OBJECT ORIENTED PROGRAMMING LAB

RECORD

SUJITH RAMAN S
RMCA B 27

Program no:**1** **Product**

date:

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.

Program:

```
public class Product
{    int pcode;
String pname;    int
price;
public static void main(String[] args) {
int smallest;
    Product p1 = new Product();
    Product p2 = new Product();
Product p3 = new Product();
p1.pcode=1001;
p1.pname="RAM";
p1.price=7000;
p2.pcode=1002;
p2.pname="Processor";
p2.price=37000;
p3.pcode=1001;
p3.pname="SSD";
p3.price=16700;
```

```
if(p1.price<p2.price) {
if(p3.price<p1.price) {
smallest = p3.price;
      } else {
         smallest = p1.price;
      }
   } else {
      if(p2.price<p3.price) {
smallest = p2.price;
      } else {
         smallest = p3.price;
      }
   }

   System.out.println(smallest + " is the cheapest.");
   }
 }
```

Output:

Program no:**2** **Matrix addition** *date:*

Aim: Read 2 matrices from the console and perform matrix addition. **Program**: import java.util.*; class matrixadd{

```
public static void main(String[] args)
{
int row,col,i,j;
Scanner sc=new Scanner(System.in);
System .out.print("enter the no of rows:");
row=sc.nextInt();
System .out.print("enter the no of
columns:"); col=sc.nextInt(); int mat1[][]=new
int[row][col]; int mat2[][]=new int[row][col];
int mat3[][]=new int[row][col];
System.out.print("enter the elements of matrix1 :");
for(i=0;i<row;i++)
{
for(j=0;j<col;j++)
{
mat1[i][j]=sc.nextInt();
}
System.out.println();
}
```

```
System.out.print("enter the elements of matrix2 :");
for(i=0;i<row;i++)
{
for(j=0;j<col;j++)
{
mat2[i][j]=sc.nextInt();
}
System.out.println();
}

for(i=0;i<row;i++)
{
for(j=0;j<col;j++)
{
mat3[i][j]=mat1[i][j]+mat2[i][j];
}
}
System.out.print("sum of matrix :\n");
for(i=0;i<row;i++)
{
for(j=0;j<col;j++)
{
System.out.print(mat3[i][j]+"\t");
}
```

```
System.out.println();
}
}
}
```

Output:

Program no:**3** **complex numbers**

date:

Aim: Add complex number **Program**:

```
public class Complex{
double a, b;
   Complex(double r, double
i){     this.a = r;      this.b = i;
   }
   public static Complex sum(Complex c1, Complex c2)
   {
       Complex temp = new Complex(0, 0);
```

```
        temp.a = c1.a + c2.a;
    temp.b = c1.b+ c2.b;
    return temp;
       }
       public static void main(String args[]) {
    Complex c1 = new Complex(5, 4);
       Complex c2 = new Complex(6, 3.5);
          Complex temp = sum(c1, c2);
          System.out.printf("Sum is: "+ temp.a+" + "+ temp.b +"i");
       }
     }
```

Output:

Program no:**4** **Symmetric** *date:*

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

Program: import java.util.Scanner;

```
    public class Symmetric
    {
       public static void main(String[] args)
```

```
{
    Scanner sc = new Scanner(System.in);

    System.out.println("Enter the no. of rows : ");

    int rows = sc.nextInt();

    System.out.println("Enter the no. of columns : ");

    int cols = sc.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] = sc.nextInt();
        }
    }

    System.out.println("Printing the input matrix :");
```

```
        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...");
        }
      }

      sc.close();
   }
}
```

Output:

Program no:**5** **cpu** *date:*

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.

Program:

```
class CPU {     double
price=27000;     class
Processor{        double
cores=8;
      String manufacturer="Intel";


   }
   protected class RAM{
double memory=16;
String manufacturer="OWC";

   }
```

```
}


public class Main2 {
   public static void main(String[] args) {
      CPU cpu = new CPU();
      CPU.Processor processor = cpu.new Processor();
      CPU.RAM ram = cpu.new RAM();
      System.out.println("CPU price = " + cpu.price);
      System.out.println("Processor cores = " + processor.cores);
      System.out.println("Processor manufacturer = " +
processor.manufacturer);
      System.out.println("RAM memory = " + ram.memory);
      System.out.println("RAM manufacturer = " + ram.manufacturer);
   }
}
```

Output:

Program:**6** **Sort String** *date:*

AIM: Program to Sort strings

Program:

```
 public class sortstringss{ public
static void main(String[] args)
{
String
names[]={"amal","jyothi","college","of","engineering"};
String temp; int n= names.length;
int i; int j;
for(i=0;i<n;i++
)
{
for(j=i+1;j<n;j++)
{
if(names[i].compareTo(names[j])>0)
{
        temp=names[i];     names[i]=names[j];         names[j]=temp;
}
}
}
System.out.println("the sorted array of string is :");
for(i=0;i<n;i++)
{
                                                    System.out.println(names[i]);
```

```
}
}
}
```

Output:

