

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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.

Input:

```
public class EXP1
{
    public static void main(String[] args)
    {
        Product p1=new Product();
        p1.pcode="car123";
        p1.pname="Benz";
        p1.price=100000;
        System.out.println("*****Displaying~p1*****");
        p1.display();

        Product p2=new Product("jaguar","car426",250000);
        System.out.println("*****Displaying~p2*****");
        p2.display();

        Product p3=new Product("maruthi","car800",550000);
        System.out.println("*****Displaying~p3*****");
        p3.display();

        Product p=p3.getprice()<(p1.price
        <p2.price?p1.price:p2.price)?p3:(p1.price<p2.price?p1:p2);
        System.out.println("\n*****Displaying~product~with~lowest~price*****
        *****");
        p.display();
    }
}
class Product
{
    String pname,pcode;
    int price;
```

```

public String getpname()
{
return pname;
}
public Product()
{

}
public Product(String pname,String pcode,int price)
{
this.pname=pname;
this.pcode=pcode;
this.price=price;
}
public void setpname(String pname)
{
this.pname=pname;
}
public String getpcode()
{
return pcode;
}
public void setpcode(String pcode)
{
this.pcode=pcode;
}
public int getprice()
{
return price;
}
public void setprice(int price)
{
this.price=price;
}
public void display()
{
System.out.println("pcode:...."+this.pcode);
System.out.println("pname:...."+this.pname);
System.out.println("price:...."+this.price);
}
}

```

Output:

```
*****Displaying~p1*****
pcode:...car123
pname:...Benz
price:...100000
*****Displaying~p2*****
pcode:...car426
pname:...jaguar
price:...250000
*****Displaying~p3*****
pcode:...car800
pname:...maruthi
price:...550000

*****Displaying~product~with~lowest~price*****
pcode:...car123
pname:...Benz
price:...100000
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To read 2 matrices from the console and perform matrix addition.

Input:

```
import java.util.Scanner;
public class MatrixAdd
{
    public static void main (String[] args)
    {
        int p,q,m,n;
        Scanner S= new Scanner(System.in);
        System.out.println("Enter the number of rows in first matrix");
        p=S.nextInt();
        System.out.println("Enter the number of columns in first matrix");
        q=S.nextInt();
        System.out.println("Enter the number of rows in second matrix");
        m=S.nextInt();
        System.out.println("Enter the number of columns in second matrix");
        n=S.nextInt();
        if(p==m&&q==n)
        {
            int a[][]=new int [p][q];
            int b[][]=new int[m][n];
            int c[][]=new int[m][n];
            System.out.println("Enter all the elements of first matrix:");
            for(int i=0;i<p;i++)
            for(int j=0;j<q;j++)
            a[i][j]=S.nextInt();
            System.out.println("Enter all the elements of the second matrix");
            for(int i=0;i<m;i++)
            for(int j=0;j<n;j++)
            b[i][j]=S.nextInt();
            System.out.println("First Matrix:");
            for(int i=0;i<p;i++)
            {
                for(int j=0;j<q;j++)
                System.out.print(a[i][j]+" ");
                System.out.println(" ");
            }
        }
    }
}
```

```

    }
    System.out.println("Second matrix:");
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
        {
            System.out.print(b[i][j]+" ");
        }
        System.out.println(" ");
    }
    for (int i=0;i<p;i++)
    for(int j=0;j<n;j++)
    for(int k=0;k<q;k++)
    c[i][j]=a[i][j]+b[i][j];
    System.out.println("Matrix after addition:");
    for(int i=0;i<p;i++)
    {
        for(int j=0;j<n;j++)
        System.out.print(c[i][j]+" ");
        System.out.println(" ");
    }
}
else
{
    System.out.println("Addition would not be possible");
}
}
}

```

Output:

```

Enter the number of rows in first matrix
2
Enter the number of columns in first matrix
2
Enter the number of rows in second matrix
2
Enter the number of columns in second matrix
2
Enter all the elements of first matrix:
4
6
7
2
Enter all the elements of the second matrix
6
8
2
1
First Matrix:
4 6
7 2
Second matrix:
6 8
2 1
Matrix after addition:
10 14
9 3

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

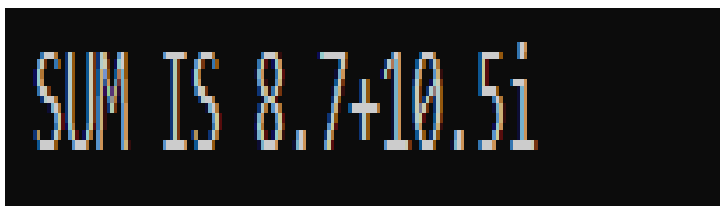
Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To add two complex numbers.

Input:

```
public class complexnumber
{
double real,img;
complexnumber(double r,double i)
{
this.real=r;
this.img=i;
}
public static complexnumber
sum(complexnumber c1, complexnumber c2)
{
complexnumber temp=new complexnumber(0,0);
temp.real=c1.real+c2.real;
temp.img=c1.img+c2.img;
return temp;
}
public static void main(String args[])
{
complexnumber c1=new complexnumber(7.5,6);
complexnumber c2=new complexnumber(1.2,4.5);
complexnumber temp=sum(c1,c2);
System.out.println(" SUM IS "+ temp.real+" "+temp.img+"i");
}
}
```

Output:



```
SUM IS 8.7+10.5i
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
import java.util.Scanner;
public class symmetricmatrixprgrm
{
    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 element");
        for(int i=0; i<rows;i++)
        for(int j=0; j<cols;j++)
        matrix[i][j]=sc.nextInt();
        sc.close();
        System.out.println("Printing 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:");
        else
        {
            boolean symmetric=true;
            for(int i=0; i<rows; i++)
            for(int j=0; j<cols; j++)
            if(matrix[i][j]!=matrix[j][i])
            {
                symmetric=false;
                break;
            }
        }
    }
}
```

```

    }
    if(symmetric)
    {
        System.out.println("The given matrix is symmetric.....");
    }
    else
    {
        System.out.println("The given matrix is not symmetric.....");
    }
}
}
}
}

```

Output:

```

Enter the no of rows
3
Enter the no of columns
3
Enter the element
2 4 4
4 8 8
4 8 8
Printing input matrix
2      4      4
4      8      8
4      8      8
The given matrix is symmetric.....

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

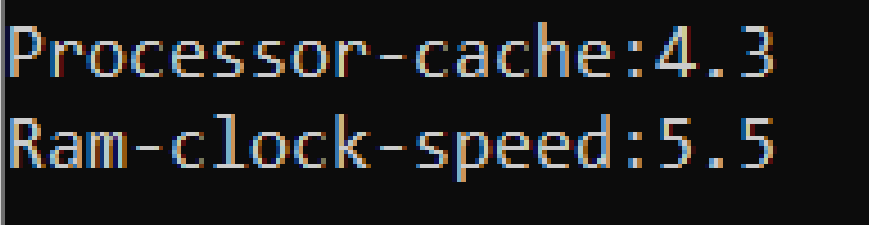
Aim: To create CPU with attribute price. Create inner class processor (no of cores, manufactures)and static nested class RAM (memory , manufacturer).Create an object of CPU and print information of processor and RAM

Input:

```
class CPU
{
double price;
class Processor
{
double cores;
String manufacturer;
double getcache()
{
return 4.3;
}
}
protected class RAM
{
double memory;
String manufacturer;
double getclockspeed()
{
return 5.5;
}
}
public class CPUdetails
{
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("Processor-cache:"+processor.getcache());
System.out.println("Ram-clock-speed:"+ram.getclockspeed());
```

```
}  
}
```

Output:

A terminal window with a black background and yellow text. The first line reads "Processor-cache:4.3" and the second line reads "Ram-clock-speed:5.5".

```
Processor-cache:4.3  
Ram-clock-speed:5.5
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To write menu driven program that would choose either inbuilt method or call a user defined method to sort an array of strings.

