=new Scanner(System.in);  System.out.println("Enter employee id: ");  Empid=s1.nextInt();  System.out.println("Enter Company name: ");  Company\_name=s1.next();  System.out.println("Enter qualification: ");  qualification=s1.next();  System.out.println("Enter salary: ");  salary=s1.nextDouble();  }  }  class Teachers extends Employees  {  int teacherid;  String department;  String subject;    public Teachers()  {  Scanner s2=new Scanner(System.in);  System.out.println("Enter teacher id: ");  teacherid=s2.nextInt();  System.out.println("Enter department details: ");  department=s2.next();  System.out.println("Enter subject taught: ");  subject=s2.next();  }  void display()  {  System.out.println("--------------DETAILS-----------------");  System.out.println("Name: "+name);  System.out.println("Gender: "+gender);  System.out.println("Address: "+address);  System.out.println("Age: "+age);  System.out.println("Employee id: "+Empid);  System.out.println("Company name: "+Company\_name);  System.out.println("Qualification: "+qualification);  System.out.println("Salary: "+salary);  System.out.println("Teacher id: "+teacherid);  System.out.println("Department details: "+department);  System.out.println("Subject: "+subject);    }  public static void main(String args[])  {  int n,i;  Scanner s3=new Scanner(System.in);  System.out.println("Enter the No of employees: ");  n=s3.nextInt();  Teachers T1[]=new Teachers[n];    for(i=0;i<n;i++)  {  System.out.println("Enter details of employee no: "+(i+1));  T1[i]=new Teachers();    }  for(i=0;i<n;i++)  {  T1[i].display();  } }  } |

**OUTPUT:**



**PROGRAM-13**

**AIM:** Write a program has class Publisher, Book, Literature and Fiction. Read the information and print the details of books from either the category, using inheritance

**ALGORITHM:**

Step 1: Start

Step 2: create class “publisher” and initialize its data members

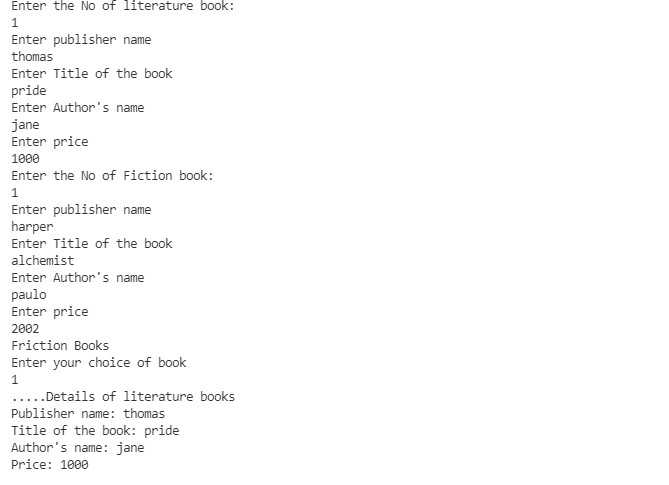
Step 3: create classes book, literature, fiction. Each class inherit from their subsequent previous class and have its own data members

Step 4: create an array of objects in the corresponding class

Step 5: Display the details of the books required

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java.util.Scanner;  public class Publisher {  String Pubname;  Publisher()  {  Scanner s=new Scanner(System.in);  System.out.println("Enter publisher name");  Pubname=s.next();  }  }  class Book extends Publisher  {  String title, author;  int price;    Book()  {  Scanner s=new Scanner(System.in);  System.out.println("Enter Title of the book");  title=s.next();  System.out.println("Enter Author's name");  author=s.next();  System.out.println("Enter price");  price=s.nextInt();  }  }  class Literature extends Book  {  Literature()  {    }  void display()  {  System.out.println("Publisher name: "+Pubname);  System.out.println("Title of the book: "+title);  System.out.println("Author's name: "+author);  System.out.println("Price: "+price);  }  }  class Fiction extends Literature  {  Fiction()  {  System.out.println("Friction Books");  }  void display()  {  super.display();  }    public static void main(String args[])  {  int n;  Scanner s=new Scanner(System.in);    System.out.println("Enter the No of literature book: ");  int a=s.nextInt();  Literature L[]=new Literature[a];  for(int i=0;i<a;i++)  {  L[i]=new Literature();  }    System.out.println("Enter the No of Fiction book: ");  int b=s.nextInt();  Fiction F[]=new Fiction[b];  for(int i=0;i<b;i++)  {  F[i]=new Fiction();  }  int no;  System.out.println("Enter your choice of book");  no=s.nextInt();  int type =no;  switch (no)  {  case 1:  System.out.println(".....Details of literature books");  for(int i=0;i<a;i++)  L[i].display();  break;  case 2:  System.out.println(".....Details of fiction books");  for(int i=0;i<b;i++)  F[i].display();  break;  default:  System.out.println("Wrong input");  }  }  } |

**OUTPUT:**



**PROGRAM-14**

**AIM:** Create classes Student and Sports. Create another class Result inherited from Student and Sports. Display the academic and sports score of a student

**ALGORITHM:**

Step 1: Start

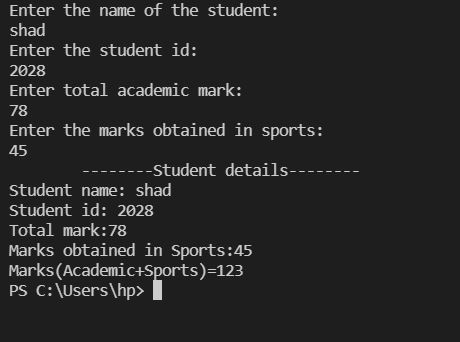
Step 2: Create an interface results.

Step 3: Create classes Student and Sports that implements the interface results

Step 4: Display the academic and sports score of the student

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java.util.Scanner;  class student{  int roll;  String name;  int phy,chem,maths;  student()  {  Scanner s= new Scanner(System.in);  System.out.println("enter the roll num");  roll =s.nextInt();  System.out.println("enter name");  name=s.next();  System.out.println("enter phyics marks");  phy =s.nextInt();  System.out.println("enter chemisty marks");  chem =s.nextInt();  System.out.println("enter maths marks");  maths =s.nextInt();  }    }  class sports extends student  {  int score;  sports()  {  Scanner sc= new Scanner(System.in);  System.out.println("enter sports score");  score=sc.nextInt();  }  }  class result extends sports  {  void diplay()  {    System.out.println("----------------- Academic Details-------------");    System.out.println("Name : " + name);  System.out.println("Roll No : " + roll);  System.out.println("");  System.out.println("-------------------MARKS------------");    System.out.println("Physics :" + phy);  System.out.println("Chemistry :" + chem);  System.out.println("Maths :" + maths);  System.out.println("");  System.out.println("--------------------SPORTS SCORE------------");    System.out.println("Score : " + score);  System.out.println("");  }  }  public class finalresult{  public static void main(String[] args) {  result r =new result();  r.diplay();  }  } |

**OUTPUT:**



**PROGRAM-15**

**AIM:** Create an interface having prototypes of functions area() and perimeter(). Create two classes Circle and Rectangle which implements the above interface. Create a menu driven program to find area and perimeter of objects.