*Program no***:7** **Search an element** *date:*

AIM: *Search an element in an array.*

Program: import java.util.*; public

```
class search{
public static void main(String[] args)
{
int n,i,b,flag=0;
Scanner s=new Scanner(System.in);
System.out.println("enter the number of elements for the array
:"); n=s.nextInt(); int a[]=new int[n];
System.out.println("enter the elements of the array :");
for(i=0;i<n;i++)
{
a[i]=s.nextInt();
}
```

```
System.out.println("enter the element u want to search
:"); b=s.nextInt(); for(i=0;i<n;i++)
{
if(a[i]==b)
{
flag=1;
break;
}
else
{
                                                    flag=0;
}
}
if(flag==1)
{
System.out.println("element found at position :"+(i+1));
}
else
{
        System.out.println("element not found");
}
}
}
```

Output:

Program no:**8** **String manupulations** *date*:

AIM: Perform string manipulations **Program:**

```
public class Sample_String{   public
static void main(String[] args){
  String str_Sample = "littlyStar";
  System.out.println("Length of String: " + str_Sample.length());
  System.out.println("Character at position 5: " + str_Sample.charAt(5));
  System.out.println("EndsWith character 'r': " + str_Sample.endsWith("r"));
  System.out.println("Replace   'little' with   'super':       "       +
str_Sample.replace("littly", "super"));
}}
```

Output:

Program no:**9** **Area of shapes** *date:*

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.

Program:

```
public class shape
{ int s,as,ar; public
void area(int a)


{
s=a;
as=a*a;
System.out.println("area of square is"+as);
}
public void area(double r)
{
double radi=r; double
ac=(22/7)*radi*radi;
System.out.println("area of circle is"+ac);
}
public void area(int l,int w)
{
int len=l;
int wid=w;
ar=len*wid;
System.out.println("area of rectangle"+ar);
```

```
}
public void area(int h,double r)
{
int he=h;
double rad=r;
double acy=(2*(22/7)*rad*he)+((22/7)*rad*rad);
System.out.println("area of cylinder"+acy);

}
public static void main(String[] args)
{
shape o=new shape();
o.area(6);//area of square
o.area(2.42);//area of circle
o.area(3,1);//area of rectangle
o.area(5,4.7);
}
}
```

Output:

Program no:**10** **Employee** *date:*

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. **Program:**

```
import java.util.Scanner;

class Person

{

String name,gender,address;

int age;

public Person(String name, String gender, String address, int age)

{ super(); this.name = name; this.gender = gender; this.address =

address; this.age = age;

}

}

class Employee extends Person {

int empid;

String company_name,qualification;

double salary;

public Employee(String name, String gender, String address, int age, int
empid, String company_name,

String qualification, double salary) {

super(name, gender, address, age);

this.empid = empid;

this.company_name = company_name;
```

```
this.qualification = qualification;
this.salary = salary;
}
}
class Teacher extends Employee
{
String subject,department;
int teacherid;
public Teacher(String name, String gender, String address, int age, int
empid, String company_name,
String qualification, double salary, String subject, String department, int
teacherid) {
super(name, gender, address, age, empid, company_name, qualification,
salary);
this.subject = subject;
this.department = department;
this.teacherid = teacherid;
}
void display()
{
System.out.println("....Personal details...");
System.out.println(" Name : "+this.name+" Gender : "+this.gender+" Age
:"+this.age);
System.out.println("...Employee details....");
System.out.println("Empid : "+this.empid +" company_name :
"+this.company_name+" Salary : "+this.salary+" Address : "+this.address+"
qualification : "+this.qualification);
```

```
System.out.println("...Teacher's details...");
System.out.println("       teacherid     :       "+this.teacherid+   "
        department : "+this.department+" Subjects : "+this.subject);
}
}
public class Main {
public static void main(String[] args) {
Scanner s=new Scanner(System.in);
int n;
System.out.println("Enter number of Teachers : ");
n=s.nextInt();
Teacher obj[]=new Teacher[n];
for(int i=0;i<n;i++) {
System.out.println("Enter the person name:");
String nam1=s.next();
System.out.println("Enter the Gender: ");
String gen1=s.next();
System.out.println("Enter the Address: ");
String adr1=s.next();
System.out.println("Enter the Age:");
int age1=s.nextInt();
System.out.println("Enter the Employee id: ");
int id1=s.nextInt();
System.out.println("Enter the Company name: ");
String cname1=s.next();
```

```
System.out.println("Enter the Salary:");
double sal1=s.nextDouble();
System.out.println("Enter the Qualification:");
String qu1=s.next();
System.out.println("Enter the Teacher id: ");
int tid1=s.nextInt();
System.out.println("Enter the Department:");
String dept1=s.next();
System.out.println("Enter the
Subject:"); String sub1=s.next();
obj[i]=new
Teacher(nam1,gen1,adr1,age1,id1,cname1,qu1,sal1,sub1,dept1,tid1);
}
System.out.println("\n-----------------------------------------------------------------------
---\n"); for(int
i=0;i<n;i++) {
obj[i].display();
}
}
}
```

Output:

Program no:**11** **Person** *Date*:

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.