Input:

```
import java.util.Arrays;
import java.util.Scanner;
public class StringSort
{
    public static void main(String[] args)
    {
        int count=0;
        String tmp;
        Scanner scan=new Scanner(System.in);
        System.out.println("Enter~number~of~strings~to~sort:");
        count=scan.nextInt();
        String str_list[]=new String[count];
        Scanner scan1=new Scanner(System.in);
        System.out.println("Enter~your~strings:");
        for(int i=0;i<count;i++)
            str_list[i]=scan1.nextLine();
        System.out.println("choose~1~or~2~from~the~menu~below");
        System.out.println("1:~inbuilt~sort()");
        System.out.println("2:~user~defined~sorting~logic()");
        int choice;
        choice=scan.nextInt();
        switch(choice)
        {
            case 1:Arrays.sort(str_list);
                System.out.println(Arrays.toString(str_list));
                break;
            case 2:for(int i=0;i<count-1;i++)
                    for(int j=i+1;j<str_list.length;j++)
                        if(str_list[i].compareTo(str_list[j])>0)
                        {
```

```

        tmp=str_list[i];
        str_list[i]=str_list[j];
        str_list[j]=tmp;
    }
    System.out.println(Arrays.toString(str_list));
    break;
}
}
}

```

Output:

```

Enter~number~of~strings~to~sort:
7
Enter~your~strings:
wolf
cat
rose
apple
goat
bat
money
choose~1~or~2~from~the~menu~below
1:~inbuilt~sort()
2:~user~defined~sorting~logic()
1
[apple, bat , cat, goat, money, rose, wolf]

```

```

Enter~number~of~strings~to~sort:
7
Enter~your~strings:
wolf
cat
rose
apple
goat
bat
money
choose~1~or~2~from~the~menu~below
1:~inbuilt~sort()
2:~user~defined~sorting~logic()
2
[apple, bat, cat, goat, money, rose, wolf]

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To write program for linear search and binary search

Input:

```
import java.util.Scanner;
public class LinearSearch
{
    public static void main(String[] args)
    {
        int c,n,search,array[];
        Scanner in=new Scanner(System.in);
        System.out.println("Enter the number of elements:");
        n=in.nextInt();
        array=new int[n];
        System.out.println("Enter those "+n+" elements");
        for(c=0;c<n;c++)
            array[c]=in.nextInt();
        System.out.println("Enter value to find");
        search=in.nextInt();
        for(c=0;c<n;c++)
            if(array[c]==search)
            {
                System.out.println(search + "is present at location"  +(c+1));
                break;
            }
        if(c==n)
            System.out.println(search+ "isn't present in array");
    }
}
```


Output:

```
Enter the number of elements:
5
Enter those 5 elements
56
35
12
45
23
Enter value to find
35
35 is present at location 2
```

BINARY SEARCH

```
import java.util.Arrays;
import java.util.Scanner;
public class BinarySearch
{
    public static int myBinarySearch(int arr[],int first,int last,int key)
    {
        int mid=(first + last)/2;
        while(first<=last)
        {
            if(arr[mid]<key)
            {
                first=mid + 1;
            }
            else if(arr[mid]==key)
            {
                System.out.println("Element is found at index:" +mid);
                return mid;
            }
            else
            {
                last=mid-1;
            }
            mid=(first + last)/2;
        }
        return -1;
    }
    public static int binarysearchRecursion(int arr[],int first,int last,int key)
    {
        if(last>=first)
        {
            int mid=first + (last-first)/2;
            if(arr[mid]==key)
            {
```

```

return mid;
}
if(arr[mid]>key)
{
return binarysearchRecursion(arr,first,mid-1,key);
}
else
{
return binarysearchRecursion(arr,mid+1,last,key);
}
}
return -1;
}
public static void main(String[] args)
{
System.out.println("Enter your choice of binary search logic:");
System.out.println("1.Simple binary search");
System.out.println("2.Recursive binary search");
System.out.println("3:Using arrays binarySearch");
Scanner sc=new Scanner(System.in);
int ch=sc.nextInt();
Scanner in=new Scanner(System.in);
System.out.println("Enter number of elements:");
int n=in.nextInt();
int array[]=new int[n];
System.out.println("Enter those " +n+ " elements:");
for(int c=0;c<n;c++)
array[c]=in.nextInt();
System.out.println("Enter value to find:");
int search=in.nextInt();
Arrays.sort(array);
int found=-1;
switch(ch)
{
case 1:found=myBinarySearch(array,0,n,search);
break;
case 2:found=binarysearchRecursion(array,0,n,search);
break;
case 3:found=Arrays.binarySearch(array,search);
break;
}
if(found>=0)
{
System.out.println("Elements is found at " +found);
}
else
{

```

```
System.out.println("Elements is not found !");  
}  
}  
}
```

Output:

```
Enter your choice of binary search logic:  
1.Simple binary search  
2.Recursive binary search  
3:Using arrays binarySearch  
1  
Enter number of elements:  
5  
Enter those 5 elements:  
56  
35  
12  
45  
23  
Enter value to find:  
35  
Element is found at index:2  
Elements is found at 2
```

```
Enter your choice of binary search logic:  
1.Simple binary search  
2.Recursive binary search  
3:Using arrays binarySearch  
2  
Enter number of elements:  
5  
Enter those 5 elements:  
56  
35  
12  
45  
23  
Enter value to find:  
35  
Elements is found at 2
```

```

Enter your choice of binary search logic:
1.Simple binary search
2.Recursive binary search
3:Using arrays binarySearch
3
Enter number of elements:
5
Enter those 5 elements:
56
35
12
45
23
Enter value to find:
45
Elements is found at 3

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To illustrate string manipulation methods.

Input:

```
public class StringManipulation
{
    public static void main(String[] args)
    {
        System.out.println("\n \n****Creating_of_Strings****");
        char arrSample[]={ 'R', 'O','S','E' };
        String strSample_1=new String(arrSample);
        System.out.println("\n Created_from_char[]_using_new_String:"+strSample_1);
        byte ascii[]={ 65,66,67,68,69,70 };
        String strSample_2=new String(ascii);
        System.out.println("\n Created from byte[]:"+strSample_2);

        System.out.println("\n****Getting_String_length****");
        System.out.println("\n Length of " +strSample_1+" is "+strSample_1.length());

        System.out.println("\n****String_concatenation****");
        String strSample_3=strSample_1.concat(strSample_2);
        System.out.println("\n using concat(): "+strSample_3);
        String strSample_4=strSample_1+strSample_2;
        System.out.println("\n using + operator: "+strSample_4);

        System.out.println("\n****String comparison****");
        System.out.println("\n \n #####USING COMPARETO#####");
        String str_Sample="RockStar";
        System.out.println("\n Compare 'RockStar' To
        'ROCKSTAR':"+str_Sample.compareTo("ROCKSTAR"));
        System.out.println("\n Compare 'RockStar' To 'ROCKSTAR' Case
        ignored:"+str_Sample.compareToIgnoreCase("ROCKSTAR"));

        System.out.println("\n \n #####USING EQUALS#####");
        System.out.println("\n 'RockStar' equals('ROCKSTAR')
        is:"+str_Sample.equals("ROCKSTAR"));
```

```

System.out.println("\n'RockStar' equals('ROCKSTAR')is if Case Ignored:"+str_Sample.equalsIgnoreCase("ROCKSTAR"));

System.out.println("\n \n #####USING INDEXOF#####");
System.out.println("\n indexof t in 'RockStar' is:"+str_Sample.indexOf("t"));
System.out.println("\n indexof 'Star' in 'RockStar' is:"+str_Sample.indexOf("Star"));

System.out.println("\n****Modifying a string****");
System.out.println("\n changing case of chaacters in the string");
System.out.println("\n All caps 'RockStar': "+str_Sample.toUpperCase());
System.out.println("\n All small 'RockStar':"+str_Sample.toLowerCase());

System.out.println("\n \n #####USING REPLACE#####");
System.out.println("\n In 'RockStar' replace 'Star' with 'et'+str_Sample.replace("Star","et"));
}
}