**ALGORITHM:**

Step 1: Start

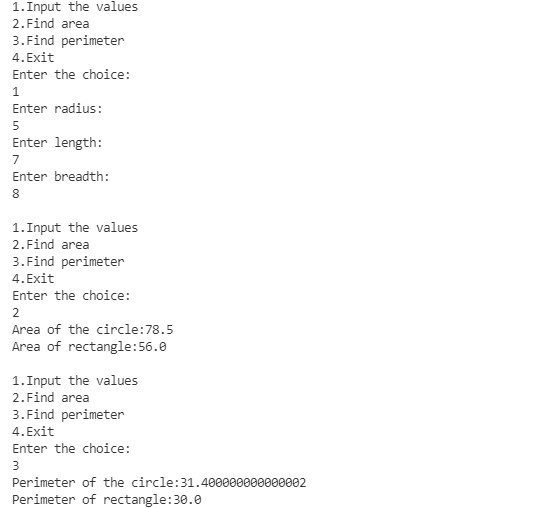
Step 2: Create an interface Calculation that has the methods to take inputs and compute area and perimeter

Step 3: Create classes Circle and Rectangle that implements calculation

Step 4: Display the area and perimeter of circle or rectangle depending upon the choice the user selects.

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java.util.Scanner;  interface calculation{  void input();  void area();  void perimeter();  }  class Circle implements calculation  {  int r;  double pi=3.14,ar,pr;  @Override  public void input()  {  Scanner sc1=new Scanner(System.in);  System.out.println("Enter radius:");  r=sc1.nextInt();  }  @Override  public void area()  {  ar=pi\*r\*r;  System.out.println("Area of the circle:"+ar);  }  @Override  public void perimeter()  {  pr=2\*pi\*r;  System.out.println("Perimeter of the circle:"+pr);  }  }  class Rectangle extends Circle{  int l,b;  double ar,pr;  public void input()  {  super.input();  Scanner sc2=new Scanner(System.in);  System.out.println("Enter length:");  l=sc2.nextInt();  System.out.println("Enter breadth:");  b=sc2.nextInt();  }  public void area()  {  super.area();  ar=l\*b;  System.out.println("Area of rectangle:"+ar);  }  public void perimeter()  {  super.perimeter();  pr=(2\*l)+(2\*b);  System.out.println("Perimeter of rectangle:"+pr);  }  }  public class areaperimeter  {    public static void main(String args[])  {  int choice;  Rectangle obj=new Rectangle();  while(true)  {  Scanner sc3=new Scanner(System.in);  System.out.println("\n" + "1.Input the values"+"\n" + "2.Find area" + "\n" + "3.Find perimeter" + "\n" + "4.Exit");  System.out.println("Enter the choice:");  choice=sc3.nextInt();  switch(choice)  {  case 1:  obj.input();  break;  case 2:  obj.area();  break;  case 3:  obj.perimeter();  break;  case 4:  return;  default:  System.out.println("Enter correct choice:");  }  }  }  } |

**OUTPUT:**



**PROGRAM-16**

**AIM:** Prepare bill with the given format using calculate method from interface :

Order No.

Date:

Product Id Name Quantity unit price Total

101 A 2 25 50

102 B 1 100 100

----------------------------------------- ----------------------------------------- -----------------------------

Net. Amount 150

**ALGORITHM:**

Step 1: Start

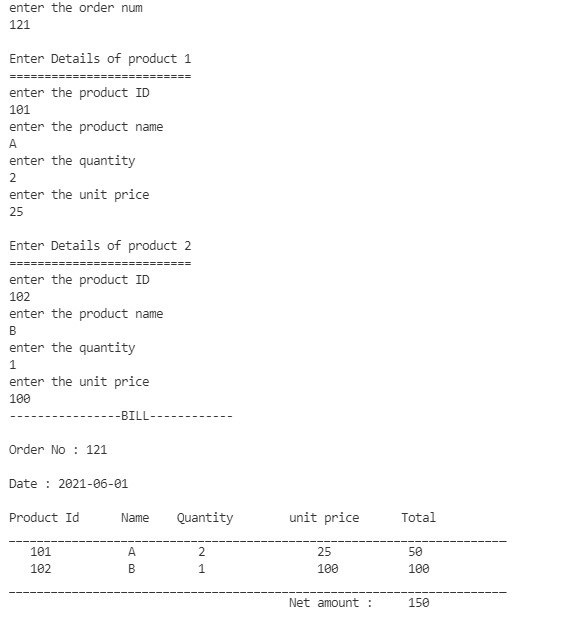
Step 2: Create interface calc that perfroms the calculation operations.

Step 3: Create class bill that implements the interface calc

Step 4: Display the net amount by acquiring the data for the specific inputs

|  |  |
| --- | --- |
| **PROGRAM CODE** | import java .util.\*;  import java.time.\*;  interface bil\_cal  {  public void calc();  }  class calculate implements bil\_cal  {  int ord,date,p\_id,qty,u\_p,tot;  String pname;  public void details()  {  Scanner s=new Scanner(System.in);  System.out.println("enter the product ID");  p\_id=s.nextInt();  System.out.println("enter the product name");  pname=s.next();  System.out.println("enter the quantity");  qty=s.nextInt();  System.out.println("enter the unit price");  u\_p=s.nextInt();  }  public void order()  {  Scanner inp=new Scanner(System.in);  System.out.println("enter the order num");  ord=inp.nextInt();  }  public void calc()  {  tot=qty\*u\_p;  System.out.println(" "+p\_id+"\t\t "+pname+"\t "+qty+"\t\t "+u\_p+"\t "+tot);  }  void display()  {  System.out.println("----------------BILL------------");  System.out.print("\n");  System.out.println("Order No : " + ord);  LocalDate obj = LocalDate.now();  System.out.print("\n");  System.out.println("Date : " + obj);  System.out.print("\n");  System.out.println("Product Id\tName\tQuantity\tunit price\tTotal");  System.out.print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");  System.out.print("\n");    }  }  public class bill  {  public static void main(String[] args)  {  int no,net=0;  Scanner inp2=new Scanner(System.in);  System.out.println("enter number of products");  no=inp2.nextInt();  calculate c= new calculate();  calculate obj[] = new calculate[no];  c.order();  for(int i=0;i<no;i++)  {  System.out.print("\n");  System.out.println("Enter Details of product "+ (i+1));  System.out.println("==========================");    obj[i] = new calculate();  obj[i].details();    }  c.display();  for(int i=0;i<no;i++)  {  obj[i].calc();  net = net + obj[i].tot;  }  System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");  System.out.println(" "+"\t\t "+"\t "+"\t\tNet amount :"+"\t "+net);  }  } |

**OUTPUT:**



**COURSE OUTCOME-4**

**PROGRAM-17**

**AIM:** Create a Graphics package that has classes and interfaces for figures Rectangle, Triangle, Square and Circle. Test the package by finding the area of these figures.