Program:

```
import java.util.*;
class Employee
 {
int empid;
String name,address;
double salary;
```

```
public Employee(int empid, String name, String address, double salary)
{ this.empid = empid; this.name = name; this.address = address;
this.salary = salary;
}
}
public class Teacher extends Employee
{
String subject,department;
public Teacher(int empid, String name, String address, double salary,String department,String subject ) { super(empid, name, address, salary); this.subject = subject; this.department = department;
}
void display()
{
System.out.println("Empid : "+this.empid+" Name : "+this.name+" Salary : "+this.salary+" Address : "+this.address+" department : "+this.department+" Subjects : "+this.subject);
}
public static void main(String[] args) {
// TODO Auto-generated method stub
Scanner sc=new Scanner(System.in);
int n;
System.out.println("Enter number of Teachers : ");
n=sc.nextInt();
```

```
Teacher obj[]=new
Teacher[n]; for(int i=0;i<n;i++)
{ int j = i+1;
System.out.print("Enter Empid of teacher "+j+" : ");
int Empid = sc.nextInt();
System.out.print("Enter Name of teacher "+j+" : ");
String Name = sc.next();
System.out.print("Enter Salary of teacher "+j+" : ");
double Salary = sc.nextDouble();
System.out.print("Enter Address of teacher "+j+" : ");
String Address = sc.next();
System.out.print("Enter department of teacher "+j+" : ");
String department =sc.next();
System.out.print("Enter Subjects of teacher "+j+" : ");
String Subjects =sc.next();
obj[i] = new Teacher(Empid, Name, Address, Salary, department, Subjects);
}
System.out.println("\n-----------------------------------------------------------------------
--\n");
System.out.println("Teacher's List
\n"); for(int i=0;i<n;i++) {
obj[i].display();
}
}
}
```

Output:

Program no:**12** **BOOKS** *date:*

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.

Program:

```
import java.util.Scanner;
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()
{
System.out.println("Literature Books");
}
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 no:***13** **Student and sports** *date:*

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

Program:

```
interface student

{

 void stresullt();

 }

interface sports

{

void spresult();

}

class result implements student,sports{
```

```
public void spresult()
      {
String hundred="First";
String twohundred="Second";
String fivehundred="First";
String relay="Second";
System.out.println("Sports Result");
 System.out.println("Hundred Meter:"+hundred);
System.out.println("Two Hundred Meter:"+twohundred);
System.out.println("Five Hundred Meter:"+fivehundred);
System.out.println("Relay:"+relay);
}
public void stresullt()
{
int physics=30;
int
chemistry=40;
int maths=45; int
english=50; int
computer=50;
 System.out.println("Marks");
System.out.println("Physics:"+physics);
System.out.println("Chemistry:"+chemistry);
```

```
            System.out.println("Mathematics:"+maths);
System.out.println("English:"+english);
            System.out.println("Computer:"+computer);
            }
            public static void main(String[] args)
            {
            result r = new result();
            r.stresullt();
            r.spresult();

            }

            }
```

Output:

Program no*:14** **Area And Perimeter** ***date:

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.

PROGRAM:

```
public class shape
{
int s,as,ar;
public void area(int a)//area of square

{
s=a;
as=a*a;
System.out.println("area of square is"+as);
}
public void area(double r)//area of circle
{
double radi=r; double
ac=(22/7)*radi*radi;
System.out.println("area of circle is"+ac);
}
public void area(int l,int w)//area of rectangle
{
int len=l;
```

```
int wid=w;
ar=len*wid;
System.out.println("area of rectangle"+ar);
}
public void area(int h,double r)//area of cylinder
{
int he=h; double rad=r; double
acy=(2*(22/7)*rad*he)+((22/7)*rad*rad);
System.out.println("area of cylinder"+acy);

}
public static void main(String[] args)
{
shape o=new shape();
o.area(6);//area of square
o.area(2.42);//area of circle
o.area(3,1);//area of rectangle
o.area(5,4.7);                                    //area of cylinder
}
}
```

OUTPUT

Program no*: 15** **ProductBill** ***date:

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

Program:

```
interface bill
{
int productdetails();
}
class product1 implements bill{ int id =
101,quantity= 2,unit=25,total=0;
String name="A";

public int productdetails()
{
total = quantity * unit;
System.out.println("Product Id :"+id);
System.out.println("Name :"+name);
```

```
System.out.println("Quantity :"+quantity);
System.out.println("Unit price :"+unit);
System.out.println("Total :"+total);
return(total);

}
}
class product2 implements bill{ int id =
102,quantity= 1,unit=100,total=0;
String name="B";

public int productdetails()
{
total = quantity * unit;
System.out.println("Product Id :"+id);
System.out.println("Name :"+name);
System.out.println("Quantity :"+quantity);
System.out.println("Unit price :"+unit);
System.out.println("Total :"+total);
return(total);

}
}
```

```
public class productbill
{
 public static void main(String[] args)
{
 product1 p1 = new
product1();  product2 p2 =
new product2(); int t1=
p1.productdetails(); int t2=
p2.productdetails(); int
t3=t1+t2;

System.out.println("Net. Amount :"+t3);

}
}
```

OUTPUT

Program no*: 16** **Authendication** ***date:

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

Program: import