```

Output:

```

****Creating_of_Strings****
Created_from_char[]_using_new_String:ROSE
Created from byte[]:ABCDEF
****Getting_String_length****
Length of ROSE is 4
****String_concatenation****
using concat(): ROSEABCDEF
using + operator: ROSEABCDEF
****String comparison****

```

```
#####USING COMPARETO#####
Compare 'RockStar' To 'ROCKSTAR':32
Compare 'RockStar' To 'ROCKSTAR' Case ignored:0

#####USING EQUALS#####
'RockStar' equals('ROCKSTAR') is:false
'RockStar' equals('ROCKSTAR')is if Case Ignored:true

#####USING INDEXOF#####
indexof t in 'RockStar' is:5
indexof 'Star' in 'RockStar' is:4

***Modifying a string***
changing case of chaacters in the string
All caps 'RockStar': ROCKSTAR
All small 'RockStar':rockstar
```

```
#####USING REPLACE#####
In 'RockStar' replace 'Star' with 'et' Rocket
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to create a class for employee having attributes eNo, eName, and eSalary. Read n employee information and search for an employee given eNo using the concept of array of objects.

Input:

```
import java.util.Scanner;
import java.util.Arrays;
class Employee
{
    int eno,esalary;
    String ename;

    public Employee()
    {

    }
    public Employee(int no,int sal,String name)
    {
        eno=no;
        esalary=sal;
        ename=name;
    }
    public void showData()
    {
        System.out.println("EMPID="+eno+" "+"NAME="+ename+" "+"salary="+
        esalary);
        System.out.println();
    }
}
```



```

public class EmpArrObjects
{
public static void main(String[] args)
{
System.out.println("Enter no of employee");
Scanner sc=new Scanner(System.in);
int n=sc.nextInt();
System.out.println("Enter employee details one by one\n");
Employee employees[ ]=new Employee[n];
Scanner sc_emp=new Scanner(System.in);
int eid,esal;
String enam;
for(int i=0;i<n;i++)
{
System.out.println("Enter"+ i +"employee details\n");
System.out.println("Enter employee id(integer):\n");
eid=sc_emp.nextInt();
System.out.println("Enter employee name(String)\n");
String nam=sc_emp.next();
enam=new String(nam);
System.out.println("Enter employee salary(integer)\n");
esal=sc_emp.nextInt();
Employee emp=new Employee(eid,esal,enam);
employees[i]=emp;
}
System.out.println("Employee are:\n");
for(Employee y: employees)
y.showData();
System.out.println("Enter employee no to search:\n");
int semp=sc.nextInt();
boolean found=false;
for(Employee e:employees)
{
if(semp==e.eno)
{
found=true;
System.out.println("Employee found");
e.showData();
break;
}
}
}

```

```

    }
    if(!found)
    {
        System.out.println("employee not found");
    }
}
}

```

Output:

```

Enter no of employee
2
Enter employee details one by one

Enter0employee details

Enter employee id(integer):
001
Enter employee name(String)
shyam
Enter employee salary(integer)
30000

```

```

Enter1employee details

Enter employee id(integer):
002
Enter employee name(String)
Rahul
Enter employee salary(integer)
30000
Employee are:

EMPID=1NAME=shyamsalary=30000
EMPID=2NAME=Rahulsalary=30000

Enter employee no to search:
1
Employee found
EMPID=1NAME=shyamsalary=30000

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To find the area of different shape using overloaded function.

Input:

```
import java.util.Scanner;
public class shapearea
{
    void calculatearea(float a)
    {
        System.out.println("\n Area of the square="+a*a);
    }
    void calculatearea(int l,int b)
    {
        System.out.println("\n Area of the rectangle="+l*b);
    }
    void calculatearea(double r)
    {
        double area=3.14*r*r;
        System.out.println("\n Area of the circle="+area);
    }
    public static void main(String[] args)
    {
        shapearea obj=new shapearea();
        System.out.println("\n\nENTER SIDE OF SQUARE\n");
        Scanner sc=new Scanner(System.in);
        float side=sc.nextFloat();
        obj.calculatearea(side);

        System.out.println("\n\nENTER RADIUS\n");
        Scanner sc1=new Scanner(System.in);
        double rad=sc.nextDouble();
        obj.calculatearea(rad);

        System.out.println("\n\nENTER SIDE OF RECTANGLE\n");
        Scanner sc2=new Scanner(System.in);
```

```

int side1=sc.nextInt();
int side2=sc.nextInt();
obj.calculatearea(side1,side2);
}
}

```

Output:

```

ENTER SIDE OF SQUARE
4
Area of the square=16.0

ENTER RADIUS
3
Area of the circle=28.259999999999998

ENTER SIDE OF RECTANGLE
3
5
Area of the rectangle=15

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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, subject taught and constructors to initialize these data members and also include display function to display all the data members. Use array of object to display details of N teachers.

Input:

```
import java.util.Scanner;
class EmployeeT
{
int empid;
String name;
float salary;
String address;
EmployeeT()
{
}
EmployeeT(int empid,String name,float salary,String address)
{
this.empid=empid;
this.name=name;
this.salary=salary;
this.address=address;
}
}
class Teacher extends EmployeeT
{
String dept,sub;
Teacher(int empid,String name,float salary,String address,String dept,String sub)
{
super(empid,name,salary,address);
this.dept=dept;
this.sub=sub;
}
}
```

```

public void display()
{
System.out.println("\nTeacher id\n" +empid);
System.out.println("\nTeacher name\n" +name);
System.out.println("\nTeacher salary\n"+salary);
System.out.println("\nTeacher address\n" +address);
System.out.println("\nTeacher department\n" +dept);
System.out.println("\nTeacher subject\n"+sub);
}
}
public class Techarrays
{
public static void main(String args[])
{
System.out.println("Enter number of teachers");
Scanner sc=new Scanner(System.in);
int n=sc.nextInt();
System.out.println("Enter teacher details one by one");
Teacher teacher[]=new Teacher[n];
Scanner sc1=new Scanner(System.in);
int tid;
String name;
float salary;
String add;
String dept,sub;
for(int i=0;i<n;i++)
{
System.out.println("Enter teacher id");
tid=sc1.nextInt();
System.out.println("Enter teacher name");
name=sc1.next();
System.out.println("Enter teacher salary");
salary=sc1.nextFloat();
System.out.println("Enter teacher address");
add=sc1.next();
System.out.println("Enter teacher department");
dept=sc1.next();
System.out.println("Enter teacher subject");
sub=sc1.next();
Teacher t=new Teacher(tid,name,salary,add,dept,sub);
teacher[i]=t;
}
for(Teacher x:teacher)
{
x.display();
System.out.println("\n");
}
}

```

```
}  
}
```

Output:

```
Enter number of teachers  
2  
Enter teacher details one by one  
Enter teacher id  
1  
Enter teacher name  
Ammu  
Enter teacher salary  
30000  
Enter teacher address  
Ranni  
Enter teacher department  
MCA  
Enter teacher subject  
OB  
Enter teacher id  
2  
Enter teacher name  
Revathy  
Enter teacher salary  
30000  
Enter teacher address  
Kozhencherry  
Enter teacher department  
MCA  
Enter teacher subject  
OOPS Lab
```

```
Teacher id  
1  
Teacher name  
Ammu  
Teacher salary  
30000.0  
Teacher address  
Ranni  
Teacher department  
MCA  
Teacher subject  
OB
```



```
Teacher id
2
Teacher name
Revathy
Teacher salary
30000.0
Teacher address
Kozhencherry
Teacher department
MCA
Teacher subject
OOPS
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Create a class 'person' with data members Name, Gender, Age, Address and constructors to initialize the data members, create another class 'Employee' that inherit the properties of class person and contain 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 contain its own data members like subject, department, Teacher id and also contain constructor and methods to display the data members. Use array of objects to display details of N teachers.