**ALGORITHM:**

Step 1: Start

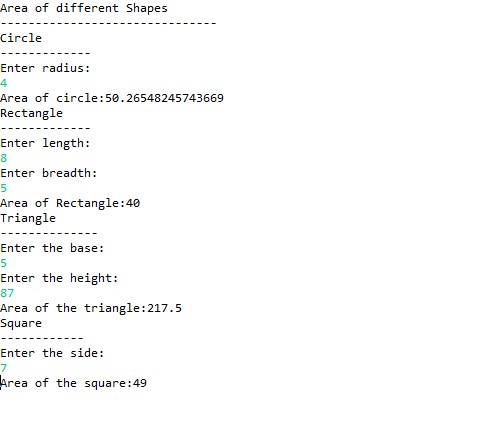
Step 2: To create a package named graphics, create a folder of the same name in the directory.

Step 3: Inside the graphics folder, create modules for finding the areas of rectangle, circle, triangle and square.

Step 4: Outside the graphics folder, write a program to access the modules mention above and print the output

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE**  **Shapes.java** | |  | | --- | | package graphics; | |  | import java.util.Scanner; | |  | interface calculate | |  | { | |  | void carea(); | |  | void rarea(); | |  | void tarea(); | |  | void sarea(); | |  |  | |  | } | |  | public class shapes implements calculate | |  | { | |  | Scanner sc=new Scanner(System.in); | |  | public void carea() | |  | { | |  | System.out.println("Enter radius:"); | |  | int r=sc.nextInt(); | |  | System.out.println("Area of circle:"+(Math.PI\*r\*r)); | |  | } | |  | public void rarea() | |  | { | |  | System.out.println("Enter length:"); | |  | int l=sc.nextInt(); | |  | System.out.println("Enter breadth:"); | |  | int b=sc.nextInt(); | |  | System.out.println("Area of Rectangle:"+(l\*b)); | |  | } | |  | public void tarea() | |  | { | |  | System.out.println("Enter the base:"); | |  | int b = sc.nextInt(); | |  |  | |  | System.out.println("Enter the height:"); | |  | int h = sc.nextInt(); | |  |  | |  | System.out.println("Area of the triangle:"+(0.5\*b\*h)); | |  | } | |  | public void sarea() | |  | { | |  | System.out.println("Enter the side:"); | |  | int s = sc.nextInt(); | |  |  | |  | System.out.println("Area of the square:"+(s\*s)); | |  | } | |  | } | |
| **Areas.java** | import graphics.shapes;  public class areas{  public static void main(String args[])  {  Shapes obj=new Shapes();  System.out.println("Area of different Shapes"+"\n"+"-------------------------------");  System.out.println("Circle"+"\n"+"-------------");  obj.carea();  System.out.println("Rectangle"+"\n"+"-------------");  obj.rarea();  System.out.println("Triangle"+"\n"+"--------------");  obj.tarea();  System.out.println("Square"+"\n"+"------------");  obj.sarea();  }  } |

**OUTPUT:**



**PROGRAM-18**

**AIM:** Create an Arithmetic package that has classes and interfaces for the 4 basic arithmetic operations. Test the package by implementing all operations on two given numbers

**ALGORITHM**

Step 1: Start

Step 2: To create a package named arithmetic, create a folder of the same name in the

directory.

Step 3: Inside arithmetic package, create module to perform addition, subtraction,

multiplication and division of 2 numbers.

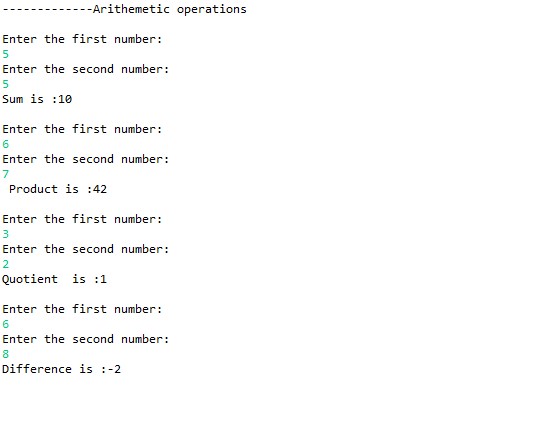
Step 4: Outside the folder, write another program that acssess the above module and print

the output.

Step 5:Stop

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE**  **Main.java** | |  | | --- | | import Arithematic.Math; | |  | public class Main | |  | { | |  |  | |  | public static void main(String[] args) | |  | { | |  |  | |  | Math obj = new Math(); | |  |  | |  | System.out.println(" ARITHMETIC OPERATIONS "); | |  | obj.addition(); | |  | obj.subtraction(); | |  | obj.multiplication(); | |  | obj.division(); | |  | } | |  |  | |  |  | |  |  | |  | } | |
| **math.java** | |  | | --- | | package Arithemetic; | |  |  | |  | import java.util.Scanner; | |  |  | |  | interface operation { | |  | void addition(); | |  | void subtraction(); | |  | void multiplication(); | |  | void division(); | |  | } | |  |  | |  | public class Math implements operation { | |  | public void addition() { | |  | Scanner sc = new Scanner(System.in); | |  | System.out.println("\nEnter the first number:"); | |  | int a = sc.nextInt(); | |  | Scanner sc1 = new Scanner(System.in); | |  | System.out.println("Enter the second number:"); | |  | int b = sc1.nextInt(); | |  | System.out.println("Sum is :"+(a+b)); | |  |  | |  | } | |  | public void subtraction() { | |  | Scanner sc2 = new Scanner(System.in); | |  | System.out.println("\nEnter the first number:"); | |  | int a = sc2.nextInt(); | |  | Scanner sc3 = new Scanner(System.in); | |  | System.out.println("Enter the second number:"); | |  | int b = sc3.nextInt(); | |  |  | |  | System.out.println("Difference is :"+(a-b)); | |  |  | |  | } | |  | public void multiplication() { | |  | Scanner sc4 = new Scanner(System.in); | |  | System.out.println("\nEnter the first number:"); | |  | int a = sc4.nextInt(); | |  | Scanner sc5 = new Scanner(System.in); | |  | System.out.println("Enter the second number:"); | |  | int b = sc5.nextInt(); | |  | System.out.println(" Product is :"+(a\*b)); | |  | } | |  | public void division() { | |  | Scanner sc6 = new Scanner(System.in); | |  | System.out.println("\nEnter the first number:"); | |  | int a = sc6.nextInt(); | |  | Scanner sc7 = new Scanner(System.in); | |  | System.out.println("Enter the second number:"); | |  | int b = sc7.nextInt(); | |  | System.out.println("Quotient is :"+(a/b)); | |  |  | |  | } | |  | } | |

**OUTPUT:**



**PROGRAM-19**

**AIM:**Write a user defined exception class to authenticate the user name and password.

**ALGORITHM:**

Step 1: Start

Step 2: Create two classes UsernameException and PasswordException that represents userdefined Exception

Step 2: Inside the main class accept username and password

Step 3: Try block defines error code,if the username doesn’t satisfies the required condition throws exception

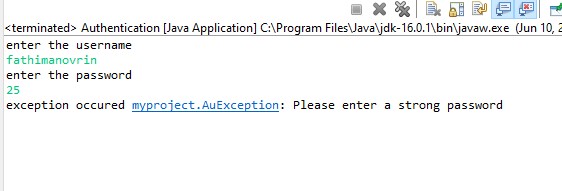
Step 4: If an error occurs in the try block catch block is executed

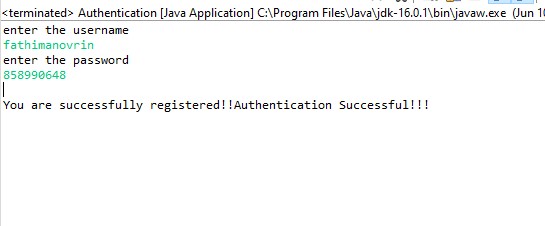
Step 6: Finally block executed regardless of the result.

