LAB CYCLE 3

SUBMITTED BY,
SIJI JOSE
20MCA237

PROGRAM -1

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
STEP 5 :Stop

PROGRAM CODE

area.java	package myproject;
	class area
	{
	void calculateArea(float x)
	{
	System.out.println("Area of the square: "+x*x+" sq units");
	}
	void calculateArea(float x, float y)
	{
	System.out.println("Area of the rectangle: "+x*y+" sq units");
	}
	void calculateArea(double r)
	{
	double area = 3.14*r*r;
	System.out.println("Area of the circle: "+area+" sq units");
	}
	public static void main(String args[]){
	area obj = new area();
	obj.calculateArea(10,20);
	obj.calculateArea(10,22);
	obj.calculateArea(6.1);
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT

```
shapes [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe  (20-May-2021, 12:17:20 pm)

*********SQUARE*********
Enter value of a :
4
Area of square is 16

*********RECTANGLE*********
Enter values of m and n :
2
2
Area of rectangle is 4

**********CIRCLE**********
Enter value of r :
4
Area of circle is 50.26548245743669

***********TRIANGLE**********
Enter the width of the Triangle:
```

PROGRAM -2

AIM: Create a class 'Employee' with data members Empid, Name, Salary, Address and constructors to initialize the data members. Create another class 'Teacher' that inherits the properties of class employees and contains its own data members department, Subjects taught and constructors to initialize these data members and also include a display function to display all the data members. Use an 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
STEP 6: Stop

PROGRAM CODE

employe_e.java	

```
package myproject;
import java.util.*;
public class employe_e {
        int empid;
        String name;
        float salary;
        String address;

        public employe_e(int id, String name,float salary, String
address ) {
                this.empid = id;
                this.name = name;
                this.salary = salary;
                this.address = address;

        }

static class Teacher extends employe_e{
    String Department;
    String Subject;
  public Teacher(int id, String name, float salary, String address,
String dept, String subj) {
    super(id, name, salary, address);


                this.Department = dept;
```

```
                this.Subject = subj;
                }
                public void Display() {

                        System.out.println("\nId: "+empid);
                        System.out.println("Name: "+name);
                        System.out.println("Salary: "+salary);
                        System.out.println("Address: "+address);
                        System.out.println("Department:
"+Department);
                        System.out.println("Subject: "+Subject);

                }

        }
        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                int i,count, id_;
                float sal;
                String nam,adr,dep,sub;


                System.out.println("Enter the number of records to
be stored:");
                count = sc.nextInt();

                Teacher[] e = new Teacher[count];


                for( i=0; i<count; i++)
                        {

                          System.out.println("Enter the ID:");
                           id_= sc.nextInt();
                          System.out.println("Enter the name:");
                           nam= sc.next();
                          System.out.println("Enter the salary:");
                           sal= sc.nextFloat();
                          System.out.println("Enter the address:");
                           adr= sc.next();
                          System.out.println("Enter the
department:");
                           dep= sc.next();
                          System.out.println("Enter the subject:");
                           sub= sc.next();

                          e[i] = new
Teacher(id_,nam,sal,adr,dep,sub);

                        }
                System.out.println("----EMPLOYEE
DETAILS-----");
                for( i=0; i<count; i++)
            {
```

```
                    e[i].Display();
                }
            }

        }
```

RESULT

The above program is executed and obtains the output.

OUTPUT

```
Console
<terminated> Main [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (31-May-2021, 1:39:54 pm – 1:41:51 pm)
Enter department name
Chemistry
Enter Subject taught
Biochemistry
Enter employee id
453
Enter employee name
Gijo
Enter employee salary
56000
Enter employee address
Pulikkotti(H),Thsr
Enter department name
Economics
Enter Subject taught
Accounting

 LIST OF TEACHERS

Employee id :567
Employee name :treesa
Employee salary :87000
employee address :Vadakethala(H),PLKD
Department  :Chemistry
Subject taken:Biochemistry

 ................

Employee id :453
Employee name :Gijo
Employee salary :56000
employee address :Pulikkotti(H),Thsr
Department  :Economics
Subject taken:Accounting

 ................
```

PROGRAM -3

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 contains constructors and methods to display the data members. Use an 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 for the number of teachers provided
STEP 5:Stop

PROGRAM CODE

person.java	

```
package myproject;
import java.util.*;

public class person {
        String Name, Gender, Address;
        int Age;

        public person(String n, String g, String ad, int age){
                Name = n;
                Gender = g;
                Address = ad;
                Age = age;
        }
        static class Employee extends person{
            int Emp_id;
            String Co_name, Qualification;
            float Salary;

            public Employee(String n, String g, String ad, int age,
int e_id, String c_name, String qualif, float sal) {
                super(n, g, ad, age);

                Emp_id = e_id;
                Co_name = c_name;
                Qualification = qualif;
                Salary = sal;
            }
```

```
        static class Teacher extends Employee{
            int teach_id;
            String Subject, Department;

            public Teacher(String n, String g, String ad, int age, int
e_id, String c_name, String qualif, float sal, int t_id, String sub,
String dept) {
                super(n, g, ad, age, e_id, c_name, qualif, sal);

                teach_id = t_id;
                Subject = sub;
                Department = dept;
                }

            void Display() {
                    System.out.println("\nName: "+Name);
                    System.out.println("Gender: "+Gender);
                    System.out.println("Address: "+Address);
                    System.out.println("Age: "+Age);
                    System.out.println("Employee_Id
"+Emp_id);
                    System.out.println("Company Name
"+Co_name);
                    System.out.println("Qualification:
"+Qualification);
                    System.out.println("Salary: "+Salary);
                    System.out.println("Teacher_Id:
"+teach_id);
                    System.out.println("Subject: "+Subject);
                    System.out.println("Department:
"+Department);

            }

  }
        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                int N, age, e_id, t_id,i;
                float sal;
                String nam,gen,adr,cname,qual,dep,sub;

                System.out.println("no. of records to be stored");
    N = sc.nextInt();

    Teacher[] t = new Teacher[N];

    for( i=0; i<N; i++)
                {

                System.out.println("Enter the Name:");
    nam= sc.next();

    System.out.println("Enter the Gender:");
```

```
        gen= sc.next();

        System.out.println("Enter the address:");
        adr= sc.next();

        System.out.println("Enter the Age:");
        age= sc.nextInt();

        System.out.println("Enter the Employee ID:");
        e_id= sc.nextInt();

        System.out.println("Enter the Company Name:");
        cname= sc.next();

        System.out.println("Enter the Qualification:");
        qual= sc.next();

        System.out.println("Enter the Salary:");
        sal= sc.nextFloat();

        System.out.println("Enter the Teacher ID:");
        t_id= sc.nextInt();

        System.out.println("Enter the Department:");
        dep= sc.next();

        System.out.println("Enter the Subject:");
        sub= sc.next();


        t[i] = new
Teacher(nam,gen,adr,age,e_id,cname,qual,sal,t_id,sub,dep);
            }


     System.out.println("*****EMPLOYEE DETAILS*****");
                for( i=0; i<N; i++)
            {
                        t[i].Display();
                    }
   }

 }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT

```
<terminated> co3_q3 [Java Application] C:\Program Files\J
w13
enter age:
24
enter emp_id:
1313
enter company_name:
kml
enter qualification:
mtech
enter salary:
678900
enter subject:
ds
enter department:
mtech
enter Teacher_id:
1414
----------****DETAILS*****------------
Name: Arun
Gender: male
Address: q12
Age: 21
Employee id: 111
Company name: ghx
Qualification: nca
Salary: 234500.0
Subject: dbms
Department details: mca
Teacher id: 1212
----------****DETAILS*****------------
Name: kiran
Gender: male
Address: w13
Age: 24
Employee id: 1313
Company name: kml
Qualification: mtech
Salary: 678900.0
Subject: ds
Department details: mtech
Teacher id: 1414
```

PROGRAM -4

AIM: Write a program that 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 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 require
STEP 6: Stop

PROGRAM CODE

fiction.java	

```
package myproject;
import java.util.Scanner;
class publisher {
	String pname;
	Scanner sc=new Scanner(System.in);
	publisher()
	{
		System.out.println("Enter publisher name");
		pname=sc.next();
	}
}
class book extends publisher {
	String author;
	int cost;
	Scanner sc=new Scanner(System.in);
	book() {
		System.out.println("Enter author");
		author=sc.next();
		System.out.println("Enter cost");
		cost=sc.nextInt();
	}
}
class literature extends book {
	String edition;
	Scanner sc=new Scanner(System.in);
	literature() {
		System.out.println("Enter which edition");
		edition=sc.next();
	}
	void display()
	{
```

```
                System.out.println("******BOOK
DETAILS*******");
                System.out.println("Publisher "+pname);
                System.out.println("Author "+author);
                System.out.println("Cost"+cost);
        }
}
public class fiction extends book {
        String edition;
        Scanner sc=new Scanner(System.in);
        fiction() {
                System.out.println("Enter which edition");
                edition=sc.next();
        }
        void display()
        {
                System.out.println("Publisher "+pname);
                System.out.println("Author "+author);
                System.out.println("Cost "+cost);

        }
        public static void main(String[] args) {
                int li,fn,s;
                Scanner sc=new Scanner(System.in);
                System.out.println("Enter no. of  literature
books");
                li=sc.nextInt();
                literature l[]=new literature[li];
                for(int i=0;i<li;i++) {
                        l[i]=new literature();
                }
                System.out.println("Enter no. of fictions books");
                fn=sc.nextInt();
                book f[]=new book[fn];
                for(int i=0;i<fn;i++) {
                        f[i]=new book();
                }
                System.out.println("\"\\nSelect the
Genre(L/F):\\nL - Literature\\nF - Fiction");
                s=sc.nextInt();
                if(s==1) {
                        for(int i=0;i<fn;i++) {
                                l[i].display();
                        }
        }
                else
                        System.out.println("no ");
        sc.close();
        }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT

```
Enter the number of Literature records to be stored:
2

Record No:1
Enter the Book title:
Pride_and_Prejudice
Enter the Author:
Jane_Austen
Enter the Publisher Name:
Thomas_Egerton

Record No:2
Enter the Book title:
A_Bried_History_of_Time
Enter the Author:
Stephen_Hawkings
Enter the Publisher Name:
Bantam_Books
```

PROGRAM -5

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: Define a class which will read a student's academic information from the user.
STEP 3: Define another class that extends first class and reads the sports data of the student.
STEP 4: Define another class that extends second class And has a Display() to display the profile, academic score and sports score of the student.
STEP 5: Define a main () method to create objects for the above classes and to call the associated member methods.
STEP 6: Stop

PROGRAM CODE

marks.java	

```
package myproject;
import java.util.*;

class student
{
        String name ;
        int sid,eng,math ;
        public  student()
        {
                Scanner x = new Scanner(System.in);
                System.out.println("Enter Name : ");
                 name = x.next();
                 System.out.println("Enter student ID : ");
                 sid= x.nextInt();
                 System.out.println("Enter Marks in English :");
                 eng = x.nextInt();
                 System.out.println("Enter Marks in Maths :");
                 math = x.nextInt();

        }
}

class sports extends student
{
        String rank;

        public sports()
        {
                Scanner y = new Scanner(System.in);
```

```
                System.out.println("Enter rank");
                 rank = y.next();
        }
}

class result extends sports
{

        public void display()
        {
                System.out.println("Name : " + name);
                System.out.println("sid : " + sid);
                System.out.println("***ACADEMIC***");
                System.out.println("English :" + eng);
                System.out.println("Maths :" + math);
                System.out.println("***SPORTS***");
                System.out.println("Sports Rank : " + rank);
        }
}
public class marks {
        public static void main(String[] args)
        {
                result student = new result();
                student.display();


        }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT

```
Please input the student details. .