Inputs:

```
import java.util.Scanner;
class Person
{
    String Name, Gender,Address;
    protected int Age;
    public Person()
    {

    }
    public Person (String n , String g, String addr, int a)
    {
        this.Name=n;
        this.Gender=g;
        this.Address =addr;
        this.Age=a;
    }
    public void displayPerson()
    {
        System.out.println("Name:___"+Name);
        System.out.println("Gender:___" + Gender);
        System.out.println("Address:___" + Address);
        System.out.println("Age:____"+Age);
    }
}
class Employee extends Person
{
```

```

int Empid,Salary;
String Company_name,Qualification;
public Employee()
{

}

public Employee (String n,String g , String addr, int a,int eid,String cname,String qual,
int sal)
{
super(n,g,addr,a);
Empid=eid;
Company_name=cname;
Qualification=qual;
Salary=sal;
}
public void displayEmployee()
{
super.displayPerson ();
System.out.println("Empid:____"+Empid);
System.out.println("Company name:____" +Company_name);
System.out.println("Qualification :____"+Qualification);
System.out.println("Salary:____"+ Salary);
}
}
class Teacher1 extends Employee
{
String Subject,Department;
int Teacherid;
public Teacher1(String n, String g, String addr, int a,int eid, String cname, String qual,
int sal ,String sub, String dept,int tid)
{
super(n,g,addr,a, eid, cname, qual, sal );
Subject=sub;
Department=dept;
Teacherid=tid;
}
public void displayTeacher()
{
super.displayEmployee();
System.out.println ("Teacher_id____"+Teacherid);
System.out.println("Subject :____" + Subject);
System.out.println ("Department:____"+ Department);
}
}
public class InheritancePersonExample
{
public static void main(String args[])

```

```

{
System.out.println("Enter number of teachers" );
Scanner sc = new Scanner (System.in);
int N= sc.nextInt ();
Teacher1[] teacherls = new Teacher1[N];
Scanner scs = new Scanner(System.in);
for (int i = 0; i<N;i++)
{
System.out.println("Enter_name_of_the_teacher.");
String name=scs.next();
System.out.println("Enter_gender of the teacher.");
String gen=scs.next();
System.out.println("Enter address of the teacher.");
String addr = scs.next();
System.out.println("Enter age of the teacher.");
int ag = sc.nextInt ();
System.out.println("Enter Empid_of_the_teacher.");
int eid=sc.nextInt ();
System.out.println("Enter Company name.");
String cn=scs.next();
System.out.println("Enter qualification of the teacher.");
String quali = scs. next();
System.out.println("Enter_salary of the teacher.");
int sal=sc. nextInt ();
System.out.println("Enter Teacher_id");
int tid=sc.nextInt ();
System.out.println("Enter Subject of the teacher.");
String sub=scs.next();
System.out.println("Enter Department of the teacher.");
String dept = scs.next();
Teacher1 t= new Teacher1 (name, gen, addr, ag, eid, cn, quali,sal,sub, dept, tid);
teacherls[i] = t;
}
for (Teacher1 t:teacherls)
{
t. displayTeacher();
}
}
}

```

Output:

```
Enter number of teachers
2
Enter_name_of_the_teacher.
Renukha
Enter_gender of the teacher.
Female
Enter address of the teacher.
Kozhencherry
Enter age of the teacher.
29
Enter Empid_of_the_teacher.
1
Enter Company name.
RD
Enter qualification of the teacher.
MCA
Enter_salary of the teacher.
30000
Enter Teacher_id
1
Enter Subject of the teacher.
OB
Enter Department of the teacher.
MCA
```

```
Enter_name_of_the_teacher.
Manu
Enter_gender of the teacher.
Male
Enter address of the teacher.
Ranni
Enter age of the teacher.
35
Enter Empid_of_the_teacher.
2
Enter Company name.
DR
Enter qualification of the teacher.
MCA
Enter_salary of the teacher.
30000
Enter Teacher_id
2
Enter Subject of the teacher.
OOPS
Enter Department of the teacher.
MCA
```

```

Name: __Renukha
Gender: __Female
Address: __Kozhencherry
Age: __29
Empid: __1
Company name: __RD
Qualification : __MCA
Salary: __30000
Teacher_id __1
Subject : __OB
Department: __MCA
Name: __Manu
Gender: __Male
Address: __Ranni
Age: __35
Empid: __2
Company name: __DR
Qualification : __MCA
Salary: __30000
Teacher_id __2
Subject : __OOPS
Department: __MCA

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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.

Input:

```
class Publisher
{
String publisher;
Publisher (String publi)
{
this.publisher=publi;
}
}

class Book
{
String name;
Publisher publisher;
Book (){}
public Book(String name,Publisher publisher)
{
this.name = name;
this.publisher = publisher;
}
}

class Literature extends Book
{
String Littype = "Literature";
Literature (String name, Publisher publisher)
{
super (name, publisher);
}

void display ()
{
System.out.println ("\nName:" + super.name);
```

```

System.out.println("\nType: " + this.Litttype);
System.out.println("\nPublisher:" + this.publisher.publisher);
}
}

```

```

class Fiction extends Book
{
String Litttype="Fiction";
Fiction (String name, Publisher publisher)
{
super (name, publisher);
}
}

```

```

void display ()
{
System.out.println ("\nName:"+super.name);
System.out.println("\nType: " + this.Litttype);
System.out.println("\nPublisher:" + this.publisher.publisher);
}
}
public class InheritanceBookExample
{
public static void main(String[] args)
{
Publisher lp= new Publisher ("S.Chand" );
Literature l = new Literature ("As you like it",lp);
l.display ();
Publisher fp = new Publisher ("Tata McGraw Hill");
Fiction f = new Fiction ("Tempest",fp);
f.display();
}
}

```

Output:

```

Name:As you like it
Type: Literature
Publisher:S.Chand
Name:Tempest
Type: Fiction
Publisher:Tata McGraw Hill

```


Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
interface student
{
int score=10;
void displayscore();
}
interface sports
{
int score=25;
void displaysports();
}
class result implements student,sports
{
public void displayscore()
{
System.out.println("Academic score is "+student.score);
}
public void displaysports()
{
System.out.println("Sports score is "+sports.score);
}
}
public class Sportsresult
{
public static void main(String[] args)
{
result r=new result();
r.displayscore();
r.displaysports();
}
}
```

Output:

```
Academic score is 10
Sports score is 25
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To create a graphics package that has classes and interfaces for figures Triangle, Square, and Circle and test the package by finding the area of these figures.

Input:

```
package org.shape;
public class Square
{
    private int side;
    public Square(int s)
    {
        side=s;
    }
    public int area( )
    {
        return(side*side);
    }
}
```

```
package org.shape;
public class Triangle
{
    private int side1,side2,side3;
    public Triangle(int s1,int s2, int s3)
    {
        side1=s1;
        side2=s2;
        side3=s3;
    }
    public double area( )
    {
        double s=( side1+side2+side3)/2;
        double a=Math.sqrt((s-side1 )+( s-side2)+( s-side3));
        return a;
    }
}
```

```

package org.shape;
public class Circle
{
private int radius;
public Circle(int r)
{
radius=r;
}
public double area( )
{
return(3.14*radius*radius);
}
}

```

```

import org.shape.*;
import java.util.*;
class TestPackage
{
public static void main(String[ ] args)
{
Scanner sc=new Scanner(System.in);
System.out.println("Enter the side of the Square : ");
int s=sc.nextInt();
Square sq=new Square(s);
System.out.println("Area of Square is "+ sq . area( ));
System.out.println( " Enter the radius of the Circle : " );
int r=sc.nextInt();
Circle ci=new Circle(s);
System.out.println( "Area of Circle is "+ ci . area());
System.out.println( "Enter the Side1 of the Triangle :");
int s1=sc.nextInt();
System.out.println( "Enter the Side2 of the Triangle: ");
int s2=sc.nextInt();
System.out.println("Enter the Side3 of the Triangle : ");
int s3=sc.nextInt();
Triangle t=new Triangle(s1,s2,s3);
System.out.println("Area of T riangle is "+ t .area( ));
}
}

```

Output:

```
Enter the side of the Square :  
3  
Area of Square is 9  
Enter the radius of the Circle :  
4  
Area of Circle is 28.259999999999998  
Enter the Side1 of the Triangle :  
4  
Enter the Side2 of the Triangle:  
3  
Enter the Side3 of the Triangle :  
6  
Area of T riangle is 2.23606797749979
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To create an arithmetic package that has classes and interfaces for the 4 basic arithmetic operations and test the packages by implementing all operations on two given numbers.