Step 7: Stop

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  |  | |  | import java.util.Scanner; | |  | class AuException extends Exception { | |  |  | |  |  | |  | /\*\* | |  | \* | |  | \*/ | |  | private static final long serialVersionUID = 1L; | |  |  | |  | AuException(String s){ | |  | super(s); | |  | } | |  |  | |  | } | |  |  | |  | public class Authentication { | |  | public static void main(String[] args) { | |  | Scanner sc= new Scanner(System.in); | |  | String username,password; | |  | System.out.println("enter the username"); | |  | username =sc.nextLine(); | |  | System.out.println("enter the password"); | |  | password=sc.nextLine(); | |  | sc.close(); | |  | try{ | |  | Authenticate(username,password); | |  |  | |  | }catch(Exception e) | |  | { | |  | System.out.println("exception occured "+e); | |  | } | |  |  | |  |  | |  | } | |  |  | |  | public static void Authenticate(String uname, String pwd) throws AuException { | |  | if((uname=="") || (pwd=="")) { | |  | throw new AuException("Fields cannot be empty!!!"); | |  | } | |  | else if (uname.length()<6) { | |  | throw new AuException("Username must be atmost 6 characters!"); | |  | } | |  | else if (pwd.length()<8) { | |  | throw new AuException("Please enter a strong password"); | |  | } | |  | else { | |  | System.out.print("\nYou are successfully registered!!"); | |  | System.out.println("Authentication Successful!!!"); | |  |  | |  | System.exit(0); | |  | } | |  | // TODO Auto-generated method stub | |  |  | |  | } | |  |  | |  | } | |

**OUTPUT:**





**PROGRAM-20**

**AIM:** Find the average of N positive integers, raising a user defined exception for each negative input.

**ALGORITHM:**

Step 1: Start

Step 2: Create a class named NegativeIntegerException that inherits Exception class with a constructor

inside which we call the Exception class constructor and pass error meaasage.

Step 3: Inside the main(), Read the limit of array

Step 4: Inside the try block,read the array and check if any element is less than 0

Step 5: If true, throw NegException with appropriate message.

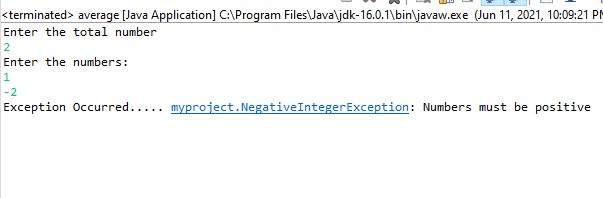
Step 6: Calculate the average of the array and print it

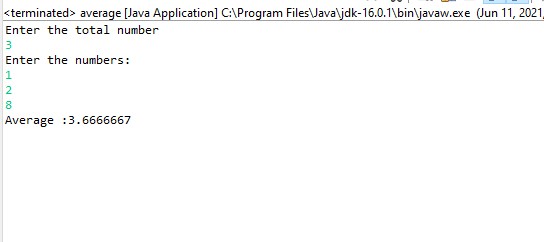
Step 7: Inside the catch exception, Print “Negative Integer”

Step 8:Stop

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  |  | |  | import java.util.Scanner; | |  |  | |  | class NegativeIntegerException extends Exception{ | |  | /\*\* | |  | \* | |  | \*/ | |  | private static final long serialVersionUID = 1L; | |  |  | |  | public NegativeIntegerException(String s) { | |  | super(s); | |  |  | |  | } | |  | } | |  |  | |  | public class average { | |  | public static void main (String args[]) | |  | { | |  | int number[]; | |  | int sum=0; | |  | int i; | |  | float avg; | |  | float count = 0; | |  | Scanner sc=new Scanner(System.in); | |  | System.out.println("Enter the total number"); | |  | int n=sc.nextInt(); | |  | number = new int[n]; | |  | System.out.println("Enter the numbers:"); | |  | for(i=0;i<n;i++) | |  | { | |  | number[i] = sc.nextInt(); | |  | } | |  |  | |  | try { | |  |  | |  | for(i=0;i<n;i++) | |  | { | |  | if(number[i]<0) | |  | { | |  | throw new NegativeIntegerException("Numbers must be positive"); | |  | } | |  | else { | |  | sum = sum + number[i]; | |  | count++; | |  | } | |  |  | |  | } | |  | avg=sum/count; | |  | System.out.println("Average :"+avg); | |  | } | |  |  | |  |  | |  | catch(NegativeIntegerException e) { | |  | System.out.println("Exception Occurred..... "+e); | |  | } | |  |  | |  |  | |  |  | |  | } | |  | } | |

**OUTPUT:**





**PROGRAM-21**

**AIM:** Define 2 classes; one for generating multiplication table of 5 and other for displaying first N prime numbers. Implement using threads. (Thread class)

**ALGORITHM:**

Step 1: Start

Step 2: Create a class named mul that inherits Thread class with member function as run()

Step 3: Inside run(), Print the multiplication table for 5

Step 4: Create a class named prime that inherits Thread class with memebr function run()

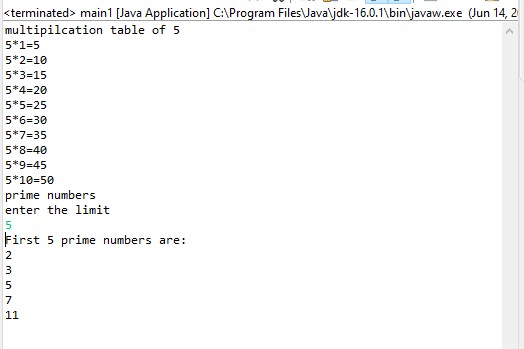
Step 5: Inside run(),Print the prime numbers upto the limit of user’s choice

Step 6: Inside the main(), create an object for the classes and call start() using each object

Step 7:Stop

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  class multiplication extends Thread  {  public void run()  {  System.out.println("multipilcation table of 5");  int n=5;  for(int i=0;i<n;i++)  {  System.out.println(n+"\*"+i+"="+n\*i);      }  }  }  class prime extends Thread  {  public void run()  {  System.out.println("prime numbers");  System.out.println("enter the limit");  Scanner s1= new Scanner(System.in);  int l=s1.nextInt();  int status = 1;  int num = 3;  if (l>= 1)  {  System.out.println("First "+l+" prime numbers are:");    System.out.println(2);  }  for ( int i = 2 ; i <=l ; )  {  for ( int j = 2 ; j <= Math.sqrt(num) ; j++ )  {  if ( num%j == 0 )  {  status = 0;  break;  }  }  if ( status != 0 )  {  System.out.println(num);  i++;  }  status = 1;  num++;  }s1.close();  }  }  public class main {  public static void main(String[] args) throws InterruptedException {  multiplication m= new multiplication();  m.start();  Thread.sleep(100);  prime p=new prime();  p.start();  Thread.sleep(100);  }  } |

**OUTPUT:**



**PROGRAM-22**

**AIM:** Define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using threads. (Runnable Interface)

**ALGORITHM:**

Step 1: Start

Step 2: Create a class named even that implements Runnable interface with function run()

Step 3: Inside run(), we read the limit for printing even numbers and print it using for loop.