Enter the Roll No:
07
Enter the Name:
Evania_Krishna

Input the Subject marks. .

Enter marks for Subject-1:
89
Enter marks for Subject-2:
86
Enter marks for Subject-3:
82

Input the Sports data. .

Enter Event-1:
Long_Jump
Enter score:
7
Enter Event-2
100m_Race
Enter score:
9
```

```
__________REPORT CARD________

___STUDENT PROFILE___
Roll No: 7
Name: Evania_Krishna

___ACADEMIC SCORE___
Subject-1 Marks: 89.0
Subject-2 Marks: 86.0
Subject-3 Marks: 82.0
TOTAL:257.0

___SPORTS SCORE___
Event-1: Long_Jump
Score: 7
Event-2: 100m_Race
Score: 9
TOTAL:16
```

PROGRAM -6

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

ALGORITHM

STEP 1: Start
STEP 2: Define an interface with methods to read inputs and calculate area and perimeter.
STEP 3: Define a class that extends Circle to initialize its data members l, b and to calculate and display the area and perimeter of a rectangle.
STEP 4: Define a main () to create objects for the above classes to invoke its member methods to print the results
STEP 5: Stop

PROGRAM CODE

intface.java	package myproject;
	interface Shape
	{
	void input();
	void area();
	void perimeter();
	}
	class Circle implements Shape
	{
	int r = 0;
	double pi = 3.14, ar = 0,pr =0;
	public void input()
	{
	r = 5;
	}
	public void area()
	{
	ar = pi * r * r;
	System.out.println("Area of circle:"+ar);
	}
	public void perimeter()
	{
	pr=2*pi*r;
	System.out.println("Perimeter of circle:"+pr);
	}
	}
	class Rectangle extends Circle

```
{
   int l = 0, b = 0;
   double ar,pr;
   public void input()
   {
      super.input();
      l = 6;
      b = 4;
   }
   public void area()
   {
      super.area();
      ar = l * b;
      System.out.println("Area of rectangle:"+ar);
   }
   public void perimeter()
   {
      super.perimeter();
      ar = 2*l + 2*b;
      System.out.println("Perimeter of rectangle:"+pr);
   }
}
public class intface
{
public static void main(String[] args)
{
   Rectangle obj = new Rectangle();
   obj.input();
   obj.area();
   obj.perimeter();
}
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT

```
CHOICES         :
    1.Circle
    2.Rectangle
    3.Exit
Enter your choice  :
1
Enter the radius of circle
2
Area of the circle is         :  12.56
Perimeter of the circle is    :  12.56
CHOICES         :
    1.Circle
    2.Rectangle
    3.Exit
Enter your choice  :
2
Enter the length of rectangle
5
Enter the breadth of rectangle
2
Area of the rectangle is      :  10
Perimeter of the rectangle is :  14
CHOICES         :
    1.Circle
    2.Rectangle
    3.Exit
Enter your choice  :
3
exit
CHOICES         :
    1.Circle
    2.Rectangle
    3.Exit
Enter your choice  :
4
Wrong choice
```

PROGRAM -7

AIM: Prepare bill with the given format using the calculate method from the 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: Define an interface Calculation with a method total().
STEP 3: Define a class Order that implements Calculation to calculate the total amount for each product and has methods to generate a bill as given in the question.
STEP 4: Define a main () to create objects for the class.
STEP 5: Invoke the above methods by passing the data collected from the user to generate the required bill.
STEP 6: Stop

PROGRAM CODE

bill.java

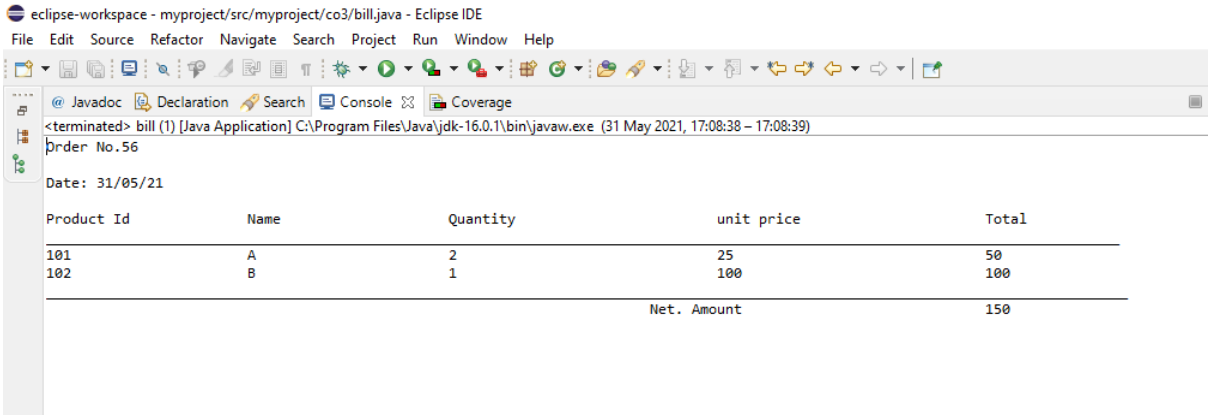
```
package myproject.co3;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

interface calc
{
	void total();
}
class  product1 implements calc
{
	int pid=101,qty=2,pr=25,total;
	String name="A";

	public void total()
	{
		total=qty*pr;
```

```
        }
}
class product2 extends product1 implements calc
{
        int p_id=102,qnty=1,_pr=100,totl;
        String nam="B";
        DateFormat df=new SimpleDateFormat("dd/MM/yy");
        Date d= new Date();
        public void total()
        {
                super.total();
                totl=qnty*_pr;
        }
        public void display()
        {
                System.out.println("Order No.56\n");
                System.out.println("Date: "+df.format(d));
                System.out.println("\nProduct
Id\at\tName\t\t\tQuantity\t\t\tunit price\t\t\tTotal");

System.out.println("_____
_____
_____");

System.out.println(pid+"\t\t\t"+name+"\t\t\t"+qty+"\t\t\t\t"+pr+"\t\t
\t\t"+total);

System.out.println(p_id+"\t\t\t"+nam+"\t\t\t"+qnty+"\t\t\t\t"+_pr+"\
t\t\t\t"+totl);

System.out.println("_____
_____
_____");
                System.out.println("\t\t\t\t\t\t\t\t\tNet.
Amount"+"\t\t\t\t"+(total+totl));

        }
}
public class bill
{

        public static void main(String[] args)
        {
                product1 p1=new product1();
                product2 p2=new product2();
                p1.total();
                p2.total();
                p2.display();
        }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



LAB CYCLE 4

PROGRAM -1

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: Create a package.
STEP 3: Define a class in the package representing geometric shapes like Triangle,

Square and Circle. It should contain methods to compute the area of each figure.
STEP 4: Compile and use this package to find areas of different shapes as chosen by the user.
STEP 5: Stop

PROGRAM CODE

shape.java	

```
package graphic;
import java.util.Scanner;
public interface shape{
        void input();
        void area();

}
class Circle implements shape
{
   int r ;
   double pi = 3.14, ar ;
   Scanner cl=new Scanner(System.in);
        public void input() {
                System.out.println("enter radius:");
                r=cl.nextInt();
        }
   public void area()
   {
      ar = pi * r * r;
      System.out.println("Area of circle:"+ar);
   }

}
class Rectangle extends Circle
{
   int x, y ;
   double ar;
   Scanner rl=new Scanner(System.in);
        public void input() {
                System.out.println("enter  length and breadth :");
                x=rl.nextInt();
                y=rl.nextInt();
        }
```

```
   public void area()
   {
      ar = x* y;
      System.out.println("Area of rectangle:"+ar);
   }

static class Triangle extends Rectangle
{
   int h , b ;
   double ar;
   Scanner tl=new Scanner(System.in);
         public void input() {
                  System.out.println("enter breadth and height :");
                  h=tl.nextInt();
                  b=tl.nextInt();
         }
   public void area()
   {
      ar = h * b*1/2;
      System.out.println("Area of triangle:"+ar);
   }
 static class Square
 {
         int s;
         double ar;
         Scanner sq=new Scanner(System.in);
         public void input() {
                        System.out.println("enter one side:");
                        s=sq.nextInt();
                  }