Input:

```
package org.calc;
public class Add
{
    private int x,y;
    public Add(int a,int b)
    {
        x=a;
        y=b;
    }
    public int add()
    {
        return(x+y);
    }
}
```

```
package org.calc;
public class Divide
{
    private int x,y;
    public Divide(int a,int b)
    {
        x=a;
        y=b;
    }
    public int div()
    {
        return(x/y);
    }
}
```

```

package org.calc;
public class Multiply
{
    private int x,y;
    public Multiply(int a,int b)
    {
        x=a;
        y=b;
    }
    public int mul()
    {
        return(x*y);
    }
}

```

```

package org.calc;
public class Sub
{
    private int x,y;
    public Sub(int a,int b)
    {
        x=a;
        y=b;
    }
    public int sub()
    {
        return(x-y);
    }
}

```

```

import org.calc.*;
import java.util.*;
public class Test
{
    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);

        System.out.println("Enter the number,a:");
        int a=sc.nextInt();

        System.out.println("Enter the number,b:");
        int b=sc.nextInt();
    }
}

```



```

Add add=new Add(a,b);
System.out.println("Addition:a+b="+add.add());

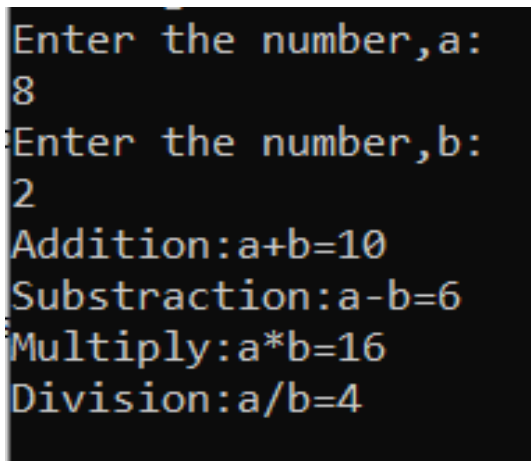
Sub s=new Sub(a,b);
System.out.println("Substraction:a-b="+s.sub());

Multiply m=new Multiply(a,b);
System.out.println("Multiply:a*b="+m.mul());

Divide d=new Divide(a,b);
System.out.println("Division:a/b="+d.div());
}
}

```

Output:



```

Enter the number,a:
8
Enter the number,b:
2
Addition:a+b=10
Substraction:a-b=6
Multiply:a*b=16
Division:a/b=4

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
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:

```

Enter username: : Raj
Enter password : : hello
UsernameException: Username must be greater than 6 characters ???
    at CheckLoginCredential.main(CheckLoginCredential.java:30)
The finally statement is executed

```

```

Enter username: : Sangram
Enter password : : 123
PasswordException: Incorrect password
Type correct password ???
    at CheckLoginCredential.main(CheckLoginCredential.java:32)
The finally statement is executed

```

```

Enter username: : Sangram
Enter password : : hello
Login Successful !!!
The finally statement is executed

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To find the average of N positive integers, raising a user defined exception for each negative inputs.

Input:

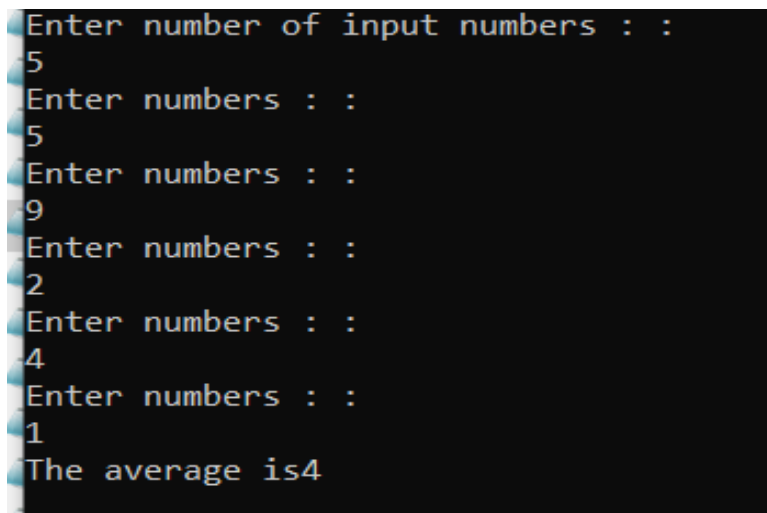
```
import java.io.IOException;
import java.util.Scanner;
class MyException extends Exception
{
    public MyException (String str)
    {
        System.out.println(str);
    }
}
public class SignException
{
    public static void main(String[ ] args)throws IOException
    {
        System.out.println("Enter number of input numbers : :");
        Scanner sc = new Scanner(System.in);
        int n =sc.nextInt( );
        int k =0, sum =0;
        Integer mynumbers[ ] = new Integer[n];
        while(n>0)
        {
            try
            {
                System.out.println("Enter numbers : : ");
                int num = sc.nextInt( );
                if(num<0)
                    throw new MyException ("Number is negative");
                else
                {
                    mynumbers[k]=num;
                    sum=sum+num;
                }
            }
            catch (MyException e)
            {
                System.out.println(e.getMessage());
            }
        }
    }
}
```

```

        k++;
    }
    n--;
}
catch(MyException m)
{
    System.out.println(m);
}
}
System.out.println("The average is"+ sum/k);
}
}

```

Output:



```

Enter number of input numbers : :
5
Enter numbers : :
5
Enter numbers : :
9
Enter numbers : :
2
Enter numbers : :
4
Enter numbers : :
1
The average is4

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To define 2 classes; one for generating multiplication table of 5 and other for displaying first N prime numbers. Implement using threads.

Input:

```
class Multiple5 extends Thread
{
    public void run( )
    {
        int num=5;
        for(int i=1;i<=10;++i)
            System.out.printf("Thread1-Table5 :%d * %d = %d \n", num , i , num * i);
    }
}
class PrimeN extends Thread
{
    public void run( )
    {
        int ct =0, n=0, i =1, j =1;
        while(n<5)
        {
            j =1; ct =0;
            while (j<=i )
            {
                if(i%j ==0)
                    ct++; j ++;
            }
            if(ct==2)
            {
                System.out.printf("Thread2-Prime : %d \n", i) ;
                n++;
            }
            i++;
        }
    }
}
```

```

    }
    }
    public class TreadClassExp
    {
    public static void main(String[ ] args)
    {
    Multiple5 m5 = new Multiple5( );
    PrimeN pn = new PrimeN( );
    pn.start( );
    m5.start( );
    }
    }

```

Output:

```

Thread2-Prime : 2
Thread2-Prime : 3
Thread2-Prime : 5
Thread2-Prime : 7
Thread2-Prime : 11
Thread1-Table5 :5 * 1 = 5
Thread1-Table5 :5 * 2 = 10
Thread1-Table5 :5 * 3 = 15
Thread1-Table5 :5 * 4 = 20
Thread1-Table5 :5 * 5 = 25
Thread1-Table5 :5 * 6 = 30
Thread1-Table5 :5 * 7 = 35
Thread1-Table5 :5 * 8 = 40
Thread1-Table5 :5 * 9 = 45
Thread1-Table5 :5 * 10 = 50

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using thread.