```
java.util.Scanner;


class UsernameException extends Exception {


 public UsernameException(String msg) {
super(msg);
 }
}

class PasswordException extends Exception {


 public PasswordException(String msg) {
super(msg);
 }
}

public class CheckLoginCredential {

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

```
  Scanner s = new Scanner(System.in);
 String username, password;

 System.out.print("Enter username :: ");
username = s.nextLine();

 System.out.print("Enter password :: ");
password = s.nextLine();

 int length = username.length();

 try {
  if(length < 6)
   throw new UsernameException("Username must be greater than 6
characters ???");
  else if(!password.equals("hello"))
   throw      new   PasswordException("Incorrect   password\nType
       correct password ???");
  else
   System.out.println("Login Successful !!!");
 }
 catch (UsernameException u) {
  u.printStackTrace();
 }
```

```
 catch (PasswordException p) {
  p.printStackTrace();
 }
 finally {
  System.out.println("The finally statement is executed");
 }
}
}
```

OUTPUT

Program no*: 17 Average *date:

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

Program:

```
import java.util.Scanner; import
java.util.InputMismatchException; public
class TestDemo
```

```
{
public static void main(String args[])
{
double total = 0, N, userInput; Scanner
input = new Scanner(System.in); while
(true)
{
 System.out.print("Enter how many numbers(N) to calculate
average:");  userInput = input.nextDouble();   if (userInput > 0)
  {
N = userInput;
break;
  }
  else
System.out.println("N must be positive.");
}
for (int i = 0; i < N; i++)
{
while (true)
{
System.out.print("Enter number:");
try
{
```

```
userInput =
input.nextDouble(); total +=
userInput; break;
}
catch (InputMismatchException e)
{input.nextLine();
System.out.println("Input must bea number. Try again");
}
}
}
System.out.println("Average: "+ total / N);
}
}
```

OUTPUT

Program no*: 18** **Thread** ***date:

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

Program:

```
import java.util.Scanner; class
        MulTable extends Thread{
        public void run() { int num = 5;
        System.out.printf("_____Multiplication Table of 5_____\n");
         for(int i = 1; i <= 10; ++i)
         {
         System.out.printf("%d * %d = %d \n", num, i, num * i);
         }
         }
        }
        class PrimeNo extends Thread{
        public void run() {
        int i, j,flag;
        Scanner s = new Scanner(System.in);
        System.out.println("\n_____To generate first N prime
        numbers_____"); System.out.println("Enter the limit (N):"); int N =
        s.nextInt();

        System.out.println("Prime numbers between 1 and " + N + " are:");
         for (i = 1; i <= N; i++)
         {
         if (i == 1 || i == 0)
```

```
 continue;  flag = 1;
for (j = 2; j <= i / 2; ++j)
 {
 if (i % j == 0)
 {
 flag = 0;
break;
 }
 }
 if (flag == 1)
 System.out.print(i + " ");
 }
}
}
public class ThreadClass { public static void main(String[] args)
throws InterruptedException { MulTable m = new MulTable();
m.start();
m.sleep(200);
PrimeNo p = new PrimeNo();
p.start();
p.sleep(200);
// TODO Auto-generated method stub
}
```

}

OUTPUT

Program no*: 19 Fibonacci *date:

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

```
public class Mythread {

   public static void main(String[] args) {
      Runnable r = new Runnable1();
Thread t = new Thread(r);
      t.start();
      Runnable r2 = new Runnable2();
Thread t2 = new Thread(r2);
      t2.start();
```

```
   }
}

class Runnable2 implements Runnable{
   public void run(){
for(int i=0;i<11;i++){
if(i%2 == 1)
            System.out.println(i);
      }
   }
}

class Runnable1 implements Runnable{
   public void run(){        int
n1=0,n2=1,n3,i,count=10;
 System.out.print(n1+" "+n2);//printing 0 and 1      for(i=2;i<count;++i)//loop
starts from 2 because 0 and 1 are already printed
 {
  n3=n1+n2;
  System.out.print("
"+n3);      n1=n2;
n2=n3;
 }
   }
```

}

OUTPUT

Program no*: 20 BubbleSort *date:

AIM: Using generic method perform Bubble sort.

Program:

```
public class BubbleSort
{
 static void bubbleSort(int[] arr)
{
 int n = arr.length;
int temp = 0;
 for(int i = 0; i < n; i++) {
for(int j=1; j < (n-i); j++)
{   if(arr[j-1] >
arr[j])
 {
```

```
 temp = arr[j-1];
arr[j-1] = arr[j];
arr[j] = temp;
 }
 }
}
 }
 public static void main(String[] args)
{
 int arr[] = { 2, 5, -2, 6, -3, 8, 0, -7, -9, 4 };
 System.out.println("Array Before Bubble Sort");
 for(int i = 0; i < arr.length; i++)
 {
 System.out.print(arr[i] + " ");
}
 System.out.println();
bubbleSort(arr);
 System.out.println("Array After Bubble Sort");
 for(int i = 0; i < arr.length; i++)
{
 System.out.print(arr[i] + " ");
 }
 }
```

}

OUTPUT

Program no: **21** **Array List** *date:*

AIM:

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

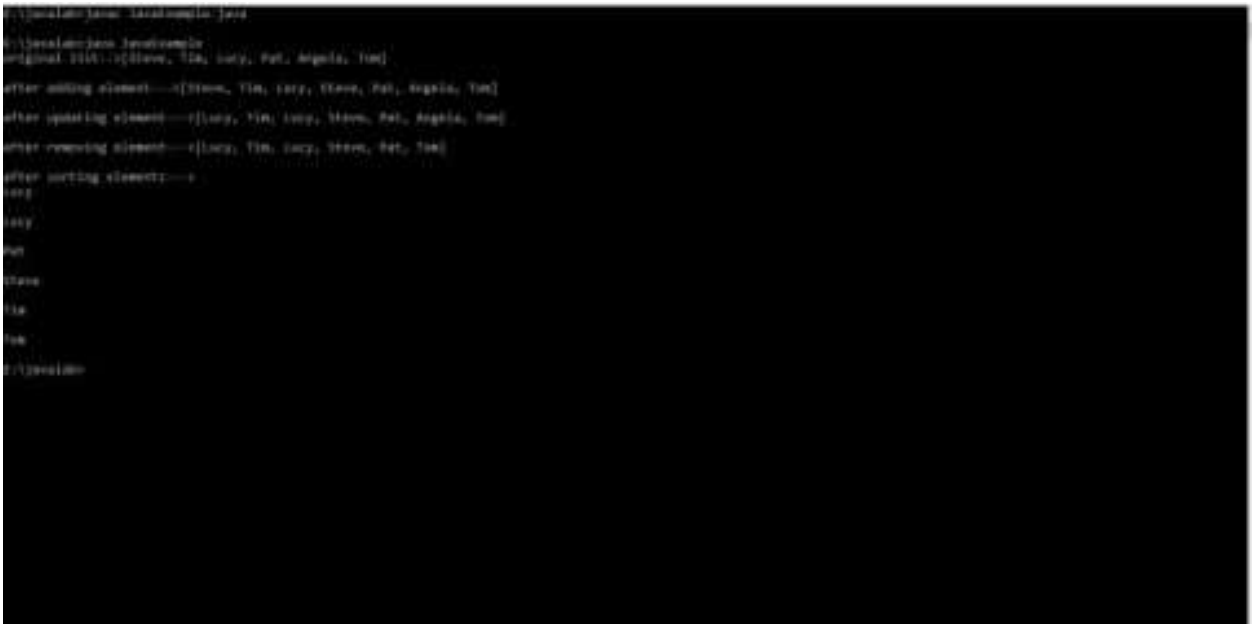
Program:

```
import java.util.*;  import
java.util.Collections;   class
JavaExample{
 public static void main(String args[]){
 ArrayList<String> alist=new
ArrayList<String>();    alist.add("Steve");
alist.add("Tim");   alist.add("Lucy");
alist.add("Pat");   alist.add("Angela");
```

```
alist.add("Tom");

//displaying elements
        System.out.print("original list-->");
System.out.println(alist);
        System.out.println();

//Adding "Steve" at the fourth position
alist.add(3, "Steve");

//displaying elements
        System.out.print("after adding element--->");
System.out.println(alist);
        System.out.println();

//update element
alist.set(0, "Lucy");

//displaying elements
        System.out.print("after updating element--->");
System.out.println(alist);
        System.out.println();


//remove elements
alist.remove("tom");
alist.remove("Angela");

//displaying elements
        System.out.print("after removing element--->");
System.out.println(alist);
        System.out.println();

//sorting arraylist
        System.out.print("after sorting elements--->");
```

```
        Collections.sort(alist);  for
        (String str : alist) {
        System.out.println();
    System.out.println(str);
    }
    }
   }
```

OUTPUT



Program no: **22** **Linked List** *date*

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

```
import java.util.*;   public class
        removelink {
    public static void main(String[] args) {
    // create an empty linked list
    LinkedList<String> l_list = new
   LinkedList<String>();   // use add() method to add
```

```
values in the linked list   l_list.add("hello");
l_list.add("how");   l_list.add("are");
l_list.add("you");   l_list.add("?");


 // print the list
 System.out.println("The Original linked list: " + l_list);

 // Removing all the elements from the linked list
l_list.clear();

 System.out.println("The New linked list: " + l_list);
 }
}
```