         public void area()
         {
                  ar=s*s;
                  System.out.println("Area of square:"+ar);
         }
 }
}
}
```

shape.java

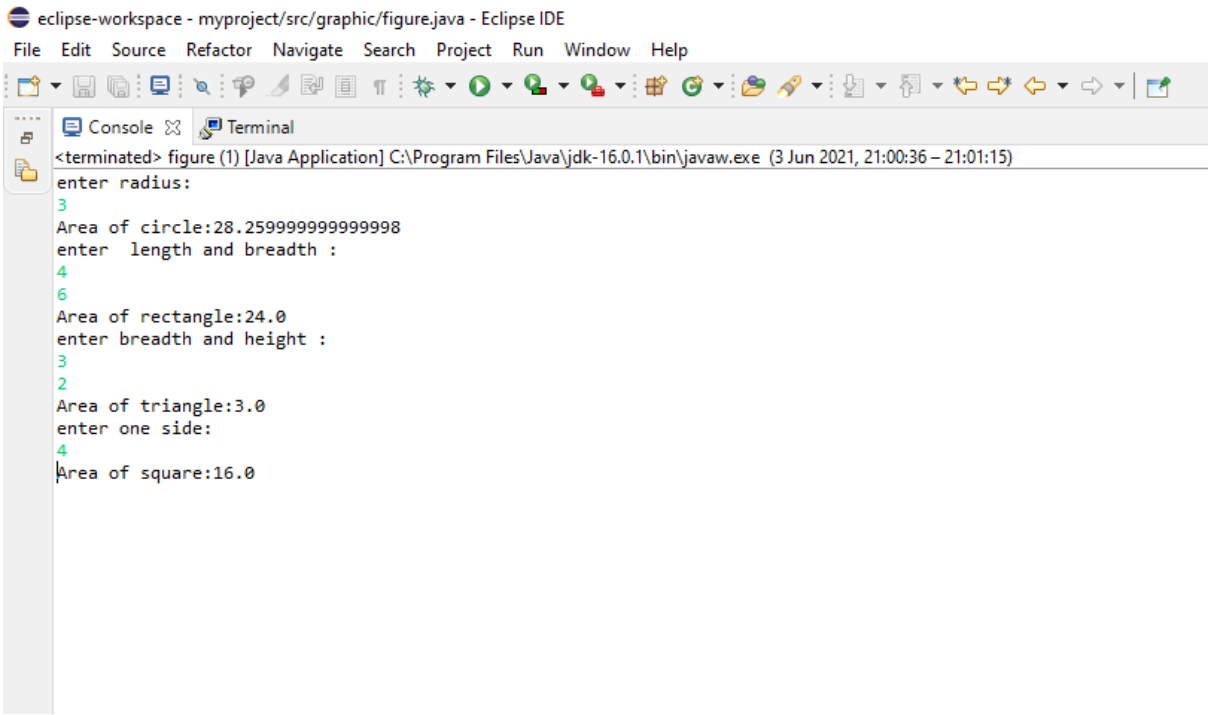
```
package graphic;
import graphic.Circle;
import graphic.Rectangle.Triangle;
import graphic.Rectangle.Triangle.Square;
public class figure {

                  public static void main(String[] args) {
                           Circle obj=new Circle();
```

```
                    obj.input();
                    obj.area();
                    Rectangle obj1=new Rectangle();
                    obj1.input();
                    obj1.area();
                Triangle obj2=new Triangle();
                obj2.input();
                    obj2.area();
                    Square obj3=new Square();
                    obj3.input();
                    obj3.area();

            }

        }
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -2

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: Create an arithmetic package that has classes and interfaces for the 4 basic arithmetic operations.

STEP 3: Define an interface in the package containing methods such as add(), sub(), div() and mul() and a class to implement these methods.

STEP 4: Compile and use this package to perform those methods from a driver class outside the package.

STEP 5: Stop

PROGRAM CODE

operation.java	package arithmetic;
	import java.util.Scanner;
	interface math{
	void input();
	void add();
	void sub();
	void div();
	void mult();
	}
	public class operation implements math {
	double a,b,ad,sb,ml,di;
	Scanner sc= new Scanner(System.in);
	public void input() {
	System.out.println("Enter first number:");
	a=sc.nextInt();
	System.out.println("Enter second number:");
	b=sc.nextInt();
	
	}
	public void add() {
	ad=a+b;
	System.out.println("Addition :"+ad);
	}
	public void sub() {
	sb=a-b;
	System.out.println("Subtraction:"+sb);
	}
	public void mult() {
	ml=a*b;
	System.out.println("Multiplication:"+ml);

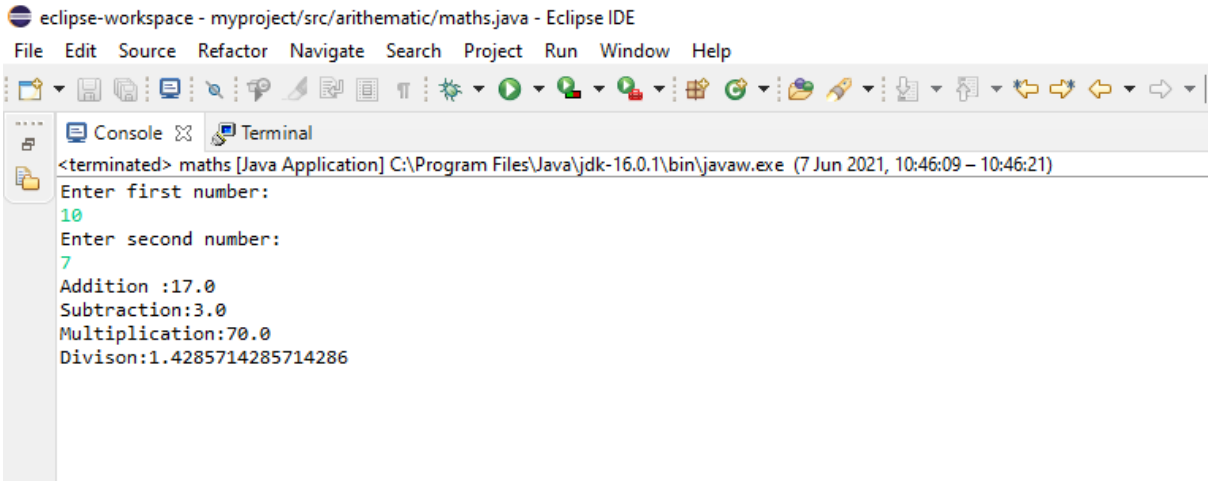
	}
	public void div() {
	di=a/b;
	System.out.println("Division:"+di);
	}
	}

maths.java	package arithmetic;
	import arithmetic.operation;
	public class maths {
	
	public static void main(String[] args) {
	operation obj=new operation();
	obj.input();
	obj.add();
	obj.sub();
	obj.mult();
	obj.div();
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -3

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

ALGORITHM

STEP 1: Start
STEP 2: Create a User-defined exception named check_username.
STEP 3: Check for the validity of the username against the required parameters.
If any parameter is not met, then throw the username failed exception.
STEP 4: Create another user-defined exception named check_password.
STEP 5: Check for the validity of the password against the required parameters.
If any parameter is not met, then throw the password failed exception.
STEP 6: To authenticate the credentials provided, check if the passwords and usernames given and entered are matching. If they match, display login successful, otherwise throw an exception
STEP 7:Stop.

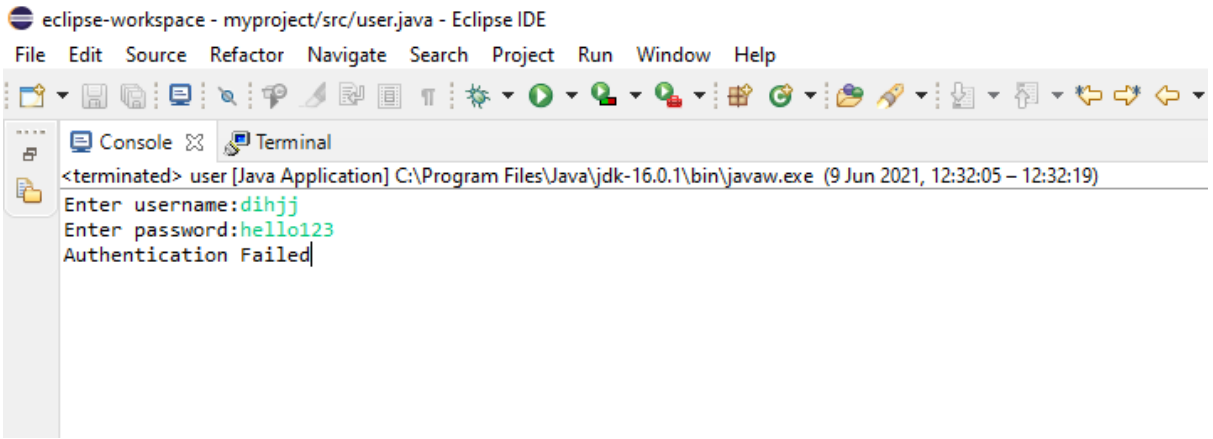
PROGRAM CODE

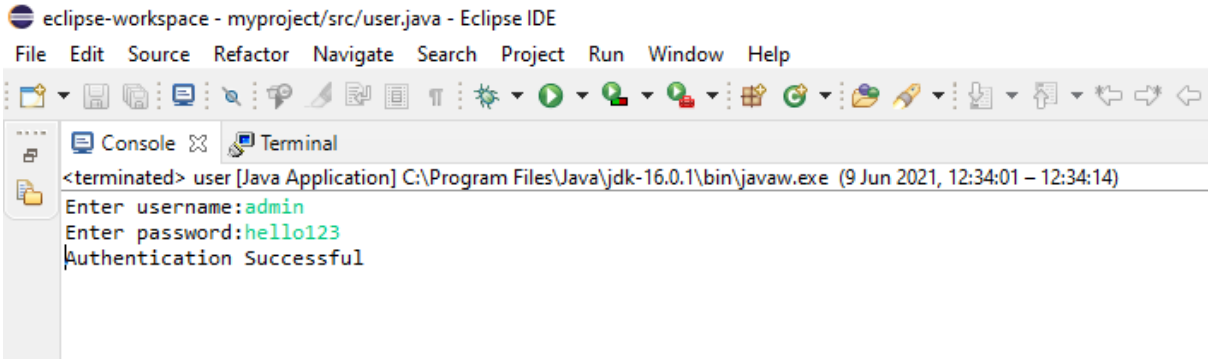
user.java	import java.util.Scanner;
	public class user
	{
	public static void main(String args[])
	{
	String username, password;
	Scanner sc = new Scanner(System.in);
	System.out.print("Enter username:");
	username = sc.nextLine();
	System.out.print("Enter password:");
	password = sc.nextLine();
	try
	{
	if(username.equals("admin") &&
	password.equals("hello123"))
	{
	System.out.println("Authentication Successful");
	}
	else
	{
	System.out.println("Authentication Failed");
	}
	}
	catch (Exception e){
	System.out.println(e.getMessage());
	
	}
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT





PROGRAM -4

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class which throws a negative number of exceptions.
STEP 3: Get user inputs for integers at run time and throw an exception if
The entered number is negative.
STEP 4: Else, proceed to calculate the average and display the result.
STEP 5: Stop