Step 4:Create another calss fib that implements Runnable interface with function run().

Step 5: Inside run(), Initialise n1 as 0,n2 as 1 and n3 as 0.

Step 6: Check if n<0, if true, print “Enter a positive number” else goto step 7

Step 7: Repeat step8 to 11 until n3>n

Step 8: Print n1

Step 9: n3=n1+n2

Step 10:n1=n2

Step 11: n2=n3

Step 12: Create object e of even and create an object t1 of Thread with its parameterized

constructor passing e as parameter

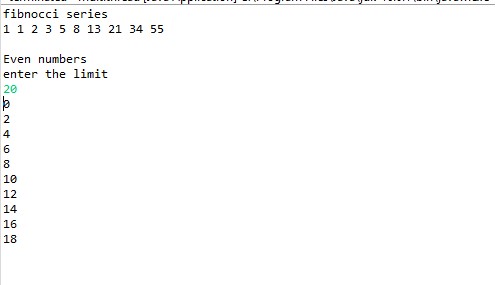
Step 13: Call start() using t1

Step 14: Do the same for class odd with Thread object t2 and call start() using t2

Step 15: Stop

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  class fibnocci implements Runnable{  public void run()  {  System.out.println("fibnocci series");  int n=10;  int n1=1,n2=1,n3,i;  System.out.print(n1+" "+n2);  for(i=2;i<n;i++)  {  n3=n1+n2;  System.out.print(" "+n3);  n1=n2;  n2=n3;  }System.out.println("\n");  }    }  class even implements Runnable  {  public void run()  {  System.out.println("Even numbers");  Scanner a=new Scanner(System.in);  System.out.println("enter the limit");  int lim=a.nextInt();  for(int i=0;i<lim;i++)  {  if(i%2 ==0) {  System.out.println(i+"");  }  }  }  }  public class multithread {  public static void main (String args[]) throws InterruptedException  {  fibnocci f=new fibnocci();  Thread a=new Thread(f);  even e=new even();  Thread b=new Thread(e);  a.start();  a.sleep(100);  b.start();  b.sleep(100);      }  } |

**OUTPUT:**



**PROGRAM-23**

**AIM:** Producer/Consumer using ITC

**ALGORITHM:**

Step 1: Start

Step 2: In PC class (A class that has both produce and consume methods), a linked list of jobs and a capacity of the list is added to check that producer does not produce if the list is full.

Step 3: In Producer class, the value is initialized as 0.

Step 4: We have an infinite outer loop to insert values in the list. Inside this loop, we have a synchronized block so that only a producer or a consumer thread runs at a time.An inner loop is there before adding the jobs to list that checks if the job list is full, the producer thread gives up the intrinsic lock on PC and goes on the waiting state.

Step 5: If the list is empty, the control passes to below the loop and it adds a value in the list.

Step 6: In the Consumer class, we again have an infinite loop to extract a value from the list.

Inside, we also have an inner loop which checks if the list is empty.

Step 7: If it is empty then we make the consumer thread give up the lock on PC and passes

the control to producer thread for producing more jobs.

Step 8: If the list is not empty, we go round the loop and removes an item from the list.

Step 9: In both the methods, we use notify at the end of all statements. The reason is simple,

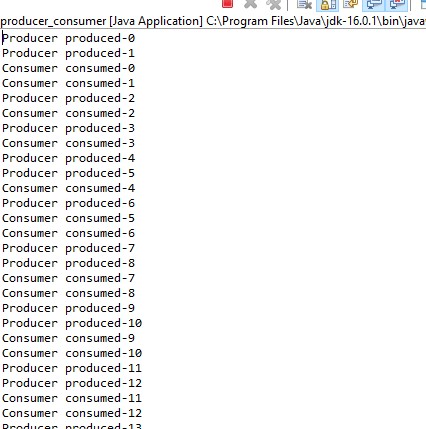
once you have something in list, you can have the consumer thread consume it, or if you have consumed something, you can have the producer produce something.

Step 10: sleep() at the end of both methods just make the output of program run in step wise manner and not display everything all at once so that you can see what actually is happening in the program.

Step 11: Stop

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  |  | |  | import java.util.LinkedList; | |  |  | |  | public class producer\_consumer{ | |  |  | |  | public static void main(String[] args)throws InterruptedException | |  | { | |  |  | |  | final PC pc = new PC(); | |  |  | |  |  | |  | Thread t1 = new Thread(new Runnable() { | |  | @Override | |  | public void run() | |  | { | |  | try { | |  | pc.produce(); | |  | } | |  | catch (InterruptedException e) { | |  | e.printStackTrace(); | |  | } | |  | } | |  | }); | |  |  | |  |  | |  | Thread t2 = new Thread(new Runnable() { | |  | @Override | |  | public void run() | |  | { | |  | try { | |  | pc.consume(); | |  | } | |  | catch (InterruptedException e) { | |  | e.printStackTrace(); | |  | } | |  | } | |  | }); | |  |  | |  |  | |  | t1.start(); | |  | t2.start(); | |  |  | |  |  | |  | t1.join(); | |  | t2.join(); | |  | } | |  |  | |  |  | |  | public static class PC { | |  |  | |  |  | |  | LinkedList<Integer> list = new LinkedList<>(); | |  | int capacity = 2; | |  |  | |  |  | |  | public void produce() throws InterruptedException | |  | { | |  | int value = 0; | |  | while (true) { | |  | synchronized (this) | |  | { | |  |  | |  | while (list.size() == capacity) | |  | wait(); | |  |  | |  | System.out.println("Producer produced-" | |  | + value); | |  |  | |  |  | |  | list.add(value++); | |  |  | |  | notify(); | |  |  | |  | Thread.sleep(1000); | |  | } | |  | } | |  | } | |  |  | |  |  | |  | public void consume() throws InterruptedException | |  | { | |  | while (true) { | |  | synchronized (this) | |  | { | |  |  | |  | while (list.size() == 0) | |  | wait(); | |  |  | |  | int val = list.removeFirst(); | |  |  | |  | System.out.println("Consumer consumed-" | |  | + val); | |  |  | |  |  | |  | notify(); | |  |  | |  |  | |  | Thread.sleep(1000); | |  | } | |  | } | |  | } | |  | } | |  |  | |  | } | |

**OUTPUT:**



**PROGRAM-24**

**AIM:** Program to create a generic stack and do the Push and Pop operations.

**ALGORITHM:**

Step 1: Start

Step 2: Create a class named stack with data members as a(an array),top(set as -1),ch,item,i;

a function named menu()

Step 3: Inside menu(), give choices to push,pop and display the stack

Step 4: If the choice is 1, then check whether the stack is full, else add an element into the stack.

Step 5: If the choice is 2, then check whether the stack is empty, else delete an element into the stack.

Step 6: If the choice is 3, then check whether the stack is empty, else print all the elements in the stack.