Input:

```
class FibThread implements Runnable
{
public void run( )
{
int a=0, b=1, c=0;
System.out.println("FibThread-"+ a);
System.out.println("FibThread- "+ b);
for(int h = 1 ; h<=7; h++)
{
c = a + b ;
System.out.println("FibThread - "+c);
a = b ;
b = c ;
}
}
}
class EvenRangeThread implements Runnable
{
public void run( )
{
int a = 2 , b = 10;
for(int k = a ; k<=b ; k+=2)
System.out.println("EvenRangeThread - " + k);
}
}
public class FibEven
{
public static void main (String args [ ] )
{
FibThread ft = new FibThread ( ) ;
```

```

EvenRangeThread er = new EvenRangeThread( );
Thread t1 = new Thread(ft);
Thread t2 = new Thread(er);
t1.start();
t2.start();
}
}

```

Output:

```

FibThread-0
EvenRangeThread - 2
EvenRangeThread - 4
FibThread- 1
EvenRangeThread - 6
EvenRangeThread - 8
EvenRangeThread - 10
FibThread - 1
FibThread - 2
FibThread - 3
FibThread - 5
FibThread - 8
FibThread - 13
FibThread - 21

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To implement producer consumer problem using ITC. This program uses.

Input:

```
import java.util.LinkedList;
public class ProConITC
{
    public static void main(String[ ] args )
    throws InterruptedException
    {
        final PC pc = new PC( );
        Thread t1 = new Thread(new Runnable( )
        {
            public void run ( )
            {
                try
                {
                    pc.produce( );
                }
                catch(InterruptedException e)
                {
                    e.printStackTrace( );
                }
            }
        } ) ;
        Thread t2 =new Thread(new Runnable( )
        {
            public void run( )
            {
                try
                {
                    pc.consume( );
                }
                catch(InterruptedException e)
                {
                    e.printStackTrace( );
                }
            }
        } ) ;
```

```

    }
    } );
    t1.start();
    t2.start();
    t1.join();
    t2.join();
}
public static class PC
{
    LinkedList<Integer>list=new LinkedList<>();
    int capacity=2;
    public void produce()throws InterruptedException
    {
        int value=0;
        while(true)
        {
            synchronized(this)
            {
                while(list.size() == capacity)
                wait();
                System.out.println("Producer produced-"+ value);
                list.add(value++);
                notify();
                Thread.sleep(1000);
            }
        }
    }
    public void consume()throws InterruptedException
    {
        while(true)
        {
            synchronized(this)
            {
                while(list.size() == 0)
                wait();
                int val = list.removeFirst();
                System.out.println("Consumer consumed-"+ val);
                notify();
                Thread.sleep(1000);
            }
        }
    }
}

```

Output:

```
Producer produced-0
Consumer consumed-0
Producer produced-1
Consumer consumed-1
Producer produced-2
Producer produced-3
Consumer consumed-2
Producer produced-4
Consumer consumed-3
Consumer consumed-4
Producer produced-5
Producer produced-6
Consumer consumed-5
Consumer consumed-6
Producer produced-7
Consumer consumed-7
Producer produced-8
Consumer consumed-8
Producer produced-9
Consumer consumed-9
Producer produced-10
Consumer consumed-10
Producer produced-11
Consumer consumed-11
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To write a program to create a generic stack and do the push and pop operations.

Input:

```
import java.io.*;
import java.util.*;
class stack<T>
{
    ArrayList<T>A;
    int top=-1;
    int size;
    stack(int size )
    {
        this.size=size ;
        this.A = new ArrayList<T>(size);
    }
    void push(T X)
    {
        if(top+1==size)
        {
            System.out.println("Stack Overflow");
        }
        else
        {
            top=top+1;
            if(A.size( )>top)
                A.set(top,X);
            else
                A.add(X);
        }
    }
    T top( )
    {
        if(top== -1)
        {
```

```

System.out.println("Stack Underflow");
return null;
}
else
return A.get(top);
}
void pop ( )
{
if(top== -1)
{
System.out.println("Stack Underflow");
}
else
top--;
}
boolean empty ( )
{
return top== -1;
}
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;
}
}
public class GenericStack
{
public static void main(String [ ] args)
{
stack<Integer> s1 = new stack<>(3);
s1.push(10);
s1.push(20);
s1.push(30);
System.out.println ("s1 after pushing 10,20 and 30 : \n" +s1 );
s1.pop();
System.out.println ("s1 after pop :\n" +s1);
stack<String> s2= new stack<>(3);
s2.push("hello");
s2.push("world");
s2.push("java");
System.out.println("\n s2 after pushing 3 elements : \n" +s2);
System.out.println("s2 after pushing 4th element : ");
}
}

```

```

s2.push("GFG");
stack<Float> s3 = new stack<>(2);
s3.push(100.0f);
s3.push(200.0f);
System.out.println("\n s3 after pushing 2 elements : \n" +s3);
System.out.println("top element of s3:\n"+ s3.top( ));
}
}

```

Output:

```

s1 after pushing 10,20 and 30 :
10â??->20â??->30
s1 after pop :
10â??->20

s2 after pushing 3 elements :
helloâ??->worldâ??->java
s2 after pushing 4th element :
Stack Overflow

s3 after pushing 2 elements :
100.0â??->200.0
top element of s3:
200.0

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To implement a generic method for bubble sorting.

Input:

```
import java.util.Arrays;
public class BubbleSortGeneric
<T extends Comparable<? super T>>
{
    T[] array;
    BubbleSortGeneric(T[] array)
    {
        this.array=array;
    }
    private T[]bubbleSort()
    {
        for(int i=array.length;i>1;i--)
        {
            for(int j=0 ;j<i-1;j++)
            {
                if(array[j].compareTo(array[j+1])>0)
                {
                    swapElements(j , array);
                }
            }
        }
        return array;
    }
    private void swapElements ( int index , T[] arr)
    {
        T temp = arr[index];
        arr[index]= arr[index+1];
        arr[index+1]=temp;
    }
    public static void main(String[] args)
    {
```

```

Integer[] intArr = {56,23,14,65,12,8,34,10,-5,-2};
BubbleSortGeneric<Integer>bsg1 = new BubbleSortGeneric<Integer>(intArr);
Integer[] sa1=bsg1.bubbleSort();
System.out.println("Sorted array:" +Arrays.toString(sa1));

String[] strArr={"hello","good","spell","write","sort"};
BubbleSortGeneric<String>bsg2 = new BubbleSortGeneric<>(strArr);
String[] sa2 = bsg2.bubbleSort();
System.out.println("Sorted array:"+Arrays.toString(sa2));
}
}

```

Output:

```

Sorted array: [-5, -2, 8, 10, 12, 14, 23, 34, 56, 65]
Sorted array: [good, hello, sort, spell, write]

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: To maintain a list of strings using Array list from collection framework, and perform built-in operations.

Input:

```
import java.util.*;
public class ArrayListExp
{
    public static void main(String args[])
    {
        ArrayList<String>list=new ArrayList<String>( );
        list.add("help");
        list.add("welcome");
        list.add("do");
        list.add("sleep");
        list.add("beep");
        System.out.println(list);
        System.out.println("Returning element:"+list.get(1));
        list.set(1,"newly inserted");
        System.out.println("List after insertion of:newly inserted");
        for(String word:list)
        System.out.println(word);
        Collections.sort(list);
        System.out.println("\nSorted list:");
        for(String word:list)
        System.out.println(word);
    }
}
```

Output:

```
[help, welcome, do, sleep, beep]
Returning element:welcome
List after insertion of:newly inserted
help
newly inserted
do
sleep
beep

Sorted list:
beep
do
help
newly inserted
sleep
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to remove all the elements from a linked list.

Input:

```
import java.io.*;
import java.util.LinkedList;
public class RemoveElementsLinkedList
{
    public static void main(String args[])
    {
        LinkedList<String>list=new LinkedList<String>();
        list.add("Good");
        list.add("Morning");
        list.add("have");
        list.add("a" );
        list.add("great day");
        list.add("2");
        list.add("day");
        System.out.println(" Original LinkedList: " + list);
        list.clear();
        System.out.println("List after clearing all elements : "+ list);
        list.add("looks");
        list.add("good");
        System.out.println(" After adding elements to empty list : " + list);
    }
}
```

Output:

```
Original LinkedList: [Good, Morning, have, a, great day, 2, day]
List after clearing all elements : []
After adding elements to empty list : [looks, good]
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to remove an object from the stack when the position is passed as parameter.