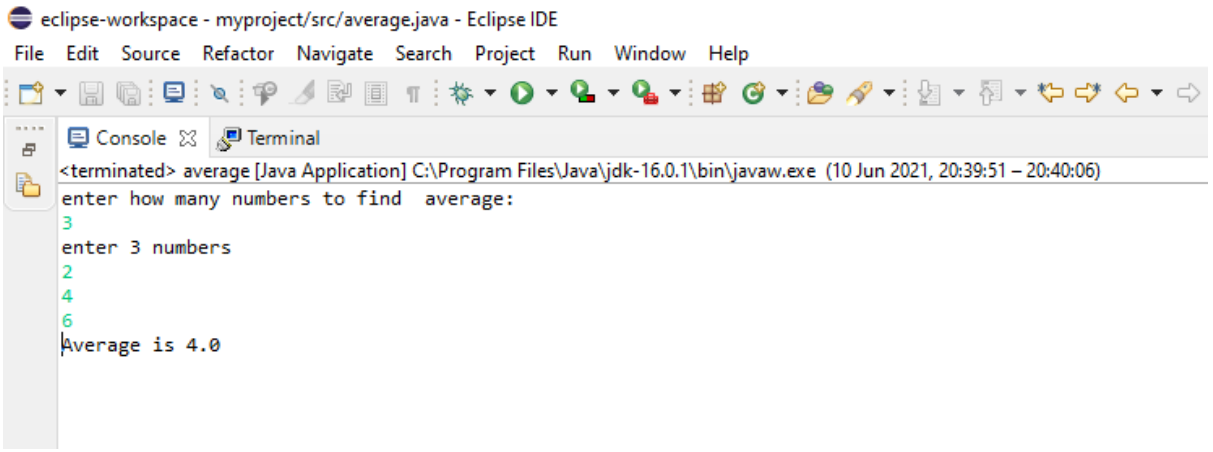
PROGRAM CODE

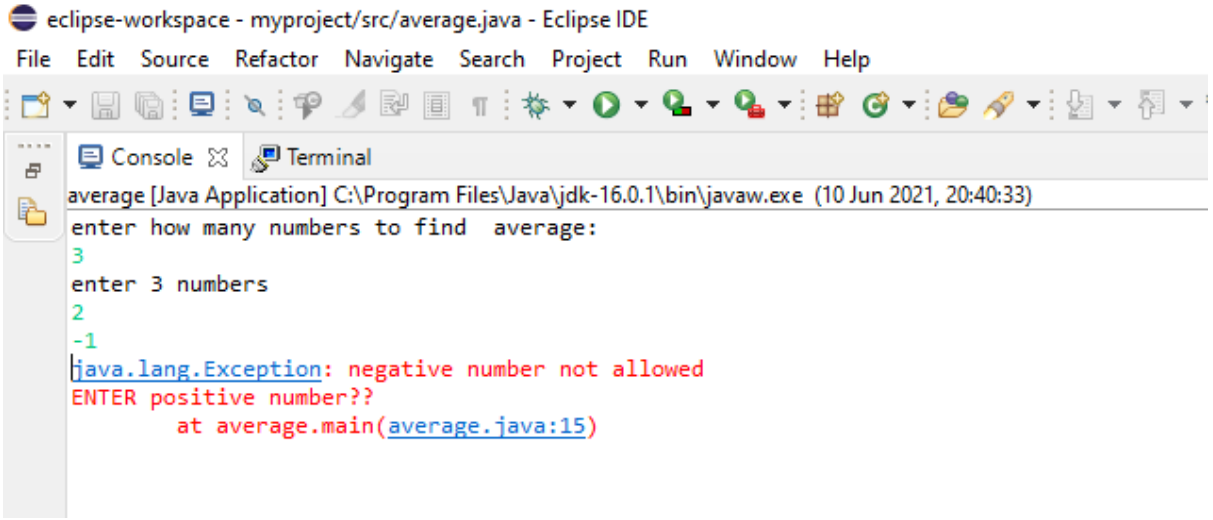
average.java	import java.util.Scanner;
	public class average {
	public static void main(String args[]) {
	int n;
	double res=0;
	Scanner sc =new Scanner(System.in);
	System.out.println("enter how many numbers to find average:");
	n=sc.nextInt();
	int a[]=new int[n];
	System.out.println("enter "+n+" numbers");
	for(int i=0;i<n;i++) {
	a[i]=sc.nextInt();
	try {
	if (a[i]<0) {
	throw new Exception("negative number not allowed\nENTER positive number??");
	}
	
	else {
	res+=a[i];
	}
	}
	catch(Exception N){
	N.printStackTrace();
	Scanner sc1 = new Scanner(System.in);
	res=sc1.nextInt();
	res+=a[i];
	}
	}
	
	double avg = res/n;
	System.out.println("Average is "+avg);
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT





PROGRAM -5

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

ALGORITHM

STEP 1: Start
STEP 2: Define classes that extend Thread class and contain methods to compute the multiplication table of 5 and to generate first N prime numbers respectively..
STEP 3:Define a main method to create objects for the classes and invoke the associated methods
STEP 4: Stop

PROGRAM CODE

thread_class.java	

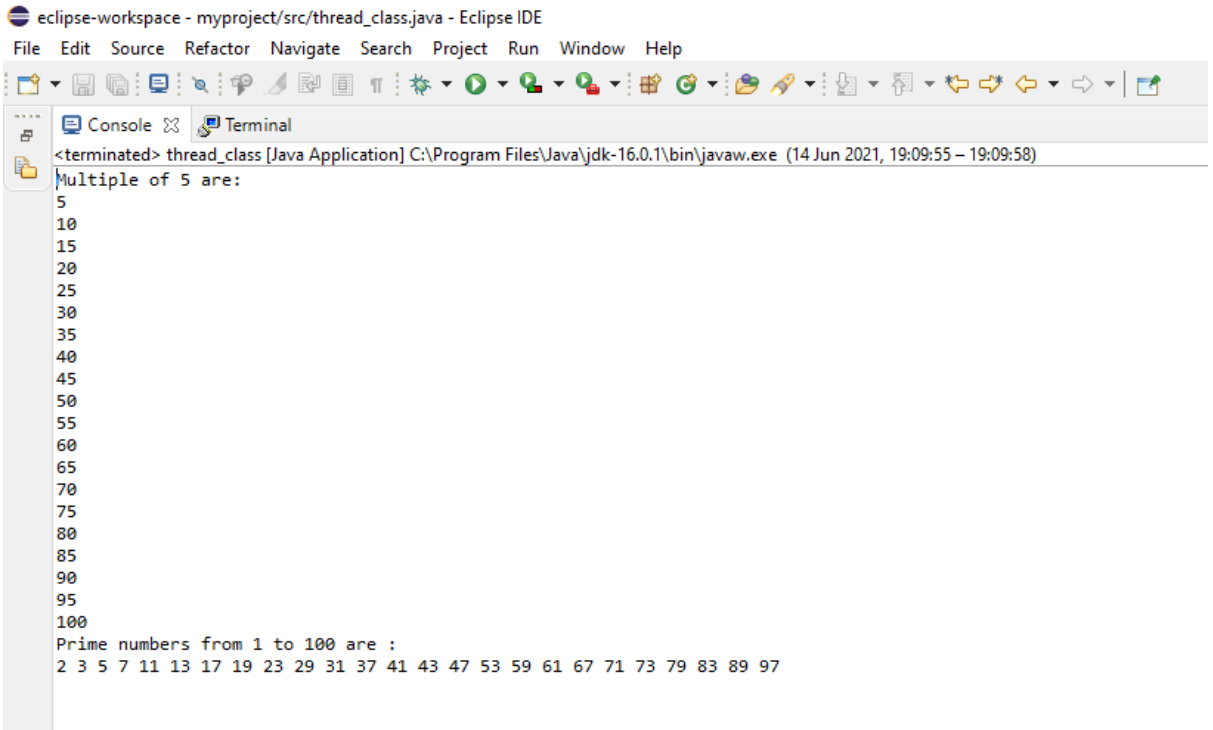
```
import java.io.*;

 class ThreadA  extends Thread
{
        public void run(){
                System.out.println("Multiple of 5 are: ");
                for(int k=1; k<=100; k++){
                        if(k%5==0){

                                System.out.println(k+"");
                                }
                        }
}
}
class ThreadB extends Thread
{
        public void run() {
                 int i =1,j=1,num=0;
            String  primeNumbers = "";{

            for (i = 1; i <= 100; i++)
            {
              int counter=0;
              for(num =i; num>=1; num--)
                  {
                if(i%num==0)
                    {
                        counter = counter + 1;
                    }
                  }
                  if (counter ==2)
                  {
                    primeNumbers = primeNumbers + i + " ";
```

```
                    }
                }
                System.out.println("Prime numbers from 1 to 100 are
:");
                System.out.println(primeNumbers);
                }
            }
}
public class thread_class {

            public static void main(String[] args) throws
InterruptedException  {
                    ThreadA t1=new ThreadA();
                    ThreadB t2=new ThreadB();
                    t1.start();
                    ThreadA.sleep(1000);
                    t2.start();
                    ThreadB.sleep(1000);

            }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/thread_class.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Terminal
<terminated> thread_class [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (14 Jun 2021, 19:09:55 – 19:09:58)
Multiple of 5 are:
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
Prime numbers from 1 to 100 are :
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

PROGRAM -6

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: Define classes that implements *Runnable interface* and contain methods to generate Fibonacci series and to display even numbers in a given range respectively.
STEP 3:Define a main method to create objects for the classes and invoke the associated methods.
STEP 4: Stop

PROGRAM CODE

runnable_thread.java	

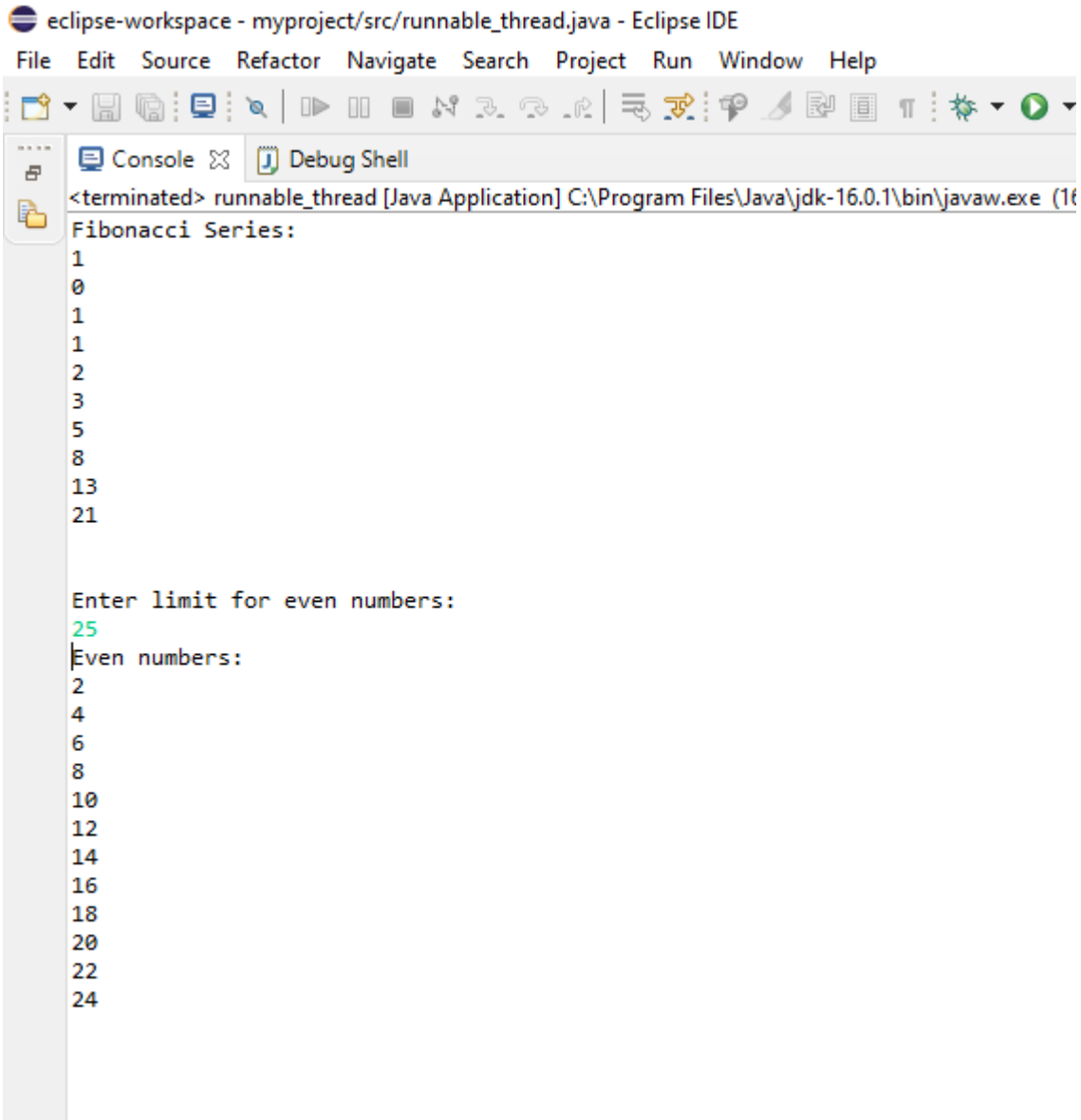
```
import java.util.Scanner;
import java.io.*;
import java.lang.*;
class fibonci implements Runnable{
        public void run() {
        int n=10, a = 0, b = 1, c = 0;