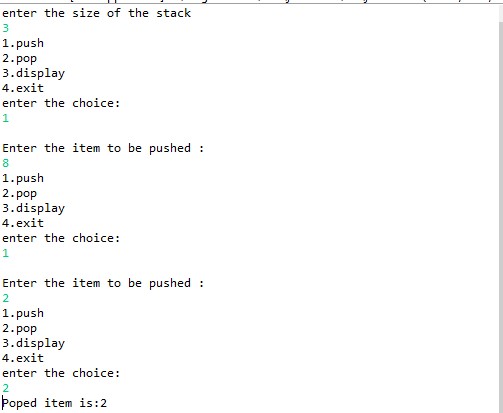
Step 7: If the choice is greater than 4, then print “Invalid option”.

Step 8: Inside the main(), create an object of type stack and call the menu() function.

Step 9:Stop

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.Scanner;  class stack{  int a[]=new int[20];  int choice,top=-1,item;    public void stack\_operation()  {  System.out.println("enter the size of the stack");  Scanner s=new Scanner(System.in);  int n=s.nextInt();  do {    System.out.println("1.push\n2.pop\n3.display\n4.exit");  System.out.println("enter the choice:");  choice=s.nextInt();  switch(choice)  {  case 1 :if(top>=n-1) {  System.out.println("Stack Overflow");  }  else {  System.out.println("\nEnter the item to be pushed : ");  item=s.nextInt();  top=top+1;  a[top]=item;  break;  }    case 2:if(top==-1) {  System.out.println("Stack Underflow");}  else{    item = a[top];  System.out.println("Poped item is:" + item);  top = top-1;  break;  }  case 3: if( top < 0)  {  System.out.println("\nStack is empty\n");    }    else {  System.out.println("\nStack elements :\n\n");  for(int i=top;i>=0;i--)  {  System.out.println(a[i] );  System.out.println("\n");  }  break;  }    case 4:  break;    default: System.out.println("Invalid Choice");  }    }while(choice!=4);  }  }  public class Genericstack {  public static void main (String args[]) {  stack ob=new stack();  ob.stack\_operation();    }  } |

**OUTPUT:**



**PROGRAM-25**

**AIM:** Using generic method perform Bubble sort.

**ALGORITHM:**

Step 1: Start

Step 2: Read number of numbers(N) to sort.

Step 3: Read the numbers

Step 4: Repeat step 5 for i=0 to N-1

Step 5: Repeat for j=i+1 to N

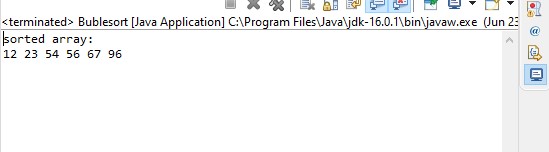
Step 6: Check if array[i] >array[j],

Step 7: if Step 6 true,swap them.End of inner loop.End of outer loop.

Step 8: Print the sorted array

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  class bubles{    void sort(int array[])  {  int n = array.length;  for (int i = 0; i < n-1; i++)  for (int j = 0; j < n-i-1; j++)  if (array[j] > array[j+1])  {    int temp = array[j];  array[j] = array[j+1];  array[j+1] = temp;  }  }  void printarray(int array[])  {  int n = array.length;  for (int i=0; i<n; ++i)  System.out.print(array[i] + " ");  System.out.println();  }  }  public class Bublesort {  public static void main(String args[])  {  bubles b=new bubles();  int array[]= {54,67,23,56,96,12};  b.sort(array);  System.out.println("sorted array:");  b.printarray(array);    }  } |

**OUTPUT:**



**PROGRAM-26**

**AIM:** Maintain a list of Strings using ArrayList from collection framework, perform built-in operations.

**ALGORITHM:**

Step 1:Start

Step 2: Create an object of the class ArrayList.

Step 3: Adding elements to the object of ArrayList using method add() and display.

Step 4: Remove elements of object of ArrayList using method remove() and display .

Step 5: Sort elements of object of ArrayList using method sort() and display.

Step 6: Getting object of list which is present at the specified index using method get() and display.

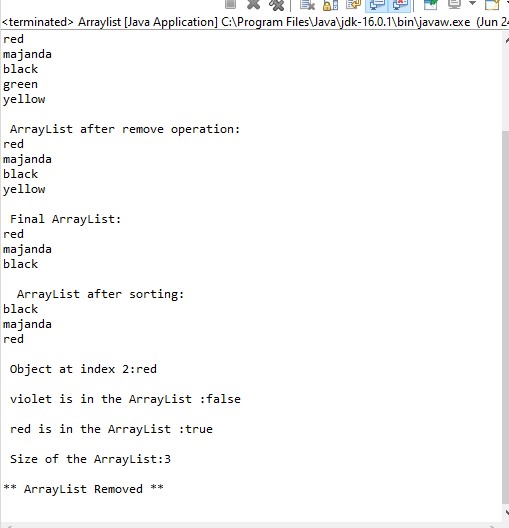
Step 7: Checking weather an element is present in list using method contains() and display True or Flase.

Step 8: Display the size of list using the method size().

Step 9: Clear List using the method clear().

|  |  |
| --- | --- |
| **PROGRAM CODE** | package myproject;  import java.util.ArrayList;  import java.util.Collections;  public class Arraylist {  public static void main(String[] args) {    // Creating ArrayList of type "String" which means we can only add "String" elements    ArrayList<String> obj = new ArrayList<String>();    //adding elements to an ArrayList    obj.add("red");  obj.add("black");  obj.add("green");  obj.add("yellow");    // Displaying elements    System.out.println("\n Original ArrayList:");  for(String str:obj)  System.out.println(str);  //Add element at the given index or replace    obj.add(1, "majanda");  // Displaying elements    System.out.println("\n ArrayList after add operation:");  for(String str:obj)  System.out.println(str);  //Remove elements from ArrayList    obj.remove("green");    // Displaying elements    System.out.println("\n ArrayList after remove operation:");  for(String str:obj)  System.out.println(str);  //Remove element from the specified index    obj.remove(3);  // Displaying elements    System.out.println("\n Final ArrayList:");  for(String str:obj)  System.out.println(str);    //Sorting the ArrayList    Collections.sort(obj);    System.out.println("\n ArrayList after sorting:");  for (String str : obj)  System.out.println(str);      //returns the object of list which is present at the specified index    // obj.get(2);    System.out.println("\n Object at index 2:"+obj.get(2));    // Checks whether the object is in the ArrayList    // obj.contains();    System.out.println("\n violet is in the ArrayList :"+obj.contains("violet"));  System.out.println("\n red is in the ArrayList :"+obj.contains("red"));    //Size of the ArrayList    //obj.size();    System.out.println("\n Size of the ArrayList:"+obj.size());    // remove ArrayList    obj.clear();    System.out.println("\n\*\* ArrayList Removed \*\*");      }  } |

**OUTPUT:**



**PROGRAM-27**

**AIM:** Program to remove all the elements from a linked list

**ALGORITHM:**

Step 1:Start

Step 2: Create an object of the class linkedlist.