OUTPUT

Program no: **23** **Deque** *date:*

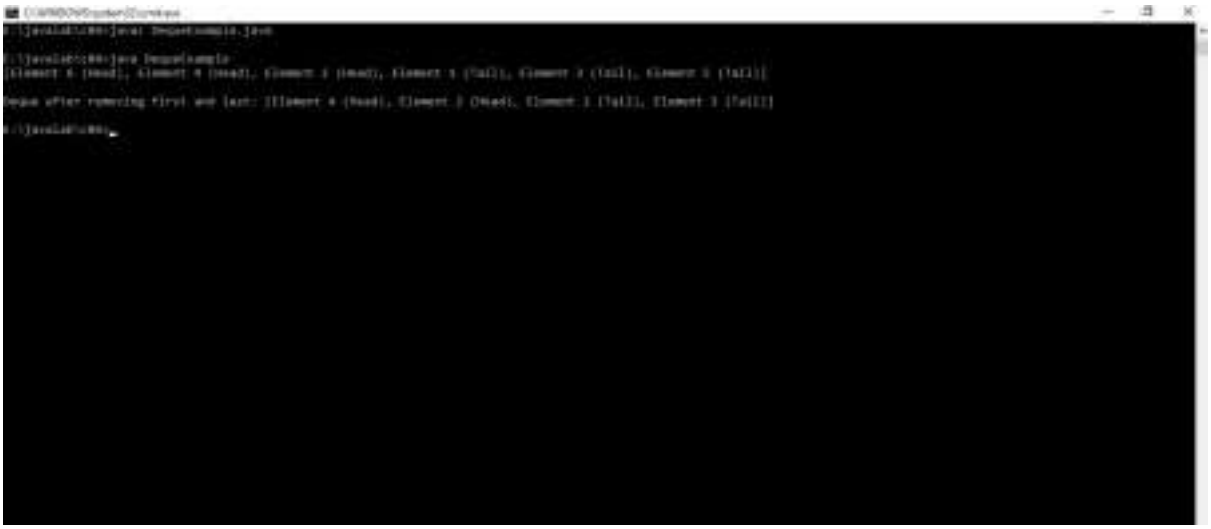
AIM: Program to demonstrate the addition and deletion of elements in deque

Program:

```
dequeue import java.util.*;  public class
    DequeExample {   public static void
    main(String[] args)
     {

     Deque<String> deque
     = new LinkedList<String>();

     // We can add elements to the queue
    // in various ways

     // Add at the last
    deque.add("Element 1 (Tail)");

     // Add at the first
    deque.addFirst("Element 2 (Head)");

     // Add at the last
    deque.addLast("Element 3 (Tail)");

     // Add at the first
    deque.push("Element 4 (Head)");

     // Add at the last
    deque.offer("Element 5 (Tail)");

     // Add at the first
    deque.offerFirst("Element 6

    (Head)");  System.out.println(deque +

    "\n");

     // We can remove the first element
    // or the last element.
    deque.removeFirst();
    deque.removeLast();
     System.out.println("Deque after removing "  +
```

```
                "first and last: " + deque);
    }
    }
```

OUTPUT



Program no: **24** **Map Interface** *date:*

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

Program:

```
: java.util.*;  class
    hashmap {
     public static void main(String args[])
     {
     Map<String, Integer> hm
     = new HashMap<String, Integer>();
```

```
 hm.put("c", new Integer(100));
hm.put("d", new Integer(700));
hm.put("e", new Integer(900));
hm.put("f", new Integer(200));   //
Traversing through the map
 for (Map.Entry<String, Integer> me : hm.entrySet()) {
 System.out.print(me.getKey() + ":");
 System.out.println(me.getValue());
 }
 }
}
```

OUTPUT

Program no: **25** **Map Interface** *date:*

AIM:

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

```
import java.util.*;
import java.util.stream.*;
public class HT {
```

```
 public static void main(String args[]) {
 Map<String, String> map = new
HashMap<>();   map.put("1", "One");
map.put("2", "Two");
 map.put("3", "Three");
map.put("4", "Four");
map.put("5", "Five");
map.put("6", "Six");
map.put("7", "Seven");
map.put("8", "Eight");
map.put("9", "Nine");
 System.out.println("HashMap = " + map);
 Map<String, String> treeMap = new TreeMap<>();
treeMap.putAll(map);
 System.out.println("TreeMap (HashMap to TreeMap) " + treeMap);
 }
}
```

OUTPUT

Program no: **26** **Stack operations** *date:*

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

Program:

```
        // Java Program to Implement Stack in Java Using Array and
// Generics
// Importing input output classes  import
java.io.*;
// Importing all utility classes  import
java.util.*;

// user defined class for generic stack  class
stack<T> {
        // Empty array list
        ArrayList<T> A;
        // Default value of top variable when stack is empty  int
        top = -1;
        // Variable to store size of array  int
        size;
        // Constructor of this class
        // To initialize stack
        stack(int size)
        {
                // Storing the value of size into global variable  this.size
                = size;
```

```
        // Creating array of Size = size  this.A
        = new ArrayList<T>(size);
}
// Method 1
// To push generic element into stack  void
push(T X)

{
        // Checking if array is full
        if (top + 1 == size)
        {

                // Display message when array is full
                System.out.println("Stack Overflow");
        }
        else
        {

                // Increment top to go to next position  top
                = top + 1;

                // Over-writing existing element
                if (A.size() > top)  A.set(top, X);
                else
                        // Creating new element
```

```
                    A.add(X);
        }
    }
// Method 2
// To return topmost element of stack
T top()
{
        // If stack is empty  if
        (top == -1)
        {

            // Display message when there are no
            elements in // the stack
            System.out.println("Stack Underflow");

            return null;
        }

        // else elements are present so
        // return the topmost element
        else  return A.get(top);
    }
// Method 3
```