   System.out.println("Fibonacci Series:");
   for(int i = 1; i <= n; i++)
   {
      a = b;
      b = c;
      c = a + b;
      System.out.println(a+" ");

   }
}
}
class even implements Runnable{
        public void run() {

        int N;
        Scanner sc = new Scanner(System.in);
        System.out.println("\n");
   System.out.println("Enter limit for even numbers:");
   N = sc.nextInt();
        System.out.println("Even numbers:");
        for (int i=1; i<=N; i++)
        {
        if (i%2==0)
        {
        System.out.println(i + " ");
```

```
        }
        }
        }
}
public class runnable_thread {
        public static void main(String args[]) throws
InterruptedException  {
                fibonci t1= new fibonci();
                Thread p= new Thread(t1);
                even t2= new even();
                Thread q= new Thread(t2);
                p.start();
                Thread.sleep(200);
                q.start();
                Thread.sleep(500);
        }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/runnable_thread.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Debug Shell
<terminated> runnable_thread [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (16
Fibonacci Series:
1
0
1
1
2
3
5
8
13
21
Enter limit for even numbers:
25
Even numbers:
2
4
6
8
10
12
14
16
18
20
22
24

PROGRAM -7

AIM: Producer/Consumer using ITC

ALGORITHM

STEP 1: Start
STEP 2: Define a class that contains two threads that will simulate producer and consumer.
STEP 3: Define a class which will contain a LinkedList of integers.
STEP 4: Define a method produce() that produces items till its capacity is reached and notify the consumer thread to start consuming.
STEP 5: Define a method consume() that consumes items till the list is empty and notify the producer thread to start producing.
STEP 6: Stop

PROGRAM CODE

inter_thread.java	

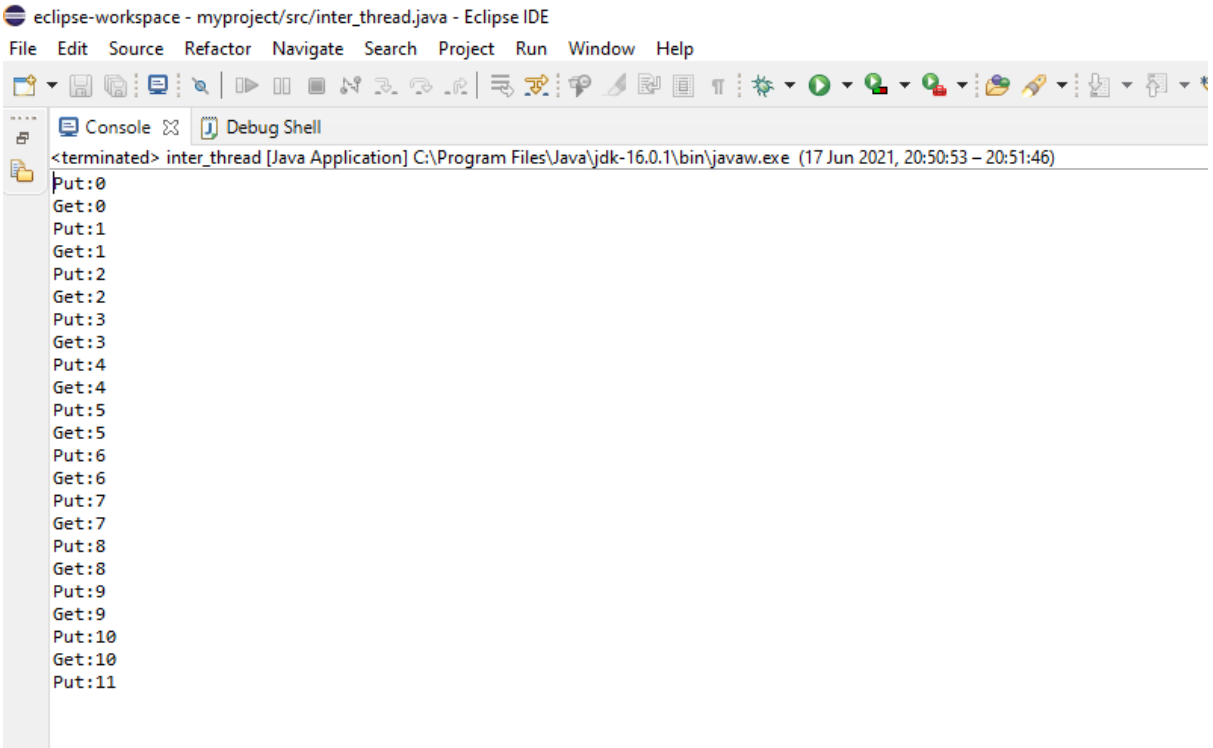
```
class Q
{
	int num;
	boolean valueSet =false;
	public synchronized void put(int num)
	{
		while(valueSet) {
			try{wait();} catch(Exception e) {}
		}
		System.out.println("Put:"+num);
		this.num=num;
		valueSet=true;
		notify();
	}
	public synchronized void get()
	{
		while(!valueSet) {
			try{wait();} catch(Exception e) {}
		}
		System.out.println("Get:"+num);
		valueSet=false;
		notify();
	}
}
class producer implements Runnable{
	Q q;

	public producer(Q q) {
		this.q = q;
```

```
                Thread t=new Thread(this,"producer");
                t.start();
        }
        public void run()
        {
                int i=0;
                while(true)
                {
                        q.put(i++);
                        try {Thread.sleep(500);} catch(Exception
e) {}
                }
        }

}
class consumer implements Runnable{
        Q q;

        public consumer(Q q) {
                this.q = q;
                Thread t=new Thread(this,"consumer");
                t.start();
        }
        public void run()
        {
                while(true)
                {
                        q.get();
                        try {Thread.sleep(5000);} catch(Exception
e) {}
                }
        }

}
public class inter_thread {
        public static void main(String[] args) {
                Q q =new Q();
                new producer(q);
                new consumer(q);
        }

}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -8

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class that contains methods to push, pop and display the stack elements.
STEP 3: Define another class to create objects for the above class and invoke the respective methods.
STEP 4: Display the results.
STEP 5: Stop

PROGRAM CODE

stack.java	

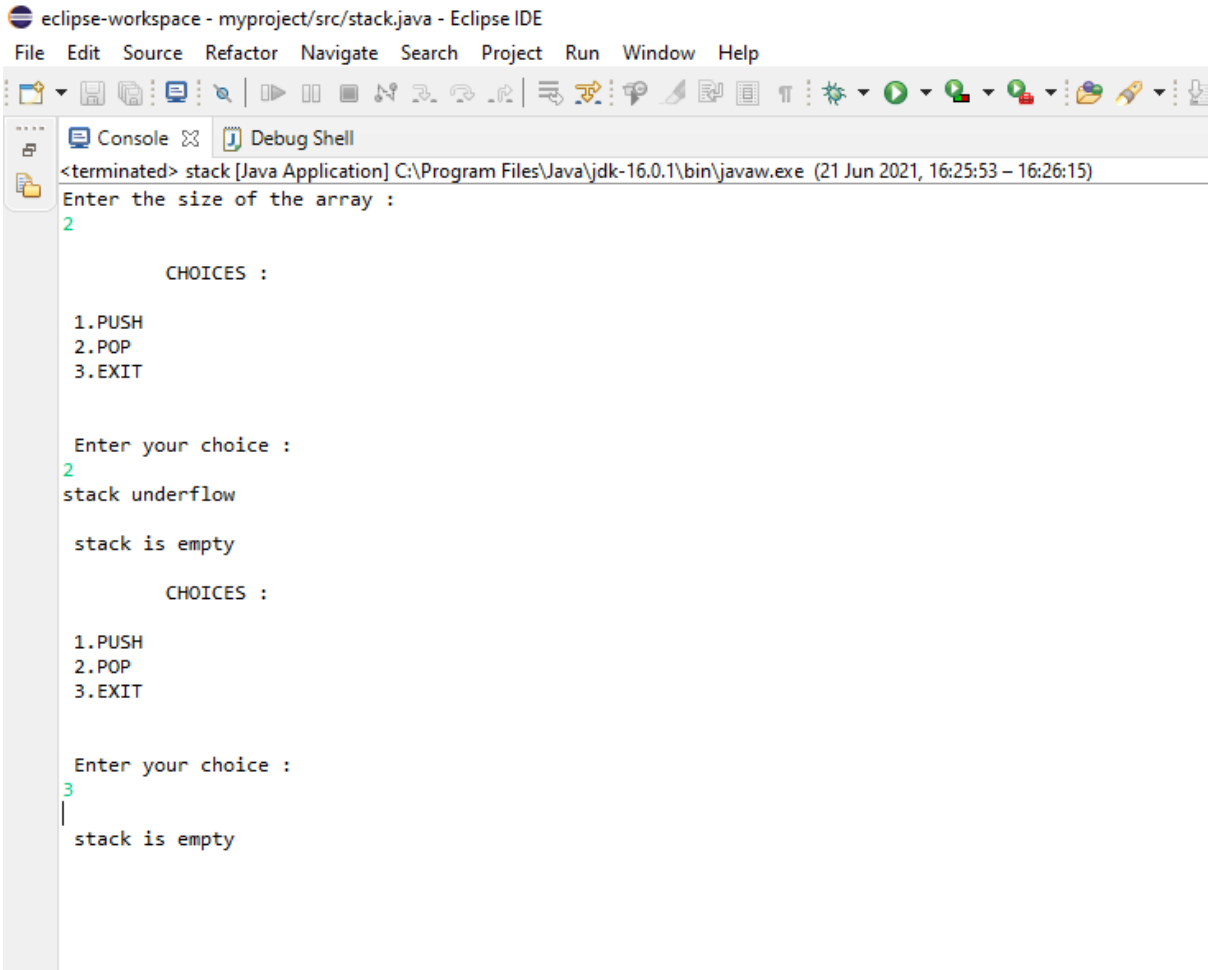
```
import java.util.Scanner;
 class StackArr{
int a[] = new int[10];
int top=-1,ch,item,i;
Scanner sc = new Scanner(System.in);
public void stackoperation()
{
        System.out.println("Enter the size of the array : ");
        int n=sc.nextInt();
do
{
System.out.println("\n\t CHOICES : ");
System.out.println("\n 1.PUSH \n 2.POP \n 3.EXIT \n");
System.out.println("\n Enter your choice : ");
ch=sc.nextInt();
switch(ch)
{
case 1: if(top >=n-1)
      {
                System.out.println("stack overflow");
      }
      else
      {
      System.out.println("enter the element :");
      item =sc.nextInt();
      top=top+1;
      a[top]=item;
      }
      break;
case 2 : if(top<0)
       {
                System.out.println("stack underflow");
       }
       else
```