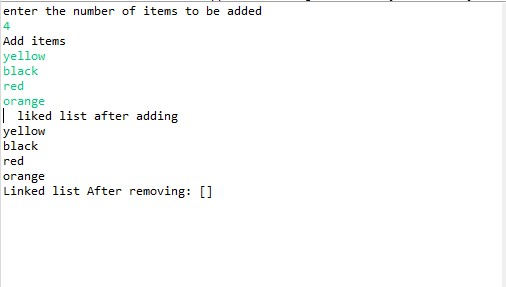
Step 3: Adding elements to the linked list using method add().

Step 4: Remove all the elements of linkedlist using method clear().

Step 5:Display linkedlist.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | |  | | package myproject; | |  | import java.util.\*; | |  |  | |  | public class remove\_elements { | |  | public static void main(String args[]) { | |  | LinkedList<String> list=new LinkedList<String>(); | |  | Scanner sc =new Scanner(System.in); | |  | System.out.println("enter the number of items to be added"); | |  | int num=sc.nextInt(); | |  | System.out.print("Add items "); | |  | for(int i=0;i<num;i++) | |  | { | |  |  | |  | String s=sc.next(); | |  | list.add(s); | |  | } | |  | System.out.print(" "); | |  | System.out.print(" "); | |  | System.out.println("liked list after adding"); | |  |  | |  | Iterator<String> itr=list.iterator(); | |  | while(itr.hasNext()){ | |  | System.out.println(itr.next()); | |  | } | |  | list.clear(); | |  | System.out.println("Linked list After removing: " + list); | |  |  | |  | } | |  |  | |  |  | |  |  | |  | } | |  |  | |

**OUTPUT:**



**PROGRAM-28**

**AIM:** Program to remove an object from the Stack when the position is passed as parameter

**ALGORITHM:**

Step 1:Start

Step 2: Create an object of the class Stack.

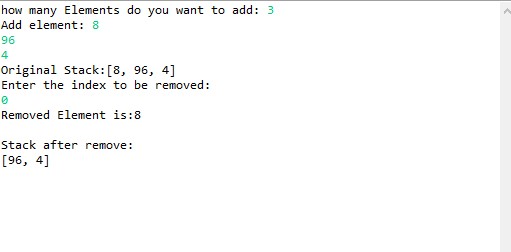
Step 3: Adding elements to the stack using method add().

Step 4: Remove the element of stack at position ‘pos’ using method remove(pos).

Step 5:Display removed element and Stack

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  | import java.util.\*; | |  | public class stack\_obj\_remove { | |  | public static void main(String[] args) | |  | { | |  | Stack<Integer> st=new Stack<Integer>(); | |  | Scanner sc=new Scanner(System.in); | |  | System.out.print("how many Elements do you want to add: "); | |  | int num=sc.nextInt(); | |  | System.out.print("Add element: "); | |  | for(int i=0;i<num;i++) | |  | { | |  |  | |  | int s=sc.nextInt(); | |  | st.add(s); | |  | } | |  | System.out.println("Original Stack:"+st); | |  | System.out.println("Enter the index to be removed:"); | |  | int index = sc.nextInt(); | |  |  | |  | int rm = st.remove(index); | |  |  | |  | System.out.println("Removed Element is:"+rm); | |  |  | |  | System.out.println("\nStack after remove:\n"+st); | |  | } | |  | } | |

**OUTPUT:**



**PROGRAM-29**

**AIM:** Program to demonstrate the creation of queue object using the PriorityQueue class

**ALGORITHM:**

Step 1:Start

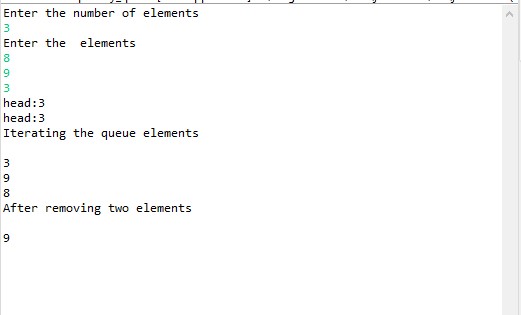
Step 2: Create an object of the class PriorityQueue.

Step 3: Adding elements to the PriorityQueue using method add().

Step 5:Display PriorityQueue.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myprojet; | |  | import java.util.\*; | |  | public class priority\_queue { | |  | public static void main(String[] args) | |  | { | |  | PriorityQueue<String> queue=new PriorityQueue<String>(); | |  | Scanner sr=new Scanner(System.in); | |  | System.out.println("Enter the number of elements"); | |  | int a=sr.nextInt(); | |  | System.out.println("Enter the elements"); | |  | for(int i=0;i<a;i++) | |  | { | |  | String s=sr.next(); | |  | queue.add(s); | |  | } | |  | System.out.println("head:"+queue.element()); | |  | System.out.println("head:"+queue.peek()); | |  | System.out.println("Iterating the queue elements\n "); | |  | Iterator itr=queue.iterator(); | |  | while(itr.hasNext()){ | |  | System.out.println(itr.next()); | |  | } | |  | queue.remove(); | |  | queue.poll(); | |  | System.out.println("After removing two elements \n"); | |  | Iterator<String> itr2=queue.iterator(); | |  | while(itr2.hasNext()){ | |  | System.out.println(itr2.next()); | |  | } | |  |  | |  | } | |  | } | |

**OUTPUT:**



**PROGRAM-30**

**AIM:** Program to demonstrate the addition and deletion of elements in deque

**ALGORITHM:**

Step 1:Start

Step 2: Create an object of the class ArrayDeque.

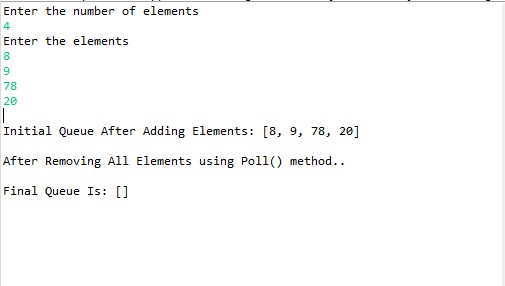
Step 3: Adding elements to the queue using method add().

Step 4: Removing elements of queue using method pop().

Step 5:Display Queue.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  | import java.util.\*; | |  | public class queue { | |  | public static void main(String args[]) | |  | { | |  | Deque<String> dq= new ArrayDeque<String>(); | |  | Scanner scn=new Scanner(System.in); | |  | System.out.println("Enter the number of elements"); | |  | int n=scn.nextInt(); | |  | System.out.println("Enter the elements"); | |  | for(int i=0;i<n;i++) | |  | { | |  | String st=scn.next(); | |  | dq.add(st); | |  |  | |  | } | |  | System.out.println("\nInitial Queue After Adding Elements: " + dq); | |  | System.out.println("\nAfter Removing All Elements using Poll() method.."); | |  |  | |  |  | |  | for(int i =0;i<n;i++) | |  | { | |  |  | |  | dq.pop(); | |  |  | |  | } | |  | System.out.println("\nFinal Queue Is: " + dq); | |  |  | |  | } | |  |  | |  |  | |  |  | |  | } | |

**OUTPUT:**



**PROGRAM-31**

**AIM:** Program to demonstrate the creation of Set object using the LinkedHashset class

**ALGORITHM:**

Step 1:Start

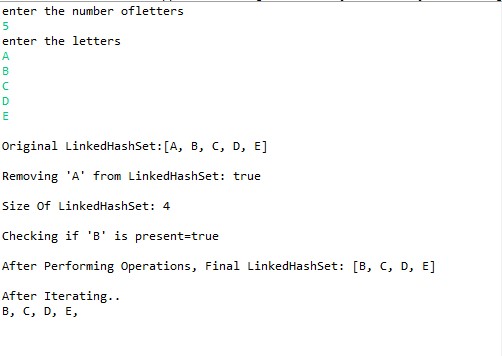
Step 2: Create an object of the class LinkedHashset.

Step 3: Adding elements to the HashSet using method add().

Step 4:Display LinkedHashSet.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | |  | | package myproject; | |  | import java.util.\*; | |  |  | |  |  | |  | public class hash\_set { | |  | public static void main (String args[]) | |  | { | |  | LinkedHashSet<String> hashset = new LinkedHashSet<String>(); | |  | Scanner sc=new Scanner(System.in); | |  | System.out.println("enter the number ofletters"); | |  | int n=sc.nextInt(); | |  | System.out.println("enter the letters"); | |  | for(int i=0;i<n;i++) | |  | { | |  | String s=sc.next(); | |  | hashset.add(s); | |  |  | |  | } | |  | System.out.println("\nOriginal LinkedHashSet:" + hashset); | |  | System.out.println("\nRemoving 'A' from LinkedHashSet: " + hashset.remove("A")); | |  | System.out.println("\nSize Of LinkedHashSet: " + hashset.size()); | |  | System.out.println("\nChecking if 'B' is present=" + hashset.contains("B")); | |  | System.out.println("\nAfter Performing Operations, Final LinkedHashSet: " + hashset); | |  |  | |  | System.out.println("\nAfter Iterating.. "); | |  | for (String s : hashset) | |  | System.out.print(s + ", "); | |  | System.out.println(); | |  |  | |  |  | |  |  | |  |  | |  | } | |  |  |   } |

**OUTPUT:**



**PROGRAM-32**

**AIM:** Write a Java program to compare two hash set.

**ALGORITHM:**

Step 1:Start

Step 2: Create two objects of the class LinkedHashset.

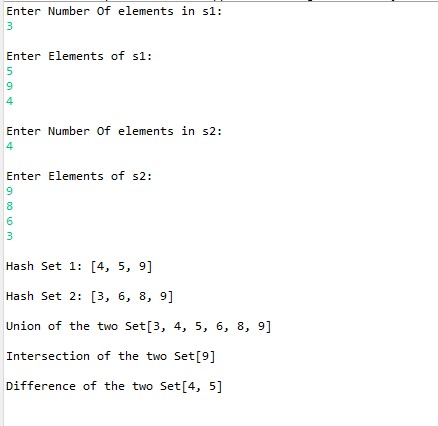
Step 3: Adding elements to the two objects of LinkedHashSet using method add().

Step 4:Checking weather elements of first Hashset is present in second Hashset.

Step 5:If Step 4 is true print Yes,else print No.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  | import java.util.\*; | |  | public class compare\_hashset { | |  | public static void main(String[] args) { | |  | Set<String> s1= new HashSet<String>(); | |  | Set<String> s2 = new HashSet<String>(); | |  |  | |  | Scanner sc=new Scanner(System.in); | |  | System.out.println("Enter Number Of elements in s1: "); | |  | int n=sc.nextInt(); | |  | System.out.println("\nEnter Elements of s1: "); | |  | for(int i =0;i<n;i++) | |  | { | |  | String st=sc.next(); | |  | s1.add(st); | |  |  | |  | } | |  |  | |  | System.out.println("\nEnter Number Of elements in s2: "); | |  | int n1=sc.nextInt(); | |  | System.out.println("\nEnter Elements of s2: "); | |  | for(int i =0;i<n1;i++) | |  | { | |  | String str=sc.next(); | |  | s2.add(str); | |  |  | |  | } | |  | System.out.println("\nHash Set 1: " + s1); | |  | System.out.println("\nHash Set 2: " + s2); | |  |  | |  | //union | |  | Set<String> union = new HashSet<String>(s1); | |  | union.addAll(s2); | |  | System.out.print("\nUnion of the two Set"); | |  | System.out.println(union); | |  |  | |  | //intersection | |  | Set<String> intersection = new HashSet<String>(s1); | |  | intersection.retainAll(s2); | |  | System.out.print("\nIntersection of the two Set"); | |  | System.out.println(intersection); | |  |  | |  | //difference | |  | Set<String> difference = new HashSet<String>(s1); | |  | difference.removeAll(s2); | |  | System.out.print("\nDifference of the two Set"); | |  | System.out.println(difference); | |  | } | |  |  | |  | } | |

**OUTPUT:**



**PROGRAM-33**

**AIM:** Program to demonstrate the working of Map interface by adding, changing and removing elements.

**ALGORITHM:**

Step 1:Start

Step 2:Initialization of a Map using Generics.

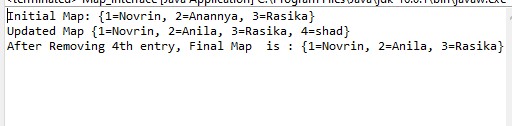
Step 3:Adding values into map using method put() and display.

Step 4:Updating values using method put() by mentioning index of value and display.

Step 5:Removing values from map using method remove() and display.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  | import java.util.\*; | |  | public class Map\_interface { | |  | public static void main (String args[]) | |  | { | |  | Map<Integer,String> hm=new HashMap<>(); | |  |  | |  | hm.put(1, "Novrin"); | |  | hm.put(2, "Anannya"); | |  | hm.put(3, "Rasika"); | |  | System.out.println("Initial Map: "+ hm); | |  |  | |  | hm.put( (2), "Anila"); | |  | hm.put((4), "shad"); | |  | //Updating.. | |  | System.out.println("Updated Map " + hm); | |  |  | |  | //Removing.. | |  | hm.remove(4); | |  |  | |  | // Final Map.. | |  | System.out.println("After Removing 4th entry, Final Map is : "+hm); | |  | } | |  |  | |  | } | |

**OUTPUT:**



**PROGRAM-34**

**AIM:** Program to Convert HashMap to TreeMap

**ALGORITHM:**

Step 1:Get the HashMap to be converted.

Step 2:Create a new TreeMap

Step 3:Pass the hashMap to putAll() method of treeMap

Step 4:Return the formed TreeMap

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PROGRAM CODE** | |  | | --- | | package myproject; | |  | import java.util.\*; | |  |  | |  | public class hash\_to\_treemap { | |  | public static void main(String[] args) { | |  | HashMap<Integer,String>hMap= new HashMap<Integer,String>(); | |  | hMap.put( 10,"red"); | |  | hMap.put( 22, "green" ); | |  | hMap.put(3, "violet" ); | |  | hMap.put( 44, "yellow"); | |  | hMap.put(15, "black"); | |  | System.out.println("HashMap Keys and Values: "+hMap); | |  | System.out.println("\n"); | |  | TreeMap<Integer, String> tMap = new TreeMap<Integer, String>(hMap); | |  | System.out.println("TreeMap Keys and Values: " +tMap); | |  | } | |  |  | |  | } | |

**OUTPUT:**