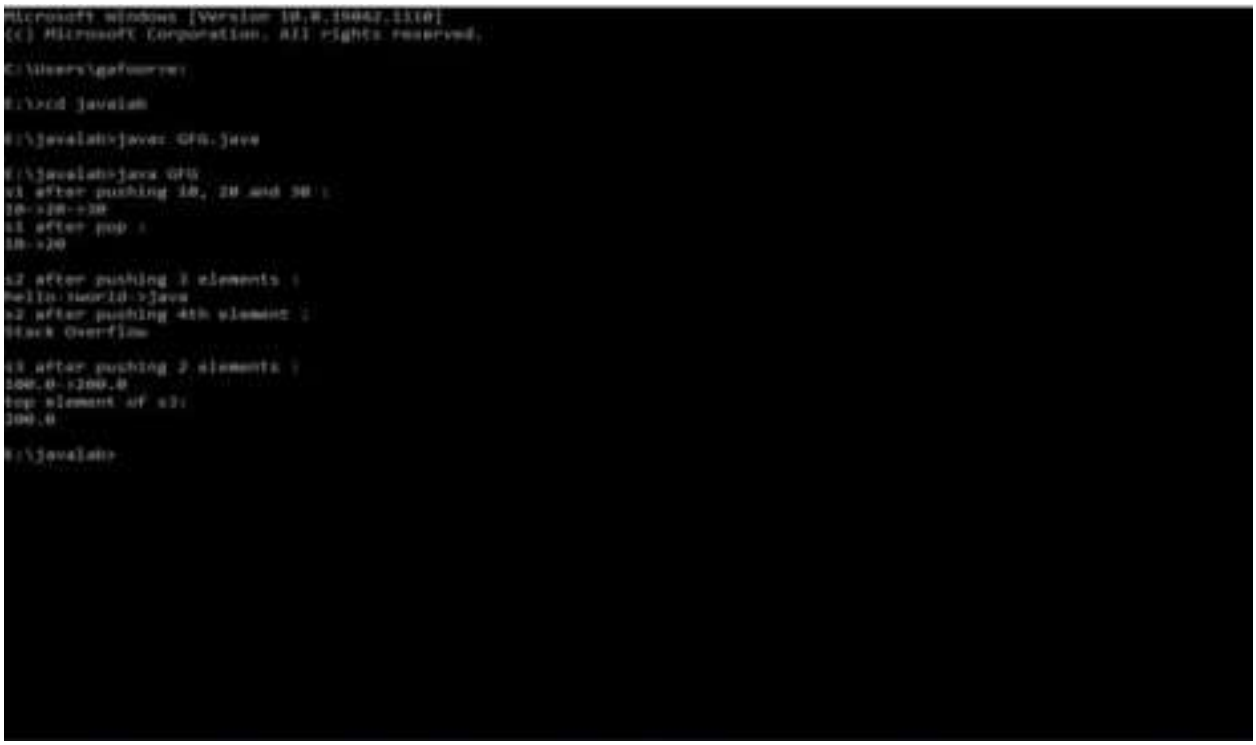
```
// To delete last element of stack  void
pop()
{
        // If stack is empty if
        (top == -1)
        {
                // Display message when there are no elements in
                // the stack
                System.out.println("Stack Underflow");
        }

        else

                // Delete the last element  //
                by decrementing the top
                top--;
}
// Method 4
// To check if stack is empty or not  boolean
empty() { return top == -1;
}
// Method 5
```

```
        // To print the stack
        // @Override  public
        String toString()
        {
                String Ans = "";  for (int
                i = 0; i < top; i++)
                 {
                        Ans += String.valueOf(A.get(i)) + "->";
                }
                Ans += String.valueOf(A.get(top));  return
                Ans;
        }
}
// Main Class  public class GFG {  // main
driver method  public static void
main(String[] args)
        {
                // Integer Stack
                // Creating an object of Stack class  //
                Declaring objects of Integer type
                stack<Integer> s1 = new stack<>(3);
                // Pushing elements to integer stack - s1
                // Element 1 - 10
                s1.push(10);  //
```

```
Element 2 - 20
s1.push(20);  //
Element 3 - 30
s1.push(30);
// Print the stack elements after pushing the
// elements
System.out.println("s1 after pushing 10,
20 and 30 :\n" + s1); // Now, pop from
stack s1  s1.pop();
// Print the stack elements after poping few
// element/s
System.out.println("s1 after pop :\n" + s1);  stack<String>
s2 = new stack<>(3);
// Pushing elements to string stack - s2
// Element 1 - hello
s2.push("hello");  //
Element 2 - world
s2.push("world");
// Element 3 - java s2.push("java");
// Print string stack after pushing above string
// elements
System.out.println("\ns2 after pushing 3
elements :\n" + s2);
```

```
            System.out.println("s2 after pushing 4th element

            :");

            // Pushing another element to above stack

            // Element 4 - GFG  s2.push("GFG");

            // Float stack

            // Creating an object of Stack class  //

            Declaring objects of Integer type

            stack<Float> s3 = new stack<>(2);

            // Pushing elements to float stack - s3

            // Element 1 - 100.0

            s3.push(100.0f);  //

            Element 2 - 200.0

            s3.push(200.0f);

            // Print string stack after pushing above float

            // elements

            System.out.println("\ns3 after pushing 2

            elements :\n" + s3); // Print and display

            top element of stack s3

            System.out.println("top element of s3:\n"+ s3.top());

        }

}
```

OUTPUT



Program no: **27** **Figures** date:

AIM: Program to draw Circle, Rectangle, Line in Applet **Program:**

```
import java.applet.*; import
java.awt.Graphics; public class
figures extends Applet
{
public void paint(Graphics g)
{
g.drawLine(30,30,300,30);
g.drawOval(100,100,100,100);
g.drawRect(250, 250, 200, 100);
}
}
```

OUTPUT

Program no: **28** **Numbers** date:

AIM: Program to find maximum of three numbers using AWT.

Program:

```
import java.awt.*; import java.awt.event.*; import
java.applet.*; public class largest extends Applet implements
ActionListener {
 int a, b, c, result;
 String str;
 TextField Txt1 = new TextField(10);
 TextField Txt2 = new TextField(10);
 TextField Txt3 = new TextField(10);
 TextField t4 = new TextField(10);
 Label l2 = new Label("enter number 1: ");
 Label l3 = new Label("enter number 2: ");
 Label l5 = new Label("enter number 3: ");
```

```
 Label l4 = new Label("largest : ");
Button b1 = new Button("click");
 public void init() {  add(l2);
add(Txt1);  add(l3);
add(Txt2);  add(l5);
add(Txt3);  add(b1);
add(l4);  add(t4);
b1.addActionListener(this);
 }
 public void actionPerformed(ActionEvent e) {
if (e.getSource() == b1)
 {
 str = Txt1.getText();  a =
Integer.parseInt(str);  str
= Txt2.getText();  b =
Integer.parseInt(str);  str
= Txt3.getText();  c =
Integer.parseInt(str);  if
(a >= b && a >= c) {
result = a;
t4.setText(String.valueO
f(a));  repaint();  } else if
(b >= a && b >= c) {
```

```
result = b;
t4.setText(String.valueO
f(b));  repaint();  } else {
result = c;
t4.setText(String.valueO
f(c));  repaint();
 }
 }
 }
}
```

OUTPUT

Program no: **29** **STudents** date:

AIM: Find the percentage of marks obtained by a student in 5 subjects. Display a happy face if he secures above 50% or a sad face if otherwise..

Program:

```
import java.awt.*; import java.awt.event.*; import
java.applet.*; public class myline extends Applet implements
ActionListener {  private int SMILE = 0;  private float k;
 int i;
float j;
 TextField T1 = new TextField(10);
 TextField T2 = new TextField(10);
 TextField t3 = new TextField(10);
 Label l2 = new Label("enter total marks obtained : ");
 Label l3 = new Label("enter total Marks : ");
 Label l4 = new Label("percentage : ");
Button b = new Button("percentage");
 public void init()
{  add(l2);
add(T1);  add(l3);
add(T2);  add(l4);
add(t3);  add(b);
 b.addActionListener(this);
 }
 public void actionPerformed(ActionEvent e)
{  if (e.getSource() == b)  i =
Integer.parseInt(T1.getText());  j =
Integer.parseInt(T2.getText());
```

```
 k = i / j;
 k = k * 100;
if (k >= 50) {
SMILE = 1;
 } else {
 SMILE = 0;
 }
 t3.setText(String.valueOf(k) + " %");
repaint();
 }
 public void paint(Graphics g) {
 g.drawOval(80, 70, 150, 150);
 g.setColor(Color.black);
 g.fillOval(120, 120, 15, 15);
 g.fillOval(170, 120, 15, 15);
if (SMILE == 1) {
 g.drawArc(130, 180, 50, 20, 180, 180);
 SMILE = 0;
 } else {
 g.drawArc(130, 180, 50, 20, 180, -180);
 }
 }
}
```

OUTPUT

Program no: **30** **Students** date:

AIM:

Using 2D graphics commands in an Applet, construct a house. On mouse click event, change the color of the door from blue to red.

Program:

import java.applet.*; import

java.awt.*;

import java.awt.event.MouseEvent; import

java.awt.event.MouseListener;

public class house extends Applet implements MouseListener, Runnable {

private Color door = Color.blue; public void paint(Graphics g) { int x[] = {

150, 300, 225 }; int y[] = { 150, 150, 25 };

g.setColor(Color.orange);

```
 g.fillRect(150, 150, 150, 200);
 g.drawRect(150, 150, 150, 200);
 g.setColor(door);
 g.fillRect(200, 200, 50, 150);
 g.drawRect(200, 200, 50, 150);
 g.setColor(Color.red);
 g.fillPolygon(x, y, 3);
 g.drawPolygon(x, y, 3);
 }
 public void init() {
this.setSize(200, 200);
addMouseListener(this);
 }
 public void run() {
while (true) {
repaint();
 try {
 Thread.sleep(5);  } catch
(InterruptedException e) {
 e.printStackTrace();
 }
 }
 }
```

```
 public void mouseClicked(MouseEvent e) {
 int x = e.getX(), y = e.getY();
if (x <= 300)  door =
Color.red;  else  door =
Color.blue;  repaint();
 }
 public void mousePressed(MouseEvent e) {
 }
 public void mouseReleased(MouseEvent e) {
 }
 public void mouseEntered(MouseEvent e) {
 }
 public void mouseExited(MouseEvent e) {
 }
}
```

OUTPUT


Applet Viewer: house.class
Applet