```
        {
        a[top]='\0';
        top=top-1;
        }
        break;
case 3 : break;
default : System.out.println("\n Invalid choice");
}
if(top < 0)
{
System.out.println("\n stack is empty");
}
else
{
System.out.println("\n stack is \n");
for(i=top;i>=0;i--)
{
System.out.println(a[i]);
}
}
}
while(ch!=3);
}
}
public class stack
{

public static void main(String[] args)
{
        StackArr sa =new StackArr();
        sa.stackoperation();
}
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/stack.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Debug Shell
<terminated> stack [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (21 Jun 2021, 16:25:53 – 16:26:15)
Enter the size of the array :
2
CHOICES :
1.PUSH
2.POP
3.EXIT
Enter your choice :
2
stack underflow
stack is empty
CHOICES :
1.PUSH
2.POP
3.EXIT
Enter your choice :
3
stack is empty

PROGRAM -9

AIM: Using generic methods perform Bubble sort.

ALGORITHM

STEP 1: Start
STEP 2: Define a class with a main().
STEP 3: Get user input num for the number of integers to be sorted.
STEP 4: Read the numbers from the user and store them into int array[].
STEP 5: for (int i = 0; i < (num - 1); i++) {
for (int j = 0; j < num - i - 1; j++) {
if (array[j] > array[j+1]) {
int temp = array[j];
array[j] = array[j+1];
array[j+1] = temp;
} } }
STEP 6: Display the sorted list of integers
STEP 7:Stop.

PROGRAM CODE

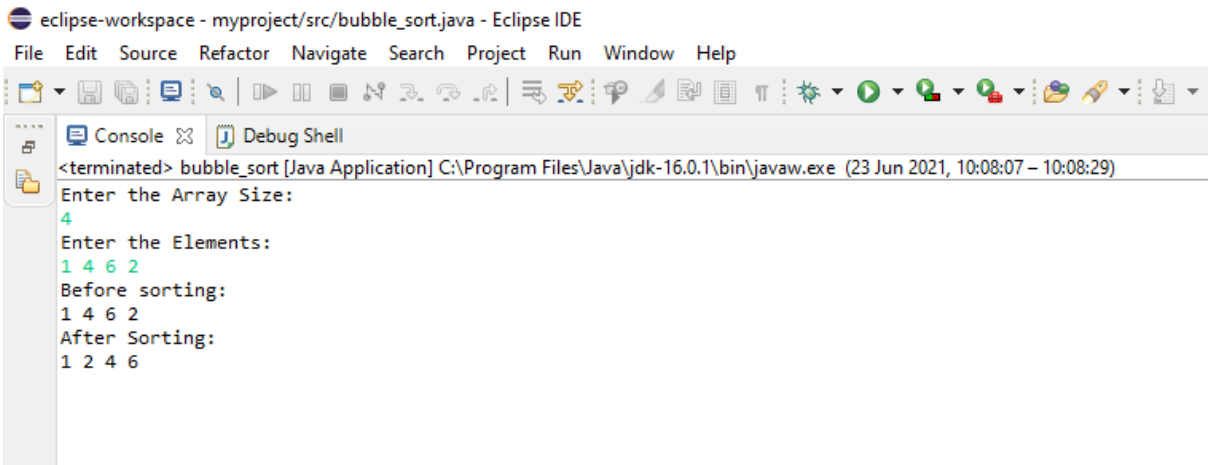
bubble_sort.java	import java.util.*;
	class BubbleSort {
	void sort(int arr[],int n) {
	int temp;
	for(int i=0;i<arr.length-1;i++) {
	for(int j=0;j<arr.length-1-i;j++) {
	if(arr[j]>arr[j+1]) {
	temp=arr[j];
	arr[j]=arr[j+1];
	arr[j+1]=temp;
	}
	}
	}
	
	}
	void display(int arr[],int n) {
	for (int i=0;i<n;i++) {
	System.out.print(arr[i]+" ");
	}
	
	}
	}
	public class bubble_sort{
	public static void main(String[] args) {
	int n,i;
	Scanner sc=new Scanner(System.in);

```
System.out.println("Enter the Array Size:");
n=sc.nextInt();
System.out.println("Enter the Elements:");
int arr[] = new int[n];
for(i=0;i<n;i++)
{
        arr[i] = sc.nextInt();
}
BubbleSort  obj= new BubbleSort  ();
System.out.println("Before sorting:");
 obj.display(arr,n);

 obj.sort(arr,n);

System.out.println("\nAfter Sorting:");
 obj.display(arr,n);
    }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -10

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class 'arrayList' with a main () to implement the concept.
STEP 3: Declare an ArrayList of strings named 'numbers' and start adding elements into it.
STEP 4: Execute different ArrayList methods and display the results.
STEP 5: Stop

PROGRAM CODE

Array_list.java	//Maintain a list of Strings using ArrayList from collection framework, perform built-in operations /* 1) add(Object o): This method adds an object o to the Array List. 2) add(int index, Object o): It adds the object o to the array list at the given index. 3) remove(Object o): Removes the object o from the ArrayList. 4) remove(int index): Removes element from a given index. 5) set(int index, Object o): Used for updating an element. 6) int indexOf(Object o): Gives the index of the object o. 7) Object get(int index): It returns the object of the list which is present at the specified index. 8) int size(): It gives the size of the ArrayList – Number of elements of the list. 9) boolean contains(Object o): It checks whether the given object o is present in the array list if it's there then it returns true else it returns false. 10) clear(): It is used for removing all the elements of the array list in one go. */ import java.util.*;

```
public class Array_List {
        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("One");
            obj.add("Three");
            obj.add("Four");
            obj.add("Five");
            obj.add(1, "Two");

            // Displaying elements

            System.out.println("\n ArrayList after add operation:");
            for(String str:obj)
              System.out.println(str);

            //Remove elements from ArrayList

            obj.remove("Five");
            obj.remove(3);

            // Displaying elements

            System.out.println("\n ArrayList after remove
operation:");
            for(String str:obj)
              System.out.println(str);

            // Displaying final Array List

            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

            System.out.println("\n Object at index 2:"+obj.get(2));
```

```
                // Checks whether the object is in the ArrayList

                System.out.println("\n Six is in the ArrayList
:"+obj.contains("Six"));
                System.out.println("\n Two is in the ArrayList
:"+obj.contains("Two"));

                //Size of the ArrayList

                System.out.println("\n Size of the
ArrayList:"+obj.size());

                // removing all the elements of the array list

                obj.clear();

                System.out.println("\n  ArrayList after clear
method:"+obj);

            }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT

eclipse-workspace - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Console Debug Shell

<terminated> Array_List [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (24 Jun 2021, 09:05:00 – 09:05:01)

```
 ArrayList after add operation:
One
Two
Three
Four
Five

 ArrayList after remove operation:
One
Two
Three

 Final ArrayList:
One
Two
Three

  ArrayList after sorting:
One
Three
Two

 Object at index 2:Two

 Six is in the ArrayList :false

 Two is in the ArrayList :true

 Size of the ArrayList:3

  ArrayList after clear method:[]
```

PROGRAM -11

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

ALGORITHM

STEP1: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 the linked list using method clear().
STEP 5:Display linked list.
STEP 6:Stop.

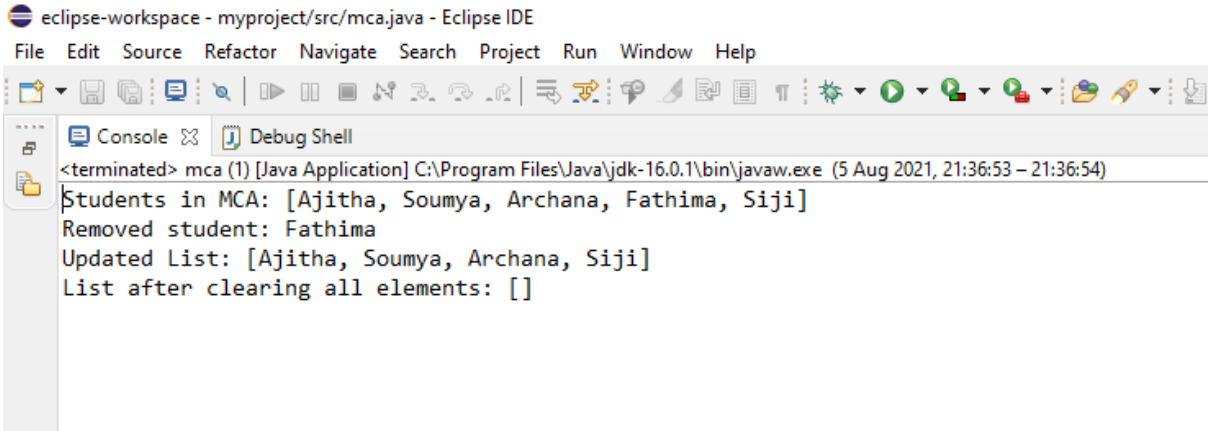
PROGRAM CODE

mca.java	import java.util.*;
	public class mca {
	public static void main(String[] args) {
	LinkedList<String> student = new LinkedList<>();
	
	// add elements in LinkedList
	
	student.add("Ajitha");
	student.add("Soumya");
	student.add("Archana");
	student.add("Fathima");
	student.add("Siji");
	System.out.println("Students in MCA: " + student);
	
	// removing an elements
	
	String str = student.remove(3);
	System.out.println("Removed student: " + str);
	
	//displaying the list
	
	System.out.println("Updated List: " + student);
	
	// clearing the list
	
	student.clear();
	
	// removing all elements from the linked list
	
	System.out.println("List after clearing all elements: " + student);
	
	}
	
	}

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -12

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class 'stack' with a main () to implement the concept.
STEP 3: Declare a Stack of integers and start adding elements into it using add() method.
STEP 4: Get user input for the index from which an element is to be removed.
STEP 5: Using remove(index) method, remove the element and display the updated stack.
STEP 6: Stop

PROGRAM CODE

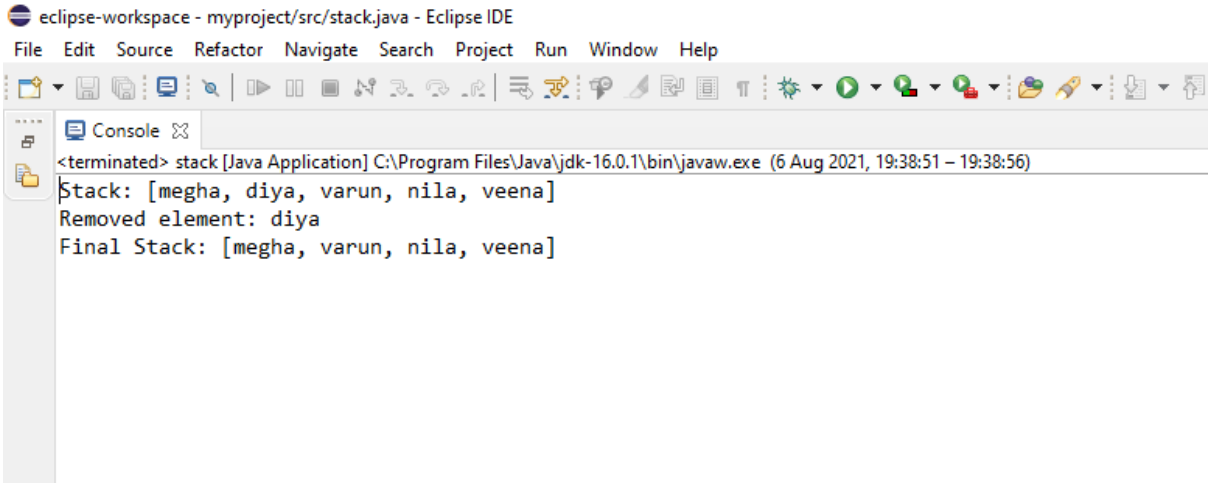
stack.java	

```
import java.util.*;
public class stack{
	public static void main(String args[])
	{
		// Creating an empty Stack
		Stack<String> stack = new Stack<String>();

		// Use add() method to add elements in the Stack
		stack.add("megha");
		stack.add("diya");
		stack.add("varun");
		stack.add("nila");
		stack.add("veena");

		// Output the Stack
		System.out.println("Stack: " + stack);

		// Remove the element using remove()
		String rem = stack.remove(1);

		// Print the removed element
		System.out.println("Removed element: "+ rem);

		// Print the final Stack
		System.out.println("Final Stack: "+ stack);
	}
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -13

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class with a main () to implement the concept.
STEP 3: Declare a PriorityQueue of integers and start adding elements into it using add() method.
STEP 4: Iterate and display the queue elements.
STEP 5: Stop

PROGRAM CODE

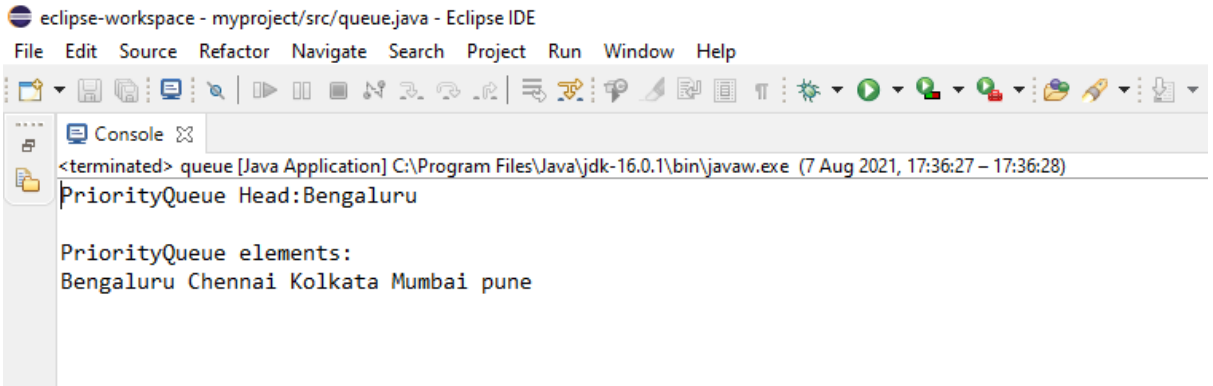
queue.java	

```
import java.util.*;
class queue{
   public static void main(String args[]){
      PriorityQueue<String> city=new PriorityQueue<String>();
      //initialize the PriorityQueue with values
      city.add("Mumbai");
      city.add("Bengaluru");
      city.add("Kolkata");
      city.add("Chennai");
      city.add("pune");
      //print the head of the PriorityQueue
      System.out.println("PriorityQueue Head:"+city.element());
      //Define the iterator for PriorityQueue and print its elements
      System.out.println("\nPriorityQueue elements:");
      Iterator iter=city.iterator();
      while(iter.hasNext()){
         System.out.print(iter.next() + " ");
      }
   }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/queue.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
<terminated> queue [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (7 Aug 2021, 17:36:27 – 17:36:28)
PriorityQueue Head:Bengaluru
PriorityQueue elements:
Bengaluru Chennai Kolkata Mumbai pune

PROGRAM -14

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

ALGORITHM

STEP1: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.
STEP 6:Stop.

PROGRAM CODE

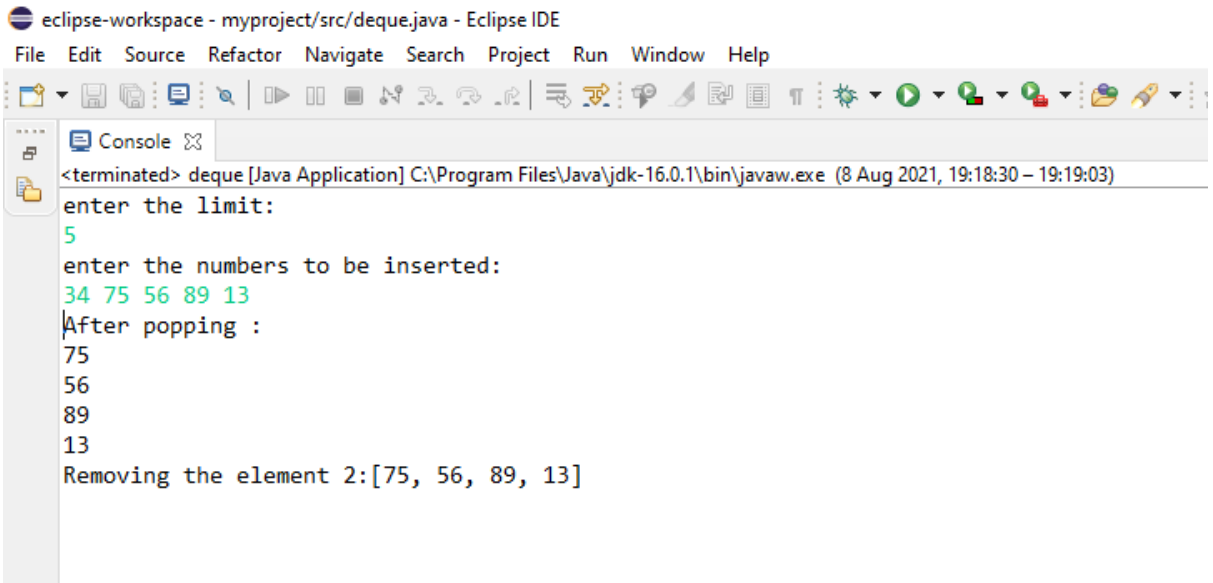
deque.java	

```
import java.util.*;
 public class deque {
 public static void main(String[] args) {
   Deque<Integer> deque = new ArrayDeque<Integer>();
   // Inserts the elements
   Scanner sc=new Scanner(System.in);
          System.out.println("enter the limit:");
          int n=sc.nextInt();
          System.out.println("enter the numbers to be inserted: ");
          for(int i =0;i<n;i++)
          {
                    Integer obj=sc.nextInt();
                    deque.add(obj);

          }

   // Popping the element
   deque.pop();
   System.out.println("After popping : ");
   for (Integer integer : deque) {
      System.out.println(integer);
   }
   deque.remove(2);
   System.out.println("Removing the element 2:"+deque);
  }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -15

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class with a main () to implement the concept.
STEP 3: Declare an LinkedHashSet of strings.
STEP 4: Start adding elements into it using add() method.
STEP 5: Execute some LinkedHashSet methods and display the results.
STEP 6: Stop

PROGRAM CODE

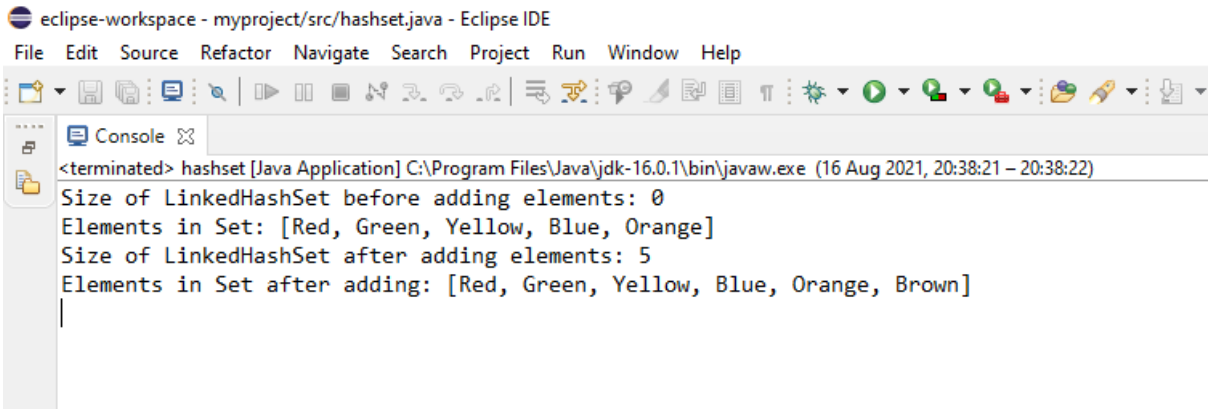
hashset.java	import java.util.*;
	public class hashset
	{
	public static void main(String[] args)
	{
	// Creating a Linked hash set
	LinkedHashSet<String> hset1= new LinkedHashSet<String>();
	
	// Checking the size of LinkedHashSet
	int size1 = hset1.size();
	System.out.println("Size of LinkedHashSet before adding elements: " +size1);
	
	// Adding elements in the linked hash set
	hset1.add("Red");
	hset1.add("Green");
	hset1.add("Yellow");
	hset1.add("Blue");
	hset1.add("Orange");
	
	System.out.println("Elements in Set: " +hset1);
	int size2 = hset1.size();
	System.out.println("Size of LinkedHashSet after adding elements: " +size2);
	
	// Adding duplicate elements
	hset1.add("Red");
	hset1.add("Yellow");
	
	// Create another set of String type.
	LinkedHashSet<String> hset2 = new LinkedHashSet<String>();
	hset2.add("Brown");
	
	// Adding elements of set2 into set.

```
  hset1.addAll(hset2);
  System.out.println("Elements in Set after adding: " +hset1);
 }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -16

AIM: Write a Java program to compare two hash set

ALGORITHM

STEP 1: Start
STEP 2: : Define a class with a main () to implement the concept. STEP 3: Declare two HashSets (s1, s2) of strings and start adding elements into them
using the add() method.
STEP 4: Display the elements of both the hashsets.
STEP 5: Perform set operations.
s1Unions2.addAll(s2) – gets all the elements in s1 and s2
s1Intersections2.retainAll(s2) – gets the elements which are common in s1 and s2
s1Differences2.removeAll(s2) – gets the difference between s1 and s2
STEP 6:Display the results.
STEP 7:Stop.

PROGRAM CODE

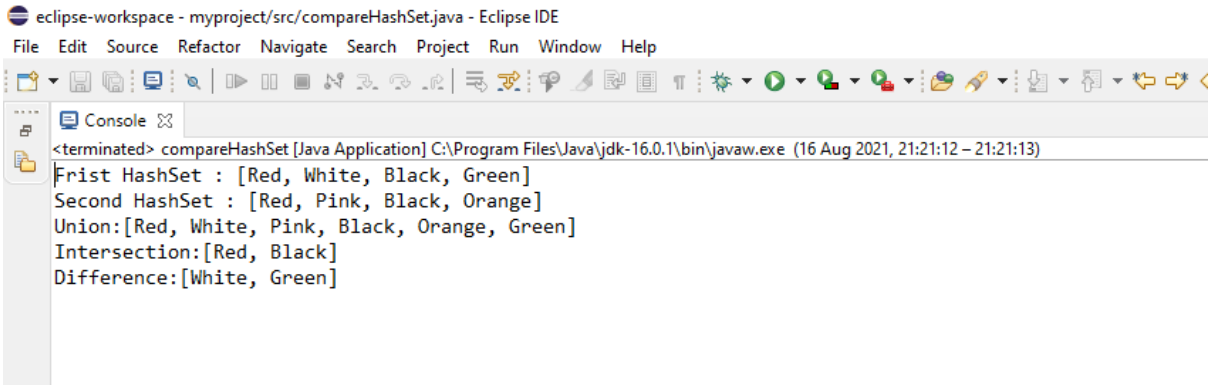
compareHashSet.java	import java.util.*;
	public class compareHashSet {
	public static void main(String[] args) {
	// Create a empty hash set
	HashSet<String> h_set1 = new HashSet<String>();
	// use add() method to add values
	h_set1.add("Red");
	h_set1.add("Green");
	h_set1.add("Black");
	h_set1.add("White");
	System.out.println("Frist HashSet : "+h_set1);
	HashSet<String>h_set2 = new HashSet<String>();
	h_set2.add("Red");
	h_set2.add("Pink");
	h_set2.add("Black");
	h_set2.add("Orange");
	System.out.println("Second HashSet : "+h_set2);
	
	// To find union
	Set<String> union = new HashSet<String>(h_set1);
	union.addAll(h_set2);
	System.out.println("Union:"+union);
	
	// To find intersection
	Set<String> intersection = new HashSet<String>(h_set1);
	intersection.retainAll(h_set2);
	System.out.println("Intersection:"+intersection);
	
	// To find the symmetric difference

```
        Set<String> difference = new HashSet<String>(h_set1);
        difference.removeAll(h_set2);
        System.out.println("Difference:"+difference);

    }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -17

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

ALGORITHM

STEP 1: Start
STEP 2: Define a class with a main () to implement the concept.
STEP 3: Declare a HashMap and insert elements into it using the put() method.
STEP 4: To update any value in hashmap using hashmap.put() or hashmap.replace()
STEP 5: Using remove(key) method, remove elements from hashmap.
STEP 6: Display the results.
STEP 7: Stop.

PROGRAM CODE

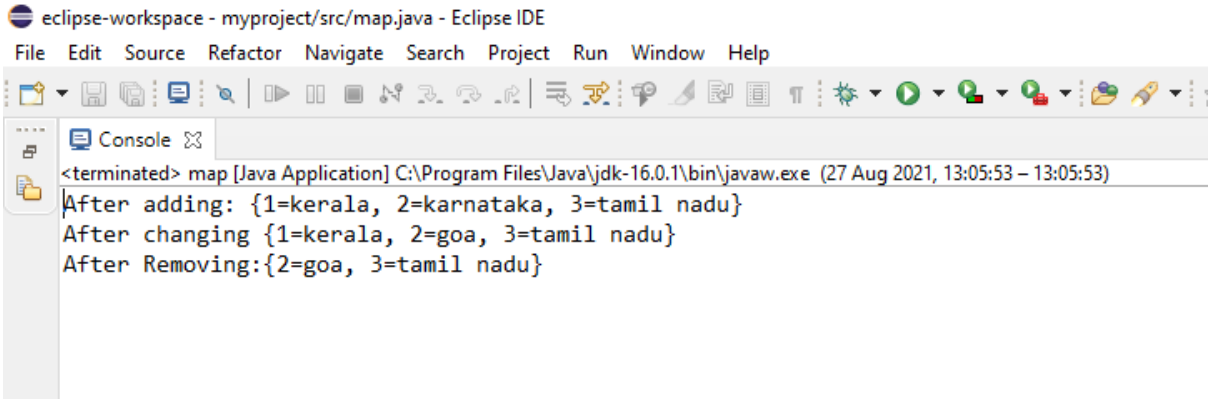
map.java	

```
import java.util.*;
public class map{
        public static void main(String args[])
        {
                // initialization of a Map
                Map<Integer, String> mp= new HashMap<Integer,
String>();

                // inserting the Elements
                mp.put(1,"kerala");
                mp.put(2,"karnataka");
                mp.put(3,"tamil nadu");

                System.out.println("After adding: " + mp);

    //changing an element
                mp.put(2,"goa");

                System.out.println("After changing " + mp);

                //removing an element
                mp.remove(1,"kerala");

    System.out.println("After Removing:"+mp);
        }
}
```

RESULT

The above program is executed and obtains the output.

OUTPUT



PROGRAM -18

AIM: Program to Convert HashMap to TreeMap

ALGORITHM

STEP 1: Start
STEP 2:Define a class with a main () to implement the concept.
STEP 3: Declare a HashMap and insert elements into it using the put() method.
STEP 4: Convert the above HashMap to TreeMap:
Map treeMap = new TreeMap<>(); treeMap.putAll(hashMap);
STEP 5: Display the resultant treeMap.
STEP 6: Stop

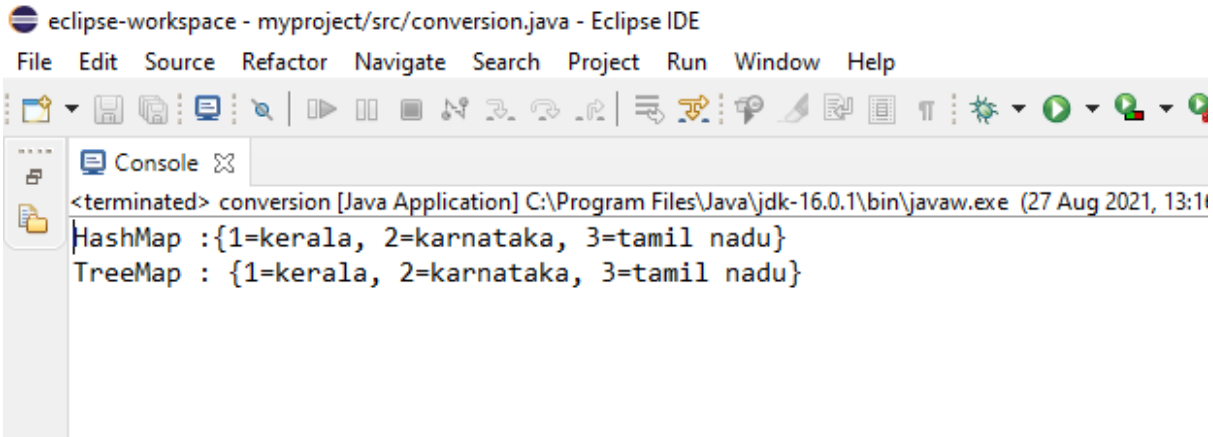
PROGRAM CODE

conversion.java	import java.util.*;
	public class conversion {
	public static void main(String args[]) {
	Map<String, String> map = new HashMap<>();
	map.put("1","kerala");
	map.put("2","karnataka");
	map.put("3","tamil nadu");
	
	System.out.println("HashMap :" + map);
	Map<String, String> treeMap = new TreeMap<>();
	treeMap.putAll(map);
	System.out.println("TreeMap : " + treeMap);
	}
	}

RESULT

The above program is executed and obtains the output.

OUTPUT


eclipse-workspace - myproject/src/conversion.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
<terminated> conversion [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (27 Aug 2021, 13:1
HashMap :{1=kerala, 2=karnataka, 3=tamil nadu}
TreeMap : {1=kerala, 2=karnataka, 3=tamil nadu}