Input:

```
import java.util.*;
public class RemoveStackEle
{
    public static void main(String args[])
    {
        Stack<Integer>stack = new Stack<Integer>();
        stack.add(-4);
        stack.add(6);
        stack.add(16);
        stack.add(46);
        stack.add(66);
        stack.add(88);
        System.out.println("Stack:" +stack);
        int rem_ele =stack.remove(5);
        System.out.println("Removed element:"+ rem_ele);
        System.out.println("Final Stack :"+ stack);
    }
}
```

Output:

```
Stack:[-4, 6, 16, 46, 66, 88]
Removed element:88
Final Stack :[-4, 6, 16, 46, 66]
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to demonstrate the creation of queue object using the priorityqueue class.

Input:

```
import java.util.PriorityQueue;
import java.util.Queue;
public class QPriorityQ
{
    public static void main(String args[])
    {
        Queue<String> pq = new PriorityQueue<>();
        pq.add("Welcome");
        pq.add("have");
        pq.add("your");
        pq.add("seat");
        System.out.println("Original Queue : " +pq);
        pq.remove("your");
        System.out.println("After Remove"+ pq);
        System.out.println("Poll Method " + pq.poll());
        System.out.println("Final Queue" +pq);
        System.out.println(pq.peek());
    }
}
```

Output:

```
Original Queue : [Welcome, have, your, seat]
After Remove[Welcome, have, seat]
Poll Method Welcome
Final Queue[have, seat]
have
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

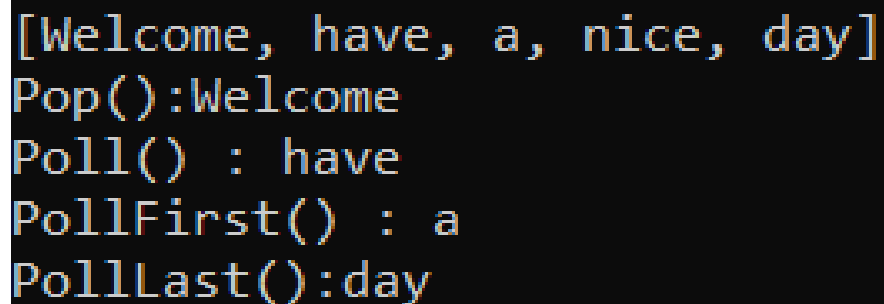
Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
import java.util.ArrayDeque;
import java.util.Deque;
public class DequeOperations
{
    public static void main(String[] args)
    {
        Deque<String>dq= new ArrayDeque<String>();
        dq.add("have");
        dq.addFirst("Welcome");
        dq.add("a");
        dq.add("nice");
        dq.addLast("day");
        System.out.println(dq);
        System.out.println("Pop():"+dq.pop());
        System.out.println("Poll() : " +dq.poll());
        System.out.println("PollFirst() : " +dq.pollFirst());
        System.out.println("PollLast(): " +dq.pollLast());
    }
}
```

Output:

A screenshot of a terminal window showing the output of the Java program. The output consists of five lines: the first line shows the deque contents as "[Welcome, have, a, nice, day]", followed by four lines showing the results of pop, poll, pollFirst, and pollLast operations, which are "Pop():Welcome", "Poll() : have", "PollFirst() : a", and "PollLast():day".

```
[Welcome, have, a, nice, day]
Pop():Welcome
Poll() : have
PollFirst() : a
PollLast():day
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to demonstrate the creation of set object using the linked Hash set class.

Input:

```
import java.io.*;
import java.util.*;
public class SetLinkedHashSet
{
    public static void main(String[] args)
    {
        Set<String>hs = new LinkedHashSet<String>();
        hs.add("hello");
        hs.add("what");
        hs.add("a");
        hs.add("nice");
        hs.add("sunny");
        hs.add("day");
        System.out.println(" Original Set :");
        for(String s:hs )
            System.out.print(s+ " , " );
        System.out.println();
        hs.clear();
        System.out.println(" Set after clear(): " +hs);
    }
}
```

Output:

```
Original Set :
hello , what , a , nice , sunny , day ,
Set after clear(): []
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Write a program to compare two hash sets.

Input:

```
import java.util.HashSet;
public class HashSetComp
{
    public static void main(String[] args)
    {
        HashSet<Integer>hSet1 = new HashSet<Integer>();
        hSet1.add(1);
        hSet1.add(2);
        hSet1.add(3);
        System.out.println("First Hash Set : " + hSet1);
        HashSet<Integer>hSet2 = new HashSet<Integer>();
        hSet2.add(3);
        hSet2.add(2);
        hSet2.add(1);
        System.out.println("Second Hash Set : "+ hSet2);
        System.out.println("hSet1 . equals ( hSet2 )?" + hSet1.equals(hSet2));
        hSet2.remove(2);
        System.out.println("Second Hash Set after removing an element :"+ hSet2);
        System.out.println("hSet1.equals ( hSet2 )? " + hSet1.equals (hSet2));
    }
}
```

Output:

```
First Hash Set : [1, 2, 3]
Second Hash Set : [1, 2, 3]
hSet1 . equals ( hSet2 )?true
Second Hash Set after removing an element :[1, 3]
hSet1.equals ( hSet2 )? false
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
import java.awt.*;
import java.util.*;
public class IllustrationMap
{
    public static void main (String[] args)
    {
        Map<String,Integer>map=new HashMap<>();
        map.put("orange",10);
        map.put("onion",30);
        map.put("apples",20);
        System.out.println("Original Map: ");
        for(Map.Entry<String , Integer>e:map.entrySet())
            System.out.println(e.getKey()+ ""+ e.getValue());
        map.remove("onion");
        System.out.println("Map after removing onion : " );
        for(Map.Entry<String, Integer>e:map.entrySet())
            System.out.println(e.getKey()+ ""+ e.getValue());
        map.replace("apples" , 500);
        System.out.println("Map after changing value of apples : ");
        for(Map. Entry<String , Integer>e:map.entrySet())
            System.out.println(e.getKey()+ ""+ e .getValue());
    }
}
```

Output:

```
Original Map:
orange10
onion30
apples20
Map after removing onion :
orange10
apples20
Map after changing value of apples :
orange10
apples500
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

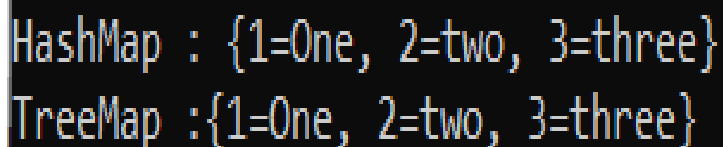
Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
import java.util.*;
import java.util.stream.*;
public class HashMap2TreeMap
{
    public static <K, V> Map<K, V> convertToTreeMap (Map<K, V> hashMap )
    {
        Map<K, V>
        treeMap = hashMap;
        return treeMap;
    }
    public static void main(String args[])
    {
        Map<String ,String>hashMap = new HashMap<>();
        hashMap.put("1","One") ;
        hashMap.put("2","two") ;
        hashMap.put("3","three") ;
        System.out.println("HashMap : " + hashMap ) ;
        Map<String,String>treeMap = convertToTreeMap(hashMap ) ;
        System.out.println("TreeMap : " + treeMap ) ;
    }
}
```

Output:



```
HashMap : {1=One, 2=two, 3=three}
TreeMap :{1=One, 2=two, 3=three}
```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

Aim: Program to implement a simple calculator using AWT components.

Input:

```
import java.awt.*;
import java.awt.event.*;
class SimpleCalculator extends Frame implements ActionListener {
    Frame f1=new Frame();
    Label l1=new Label("Enter First Num");
    Label l2=new Label("Enter Second Num");
    Label l3=new Label("Result");
    TextField t1=new TextField();
    TextField t2=new TextField();
    TextField t3=new TextField();
    Button b1=new Button("Add");
    Button b2=new Button("Sub");
    Button b3=new Button("Mul");
    Button b4=new Button("Div");
    Button b5=new Button("Cancel");
    SimpleCalculator()
    {
        l1.setBounds(50,100,100,20);
        l2.setBounds(50,140,100,20);
        l3.setBounds(50,180,100,20);
        t1.setBounds(200,100,100,20);
        t2.setBounds(200,140,100,20);
        t3.setBounds(200,180,100,20);
        b1.setBounds(50,250,50,20);
        b2.setBounds(110,250,50,20);
        b3.setBounds(170,250,50,20);
        b4.setBounds(230,250,50,20);
        b5.setBounds(290,250,50,20);
        b1.setBackground(Color.red);
        b2.setBackground(Color.green);
        b3.setBackground(Color.red);
        b4.setBackground(Color.green);
```

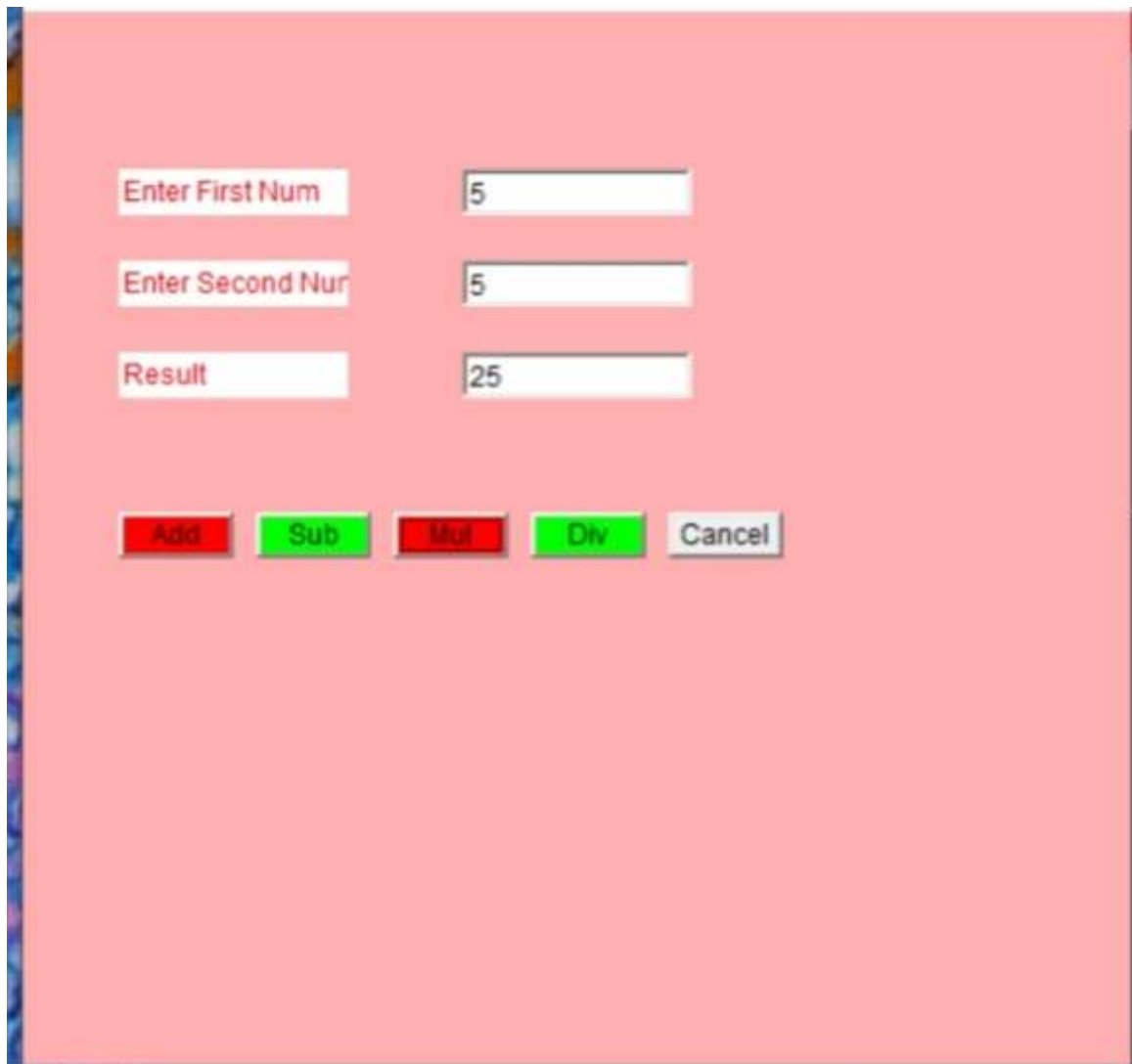
```

f1.add(l1);
f1.add(l2);
f1.add(l3);
f1.add(t1);
f1.add(t2);
f1.add(t3);
f1.add(b1);
f1.add(b2);
f1.add(b3);
f1.add(b4);
f1.add(b5);
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
f1.setLayout(null);
f1.setVisible(true);
f1.setSize(500,500);
l1.setForeground(Color.red);
l2.setForeground(Color.red);
l3.setForeground(Color.red);
f1.setBackground(Color.pink);
}
public void actionPerformed(ActionEvent e)
{
int n1=Integer.parseInt(t1.getText());
int n2=Integer.parseInt(t2.getText());
if(e.getSource()==b1)
{
t3.setText(String.valueOf(n1+n2));
}
if(e.getSource()==b2)
{
t3.setText(String.valueOf(n1-n2));
}
if(e.getSource()==b3)
{
t3.setText(String.valueOf(n1*n2));
}
if(e.getSource()==b4)
{
t3.setText(String.valueOf(n1/n2));
}
if(e.getSource()==b5)
{
System.exit(0);
}
}

```

```
}  
}  
public static void main(String args[])  
{  
    new SimpleCalculator();  
}  
}
```

Output:



The screenshot shows a Java Swing window titled "Simple Calculator" with a light pink background. It contains three input fields with labels "Enter First Num", "Enter Second Nur", and "Result". The first two fields contain the value "5", and the "Result" field contains "25". Below the input fields is a row of five buttons: "Add" (red), "Sub" (green), "Mul" (red), "Div" (green), and "Cancel" (gray). The window has a standard Mac OS X title bar with red, yellow, and green buttons on the top left.

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor:

REPORT ON LABORATORY WORK

Name: Seena Ann Mathew	Roll No:26	Name of lab: Object Oriented Programming Language	Period:
Class: S2, MCA	Date:	Name of lab work: practical	Batch:2020- 2022

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

Input:

```
import java.awt.*;
import java.applet.*;
import java.io.*;
/*
<applet code="AppletMax3" width=500 height=500>
<param name="a" value="10">
<param name="b" value="20">
<param name="c" value="30">
</applet>
*/
public class AppletMax3 extends Applet
{
    int a;
    int b;
    int c;
    int d;
    String str;
    public void start()
    {
        String s1;
        s1 = getParameter("a");
        a = Integer.parseInt(s1);
        s1 = getParameter("b");
        b = Integer.parseInt(s1);
        s1 = getParameter("c");
        c = Integer.parseInt(s1);
    }
    public void paint(Graphics g)
    {
        if( a >= b && a >= c)
            d = a;
        else if (b >= a && b >= c)
            d=b;
        else
```

```

d=c;
g.setColor(Color.blue);
Font myFont = new Font("Courier", Font.BOLD,20);
g.setFont(myFont);
g.drawString("First Num is " + a, 100, 100);
g.drawString("Second Num is " + b, 100, 200);
g.drawString("Third Num is " + c, 100, 300);
g.drawString("Max of Three Numbers is " + d, 100, 400);
}
}

```

Output:

```

First Num is    10

Second Num is   20

Third Num is    30

Max of Three Numbers is  30

```

Result/observation: successfully completed the program and output is obtained

Marks:

Viva (5)	Performance (5)	Total (10)

Assessor: