



MADRAS INSTITUTE OF TECHNOLOGY
ANNA UNIVERSITY
DEPARTMENT OF INFORMATION TECHNOLOGY

IT5512 WEB TECHNOLOGY
LAB RECORD
REGULATION – 2019

NAME : Babu M

REG NO : 2022506095

DEPARTMENT : DEPARTMENT OF INFORMATION TECHNOLOGY

SUBJECT CODE : IT5512

SUBJECT TITLE : WEB TECHNOLOGIES LABORATORY

ANNA UNIVERSITY

**MADRAS INSTITUTE OF TECHNOLOGY CHROMPET,
CHENNAI-600 044.**

BONAFIDE CERTIFICATE

NAME : Babu M

SUBJECT CODE : IT5512

SUBJECT TITLE : WEB TECHNOLOGIES LABORATORY

REGISTER NO : 2022506095

Certified that the bonafide record of practical work done by V.Monisa in the Laboratory subject code IT5512 during the period AUGUST 2024 - NOVEMBER 2024

DATE:

COURSE-IN-CHARGE

Submitted for the Practical Examination held on

Examiners

1.

2.

TABLE OF CONTENT

EXP NO	DATE	NAME OF THE EXPERIMENT	PAGE NO	SIGNATURE
1	8/8/24	Basic java programs	5	
2	22/8/24	Java programs using arrays and list	20	
3	29/8/24	Java programs using classes and interface	35	
4	5/9/24	Java programs using exception handling and regular expression	56	
5	26/9/24	Java programs using multithreading	62	
6	3/10/24	I/O Streams and object serialization	75	
7	10/10/24	Java networking programs	79	

--	--	--	--	--

8	17/10/24	Java swing programs	88	
9	7/11/24	Java server based web applications	108	
10	24/10/2024	Web applications using JSP		
11		Java Server Faces		
12		Android Application		

EXP NO : 1	Basic java programs
DATE : 8/8/24	

1. AIM :

To write a program in java to display the number in reverse order.

SOURCE CODE :

```
import java.util.Scanner;
import java.lang.*;
class pl
{
    public static void main(String args[])
    {
        int n;
        Scanner sc =new Scanner(System.in);
        System.out.print("Enter a Number : ");
        n=sc.nextInt();
        for(int i=n;i>=0;i--)
        {
            System.out.print(i+" ");
        }
        System.out.println();
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p1.java
```

```
D:\MIT\sem5\webtech\RECORD>java p1
```

```
Enter a Number : 20
```

```
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
```

2. AIM :

To write a program in java to find the HCF (Highest Common Factor) of two numbers.

SOURCE CODE :

```
import java.util.Scanner;
import java.lang.*;
class p2
{
    static int gcd(int a,int b)
    {
        if(a==b)
            return a;
        if(a>b)
            return gcd(a-b,b);
        else
            return gcd(a,b-a);
    }
    public static void main(String args[])
    {
        int a,b;
        Scanner sc =new Scanner(System.in);
        System.out.print("Enter a Number 1 : ");
        a=sc.nextInt();
        System.out.print("Enter a Number 2 : ");
        b=sc.nextInt();
        System.out.println("GCD : "+gcd(a,b));
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p2.java
```

```
D:\MIT\sem5\webtech\RECORD>java p2
```

```
Enter a Number 1 : 10
```

```
Enter a Number 2 : 100
```

```
GCD : 10
```

3. AIM :

To write a java program to determine the number of multiples of 3 between 20 and 80.

SOURCE CODE :

```

import java.util.Scanner;
import java.lang.*;
class p3
{
    public static void main(String args[])
    {
        System.out.print("mULTIPLES OF 3 BTW 20 TO 80 :");
        for(int i=20;i<80;i++)
            if(i%3==0)
                System.out.print(i+" ");
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p3.java
D:\MIT\sem5\webtech\RECORD>java p3.java
mULTIPLES OF 3 BTW 20 TO 80 :21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78

```

4. AIM :

To write a java program to find the largest palindrome number less than 1000.

SOURCE CODE :

```

import java.util.Scanner;
import java.lang.*;
class p4
{
    public static void main(String args[])
    {
        System.out.print("Largest palindrome less than 1000 is :");
        for(int i=1000;i>=0;i--)
        {
            int t=0;
            int k=i;
            while(k!=0)
            {
                t=t*10+k%10;
                k/=10;
            }
            if(i==t)
            {
                System.out.print(i);
                break;
            }
        }
    }
}

```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p4.java
D:\MIT\sem5\webtech\RECORD>java p4
Largest palindrome less than 1000 is :999
```

5. AIM :

To write a java program to generate the first n terms of the Fibonacci sequence.

SOURCE CODE :

```
import java.util.Scanner;
import java.lang.*;
class p5
{
    public static void main(String args[])
    {
        int n;
        Scanner sc =new Scanner(System.in);
        System.out.print("Enter value of n :");
        n=sc.nextInt();
        int a=-1;
        int b=1;
        while(n!=-0)
        {
            n--;
            int c=a+b;
            System.out.print(c+" ");
            a=b;
            b=c;
        }
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>java p5
Enter value of n :10
0 1 1 2 3 5 8 13 21 34
```

6. AIM :

To write a java program to calculate the product of all odd numbers from 1 to 50.

SOURCE CODE :

```
import java.util.Scanner;
import java.math.BigInteger;
import java.lang.*;
class p6
{
    public static void main(String args[])
    {

        System.out.print("Product of odd nos from 1-50 : ");
        BigInteger p=new BigInteger("1");
        for(int i=1;i<50;i++)
            if(i%2==1)
                p=p.multiply(BigInteger.valueOf(i));
        System.out.print(p);
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p6.java
```

```
D:\MIT\sem5\webtech\RECORD>java p6
Product of odd nos from 1-50 : 58435841445947272053455474390625
```

7. AIM :

To create a BankAccount class with attributes like account number, balance, and account type. Implement methods for deposit, withdrawal, and checking balance.

SOURCE CODE :

```
import java.lang.*;
public class p8
{
    String acc="12039278920";
    double bal=0;
    String type="Savings";
    void deposit(double amt)
    {
        bal+=amt;}
    void withdrawl(double amt)
    {
        if(amt<=bal)
        {
            bal-=amt;
            System.out.println("Please Collect Your Money .");
        }
        else
        {
            System.out.println("Insufficient Balance .");
        }
    }
}
```

```

void checkBal()
{
    System.out.println("Balance : "+bal);}
public static void main(String args[])
{
    p8 o=new p8();
    o.deposit(1000);
    o.withdrawl(100);
    o.withdrawl(10000);
    o.checkBal();
}}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p8.java

D:\MIT\sem5\webtech\RECORD>java p8
Please Collect Your Money .
Insufficient Balance .
Balance : 900.0

```

8. AIM :

To create classes for different shapes like Circle, Rectangle, and Triangle. Each class should have necessary attributes and methods to calculate area and perimeter. Create objects of different shapes and perform calculations.

SOURCE CODE :

```

import java.lang.*;
class circle
{
    int r;
    circle(int r)
    {
        this.r=r;
    }
    void area()
    {
        System.out.println("Area(circle): "+3.14*r*r);
    }
    void perimeter()
    {
        System.out.println("Perimeter(circle) : "+2*3.14*r);
    }
}
class triangle
{
    int a,b,c;

```

```

triangle(int a,int b,int c)
{
    this.a=a;
    this.b=b;
    this.c=c;

}
void area()
{
    double s=(a+b+c)/2.0;
    double ar=Math.sqrt(s*(s-a)*(s-b)*(s-c));
    System.out.println("Area(triangle): "+ar);
}
void perimeter()
{
    System.out.println("Perimeter(triangle) : "+(a+b+c));
}
}

class Rectangle
{
    int l,b;
    Rectangle(int l,int b)
    {
        this.l=l;
        this.b=b;
    }
    void area()
    {
        System.out.println("Area(rectangle): "+l*b);
    }
    void perimeter()
    {
        System.out.println("Perimeter(rectangle) : "+2*(l+b));
    }
}

public class p9
{
    public static void main(String args[])
    {
        Rectangle a=new Rectangle(3,5);
        a.area();
        a.perimeter();
        circle b=new circle(3);
        b.area();
        b.perimeter();
        triangle c=new triangle(3,5,4);
        c.area();
        c.perimeter();
    }
}

```

```
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p9.java

D:\MIT\sem5\webtech\RECORD>java p9
Area(rectangle): 15
Perimeter(rectangle) : 16
Area(circle): 28.259999999999998
Perimeter(circle) : 18.84
Area(triangle): 6.0
Perimeter(triangle) : 12
```

9. AIM :

To create a Calculator class with methods for basic arithmetic operations.

SOURCE CODE :

```
import java.lang.*;
class calculator
{
    void add(int a,int b)
    {
        System.out.println("Sum("+a+"+"+b+") : "+(a+b));
    }
    void sub(int a,int b)
    {
        System.out.println("Difference("+a+"-"+b+") : "+(a-b));
    }
    void mul(int a,int b)
    {
        System.out.println("Product("+a+"*"+b+") : "+(a*b));
    }
    void div(int a,int b)
    {
        System.out.println("division("+a+"/"+b+") : "+(a/b));
    }
}
class p10
```

```

{
    public static void main(String args[])
    {
        calculator c=new calculator();
        c.add(3,5);
        c.sub(5,3);
        c.mul(3,4);
        c.div(10,2);
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p10.java

D:\MIT\sem5\webtech\RECORD>java p10
Sum(3+5) : 8
Difference(5-3) : 2
Product(3*4) : 12
division(10/2) : 5

```

10. AIM :

To create a Student class with attributes like name, roll number, and marks in different subjects. Implement methods to calculate total marks, percentage, and grade.

SOURCE CODE :

```

import java.util.Scanner;
import java.util.Scanner;
class Student
{
    int tot;
    double per;
    char grade;
    String name, roll;
    int mat, phy, chem;
    void tcalc()
    {
        tot = mat + phy + chem;
    }
    void perc()
    {
        tcalc();
        per = tot / 3.0;
    }
    void grad() {
        perc();
    }
}

```

```

    if (per >= 90)
        grade = 'O';
    else if (per >= 80)
        grade = 'A';
    else if (per >= 70)
        grade = 'B';
    else if (per >= 60)
        grade = 'C';
    else if (per >= 40)
        grade = 'D';
    else
        grade = 'F';
}
}
public class p7
{
    public static void main(String[] args) {
        Student a = new Student();
        a.name = "Bharath";
        a.roll = "2022506116";
        a.mat = 97;
        a.chem = 96;
        a.phy = 98;
        a.grad();
        System.out.printf("Total : %d\nPercentage : %.2f%%\nGrade : %c\n", a.tot, a.per, a.grade);
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p7.java

D:\MIT\sem5\webtech\RECORD>java p7
Total : 291
Percentage : 97.00%
Grade : O

```

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Thus, the basic java programs has been implemented successfully and the output has been verified.

EXP NO : 2	Java Programs Using Array and List
DATE : 22/8/24	

1. AIM :

To write a java program to copy a subset of array elements to another array.

SOURCE CODE :

```
import java.lang.*;
import java.util.*;
public class p1
{
    public static void main(String[] args)
    {
        Scanner sc =new Scanner(System.in);
        System.out.print("Enter length of array : ");
        int n;
        n=sc.nextInt();
        int ar[]=new int[n];
        System.out.print("Enter Elements of the array : ");
        for(int i=0;i<n;i++)
        {
            ar[i]=sc.nextInt();
        }
    }
}
```

```

    }
    int m,s;
    System.out.print("Enter length and Start position : ");
    m=sc.nextInt();
    s=sc.nextInt();
    int ar2[]=new int[m];
    for(int i=s;i<n && i-s<m;i++)
    {
        ar2[i-s]=ar[i];
    }
    System.out.print("Original Array : ");
    for(int i=0;i<n ;i++)
    {
        System.out.print(ar[i]+" ");
    }
    System.out.println();
    System.out.print("Subset  Array : ");
    for(int i=0;i<m ;i++)
    {
        System.out.print(ar2[i]+" ");
    }
    System.out.println();
}}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>java p1
Enter length of array : 5
Enter Elements of the array : 1
2
3
4
5
Enter length and Start position : 2 2
Original Array : 1 2 3 4 5
Subset  Array : 3 4

```

2. AIM :

To write a java program to find the largest and smallest element in an array.

SOURCE CODE :

```

import java.lang.*;
public class p2
{
    public static void main(String args[])
    {
        int ar[]={1,2,3,4,5,6,7,8,9,10};
    }
}

```



```

int mi=ar[0];
int ma=ar[0];
for(int i=1;i<10;i++)
{
    if(ar[i]>ma)
    {
        ma=ar[i];
    }
    if(ar[i]<mi)
    {
        mi=ar[i];
    }
}
System.out.println("Largest Element : "+ma);
System.out.println("Smallest Element : "+mi);
} }

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p2.java

D:\MIT\sem5\webtech\RECORD>java p2
Largest Element : 10
Smallest Element : 1

```

3. AIM :

To write a java program to find the frequency of each element in an array.

SOURCE CODE :

```

import java.lang.*;
import java.util.*;
public class p3
{
    public static void main(String[] args)
    {
        int ar[]={1,2,3,3,3,3,4,5,6,7,8,9,9,9,9,10,2,2};
        HashMap<Integer,Integer> m=new HashMap<>();
        for(int i=0;i<ar.length;i++)
        {
            if(m.containsKey(ar[i]))
            {
                m.put(ar[i],m.get(ar[i])+1);
            }
            else
            {
                m.put(ar[i], 1);
            }
        }
    }
}

```

```

    for(HashMap.Entry<Integer,Integer> e: m.entrySet())
    {
        System.out.println(e.getKey()+" : "+e.getValue());
    }
}
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p3.java

D:\MIT\sem5\webtech\RECORD>java p3
1 : 1
2 : 3
3 : 4
4 : 1
5 : 1
6 : 1
7 : 1
8 : 1
9 : 5
10 : 1

```

4. AIM :

To perform matrix addition ,multiplication,Transpose.

SOURCE CODE :

ADDITION:

```

import java.lang.*;
import java.util.*;
public class p41
{
    public static void main(String args[])
    {
        int n1,m1,n2,m2;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Order of Matrix 1: ");
        n1=sc.nextInt();
        m1=sc.nextInt();
        System.out.println("Enter Order of Matrix 2: ");
        n2=sc.nextInt();
    }
}

```

```

m2=sc.nextInt();
if(n1!=n2 || m1!=m2)
{
    System.out.println("Not Possible . ");
}
else
{
    int ar1[][]=new int[n1][m1];
    int ar2[][]=new int[n2][m2];
    int s[][]=new int[n1][m1];
    System.out.println("Enter Elements of Matrix 1: ");
    for(int i=0;i<n1;i++)
    {
        for(int j=0;j<m1;j++)
        {
            ar1[i][j]=sc.nextInt();
        }
    }
    System.out.println("Enter Elements of Matrix 2: ");
    for(int i=0;i<n1;i++)
    {
        for(int j=0;j<m1;j++)
        {
            ar2[i][j]=sc.nextInt();
        }
    }
    System.out.println("Sum Matrix: ");
    for(int i=0;i<n1;i++)
    {
        for(int j=0;j<m1;j++)
        {
            s[i][j]=ar1[i][j]+ar2[i][j];
            System.out.print(s[i][j]+" ");
        }
        System.out.println();
    }
}
}
}
}

```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>java p41
Enter Order of Matrix 1:
2 2
Enter Order of Matrix 2:
2 2
Enter Elements of Matrix 1:
1 2
3 4
Enter Elements of Matrix 2:
1 2
3 4
Sum Matrix:
2 4
6 8
```

MULTIPLICATION:

```
import java.lang.*;
import java.util.*;
public class p42
{
    public static void main(String args[])
    {
        int n1,m1,n2,m2;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Order of Matrix 1: ");
        n1=sc.nextInt();
        m1=sc.nextInt();
        System.out.println("Enter Order of Matrix 2: ");
        n2=sc.nextInt();
        m2=sc.nextInt();
        if(m1!=n2 )
        {
            System.out.println("Not Possible . ");
        }
        else
        {
            int ar1[][]=new int[n1][m1];
            int ar2[][]=new int[n2][m2];
            int s[][]=new int[n1][m2];
            System.out.println("Enter Elements of Matrix 1: ");
            for(int i=0;i<n1;i++)
            {
                for(int j=0;j<m1;j++)
                {
                    ar1[i][j]=sc.nextInt();
                }
            }
        }
    }
}
```

```

    }
    System.out.println("Enter Elements of Matrix 2: ");
    for(int i=0;i<n1;i++)
    {
        for(int j=0;j<m1;j++)
        {
            ar2[i][j]=sc.nextInt();
        }
    }
    System.out.println("Sum Matrix: ");
    for(int i=0;i<n1;i++)
    {
        for(int j=0;j<m1;j++)
        {
            for(int k=0;k<m1;k++)
            {
                s[i][j]+=ar1[i][k]*ar2[k][j];
            }
            System.out.print(s[i][j]+" ");
        }
        System.out.println();
    }
} } }

```

OUTPUT:

```

D:\MIT\sem5\webtech\RECORD>java p42
Enter Order of Matrix 1:
2 2
Enter Order of Matrix 2:
2 2
Enter Elements of Matrix 1:
1 2
3 4
Enter Elements of Matrix 2:
1 2
3 4
Sum Matrix:
7 10
15 22

```

TRANSPOSE:

```

import java.lang.*;
import java.util.*;
public class p43
{
    public static void main(String args[])
    {
        int n1,m1;
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter Order of Matrix 1: ");
    }
}

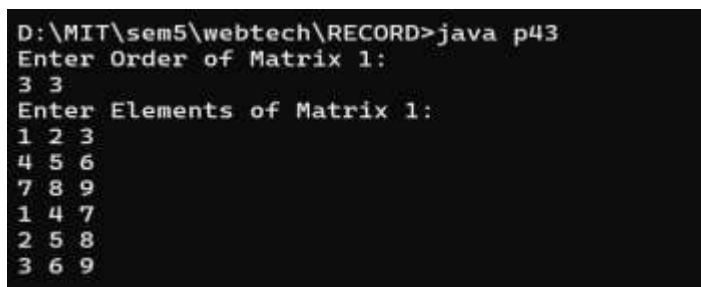
```

```

n1=sc.nextInt();
m1=sc.nextInt();
int ar1[][]=new int[n1][m1];
int tra[][]=new int[m1][n1];
System.out.println("Enter Elements of Matrix 1: ");
for(int i=0;i<n1;i++)
{
    for(int j=0;j<m1;j++)
    {
        ar1[i][j]=sc.nextInt();
    }
}
for(int i=0;i<n1;i++)
{
    for(int j=0;j<m1;j++)
    {
        tra[j][i]=ar1[i][j];
    }
}
for(int i=0;i<m1;i++)
{
    for(int j=0;j<n1;j++)
    {
        System.out.print(tra[i][j]+" ");
    }
    System.out.println();
}
}
}

```

OUTPUT:



```

D:\MIT\sem5\webtech\RECORD>java p43
Enter Order of Matrix 1:
3 3
Enter Elements of Matrix 1:
1 2 3
4 5 6
7 8 9
1 4 7
2 5 8
3 6 9

```

5. AIM :

To write a java program to merge two sorted arrays.

SOURCE CODE :

```

import java.lang.*;
public class p5

```

```

{
    public static void main(String[] args)
    {
        int ar1[]={1,3,5,7,9,10};
        int ar2[]={2,4,6,8,11};
        int n1=ar1.length;
        int n2=ar2.length;
        int ar3[]=new int[n1+n2];
        int i=0,j=0,k=0;
        while(i<n1 && j<n2)
        {
            if(ar1[i]<ar2[j])
            {
                ar3[k]=ar1[i];
                i++;
                k++;
            }
            else
            {
                ar3[k]=ar2[j];
                j++;
                k++;
            }
        }
        while(i<n1)
        {
            ar3[k]=ar1[i];
            i++;
            k++;
        }
        while(j<n2)
        {
            ar3[k]=ar2[j];
            j++;
            k++;
        }
        System.out.print("Array 1 : ");
        for(i=0;i<n1;i++)
        {
            System.out.print(ar1[i]+" ");
        }
        System.out.println("");
        System.out.print("Array 2 : ");
        for(i=0;i<n2;i++)
        {
            System.out.print(ar2[i]+" ");
        }
    }
}

```

```

        System.out.println("");
        System.out.print("Merge : ");
        for(i=0;i<n1+n2;i++)
        {
            System.out.print(ar3[i]+" ");
        }
        System.out.println("");
    }
}

```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p5.java
```

```
D:\MIT\sem5\webtech\RECORD>java p5
Array 1 : 1 3 5 7 9 10
Array 2 : 2 4 6 8 11
Merge : 1 2 3 4 5 6 7 8 9 10 11
```

6. AIM :

To write a java program to Create a list of strings and sort it alphabetically.

SOURCE CODE :

```

import java.lang.*;
import java.util.*;
public class p6
{
    public static void main(String ar[])
    {
        ArrayList<String> ar1=new ArrayList<String>();
        ar1.add("Bharath");
        ar1.add("Mithun");
        ar1.add("Bk");
        ar1.add("Harini");
        ar1.add("Aadhira");
        ar1.add("Babu");
        Collections.sort(ar1);
        for(int i=0;i<ar1.size();i++)
        {
            System.out.print(ar1.get(i)+" ");
        }
        System.out.println();
    }
}

```


OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p6.java  
  
D:\MIT\sem5\webtech\RECORD>java p6  
Aadhira Babu Bharath Bk Harini Mithun
```

7. AIM :

To write a java program to Remove all occurrences of a specific element from a list.

SOURCE CODE :

```
import java.lang.*;  
import java.util.*;  
public class p7  
{  
    public static void main(String ar[])  
    {  
        ArrayList<Integer> ar1=new ArrayList<Integer>();  
        ar1.add(1);  
        ar1.add(2);  
        ar1.add(2);  
        ar1.add(2);  
        ar1.add(1);  
        ar1.add(4);  
        ar1.add(3);  
        ar1.add(1);  
        ar1.add(5);  
        ar1.add(2);  
        for(int i=0;i<ar1.size();i++)  
        {  
            System.out.print(ar1.get(i)+" ");  
        }  
        System.out.println("\n Enter Element to be removed : ");  
        Scanner sc=new Scanner(System.in);  
        int n=sc.nextInt();  
        ar1.removeIf(e->e==n);  
        for(int i=0;i<ar1.size();i++)  
        {  
            System.out.print(ar1.get(i)+" ");  
        }  
        System.out.println();  
    }  
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p7.java
```

```
D:\MIT\sem5\webtech\RECORD>java p7
```

```
1 2 2 2 1 4 3 1 5 2
Enter Element to be removed :
2
1 1 4 3 1 5
```

8. AIM :

To write a java program to Convert an array to a list and vice versa.

SOURCE CODE :

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class ArrayListConversionExample {
    public static void main(String[] args) {
        Integer[] array = {1, 2, 3, 4, 5};
        List<Integer> list = Arrays.asList(array);

        System.out.println("Array to List:");
        System.out.println(list);
        List<Integer> anotherList = new ArrayList<>(list);
        Integer[] newArray = new Integer[anotherList.size()];
        newArray = anotherList.toArray(newArray);
        System.out.println("\nList to Array:");
        System.out.println(Arrays.toString(newArray));
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac ArrayListConversionExample.java
```

```
D:\MIT\sem5\webtech\RECORD>java ArrayListConversionExample
```

```
Array to List:
[1, 2, 3, 4, 5]
```

```
List to Array:
[1, 2, 3, 4, 5]
```

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

RESULT:

Thus, the java programs using arrays and list has been implemented successfully and the output has been verified

EXP NO : 3	Java programs using classes and interface
DATE : 29/8/24	

1.AIM :

To implement a Java program to create a CreditCard class using constructors to initialize fields for the owner, balance, and credit limit with customization options.

SOURCE CODE :

```
public class Person{
String name;
Person(String n){
name=n;}
public String getName(){
return name;}
}
public class Money{
double amt;
Money(double a){
amt=a;}
Money(Money obj){
amt=obj.amt;}
public double getAmount(){
```

```

return amt;}
}
public class creditCard{
    Person owner;
    Money balance;
    Money creditLimit;
    creditCard(Person own,Money m){
        owner=own;
        creditLimit=new Money(m);
        balance=new Money(0);
    }
    creditCard(Person own,Money m,Money mm){
        owner=own;
        creditLimit=new Money(m);
        balance=new Money(mm);
    }
    public void disp(){
        System.out.println(owner.getName());
        System.out.println(balance.getAmount());
        System.out.println(creditLimit.getAmount());
    }
    public static void main(String args[]){
        Person p1=new Person("babu");
        Money m1=new Money(20000);
        creditCard c1=new creditCard(p1,m1);
        c1.disp();}}

```

OUTPUT:

```

D:\MIT\sem5\webtech\RECORD>javac creditCard.java

D:\MIT\sem5\webtech\RECORD>java creditCard
babu
0.0
20000.0

```

2. AIM :

To create a class to print an integer and a character with two methods having the same name but different sequence of the integer and the character parameters. For example, if the parameters of the first method are of the form (int n, char c), then that of the second method will be of the form (char c, int n).

SOURCE CODE :

```

class p1
{
    void print(int n,char c)
    {
        System.out.println(n+" "+c);
    }
}

```

```

    }
    void print(char c,int n)
    {
        System.out.println(c+" "+n);
    }
    public static void main(String[] args)
    {
        p1 a=new p1();
        a.print(10,'a');
        a.print('b',11);

    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac p1.java

D:\MIT\sem5\webtech\RECORD>java p1
10  a
b   11

```

3. AIM :

To create a class called Calculator with the following methods:

- A static method called powering (int num1,int num2).This method should return num1 to the power num2.
- A static method called powerDouble(double num1,double num2). This method should return

num1 to the power num2.

- Invoke both the methods and test the functionalities.

SOURCE CODE :

```

class p2
{
    static void power(int a ,int b)
    {
        System.out.println(Math.pow(a,b));
    }
    static void powerdouble(double a,double b)
    {
        System.out.println(Math.pow(a,b));
    }
    public static void main(String[] args)
    {
        power(5, 3);
        powerdouble(4.3, 2);
    }
}

```

```
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p2.java
```

```
D:\MIT\sem5\webtech\RECORD>java p2  
125.0  
18.49
```

4. AIM :

To write program in Java for String handling (StringBuffer, StringBuilder) which performs the following:

- Checks the capacity.
- Reverse the contents of a string.
- Convert the string in upper case. Reads a string from console and appends it to the resultant string.

SOURCE CODE :

```
class p3  
{  
    public static void main(String[] args)  
    {  
        StringBuffer s1=new StringBuffer(30);  
        StringBuilder s2=new StringBuilder(20);  
        System.out.println("Capacity of String Buffer : "+s1.capacity());  
        System.out.println("Capacity of String Builder : "+s2.capacity());  
        s1.append("Hello world");  
        s2.append("Welcome to java Programming ...");  
        System.out.println("Reverse of String Buffer : "+s1.reverse());  
        System.out.println("Reverse of String Builder : "+s2.reverse());  
  
        System.out.println("Upper case of String Buffer : "+s1.toString().toUpperCase());  
        System.out.println("Upper Case of String Builder : "+s2.toString().toUpperCase());  
        s1.append(" temp");  
        s2.append(" temp");  
        System.out.println("After Appending of String Buffer : "+s1);  
        System.out.println("After Appending of String Builder : "+s2);  
    }  
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p3.java

D:\MIT\sem5\webtech\RECORD>java p3
Capacity of String Buffer : 30
Capacity of String Builder : 20
Reverse of String Buffer : dlrow olleH
Reverse of String Builder : ... gnimmargorP avaj ot emocleW
Upper case of String Buffer : DLROW OLLEH
Upper Case of String Builder : ... GNIMMARGORP AVAJ OT EMOCLEW
After Appending of String Buffer : dlrow olleH temp
After Appending of String Builder : ... gnimmargorP avaj ot emocleW temp
```

5. AIM :

To write program in Java which accepts an integer as input and performs the following operations:

- Reverse
- Check Equality conditions
- Conversion to Binary, hexadecimal, octal.

SOURCE CODE :

```
class p4
{
    public static void main(String[] args)
    {
        int a=1234;
        int rev=0;
        while(a!=0)
        {
            rev=rev*10+a%10;
            a/=10;
        }
        System.out.println("Reverse : "+rev);
        System.out.print("To check whether reverse in range (100-1000) : ");
        if(rev>100 && rev<1000)
            System.out.println("Yes");
        else
            System.out.println("No");
        System.out.println("Binary : "+Integer.toBinaryString(rev));
        System.out.println("Octal : "+Integer.toOctalString(rev));
        System.out.println("Hexa : "+Integer.toHexString(rev));
    }
}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac p4.java

D:\MIT\sem5\webtech\RECORD>java p4
Reverse : 4321
To check whether reverse in range (100-1000) : No
Binary : 1000011100001
Octal : 10341
Hexa : 10e1
```

6. AIM :

To write a Java program to create a vehicle class hierarchy. The base class should be Vehicle, with subclasses Truck, Car and Motorcycle. Each subclass should have properties such as make, model, year, and fuel type. Implement methods for calculating fuel efficiency, distance traveled, and maximum speed.

SOURCE CODE :

```
class Vehicle
{
    String make;
    String model;
    int year;
    String fuelType;

    public Vehicle(String make, String model, int year, String fuelType) {
        this.make = make;
        this.model = model;
        this.year = year;
        this.fuelType = fuelType;
    }

    public double calculateFuelEfficiency() {
        return 0.0;
    }

    public double calculateDistanceTraveled(double fuelUsed) {
        return 0.0;
    }

    public double calculateMaxSpeed() {
        return 0.0;
    }

    public void displayDetails() {
        System.out.println("Make: " + make);
        System.out.println("Model: " + model);
        System.out.println("Year: " + year);
        System.out.println("Fuel Type: " + fuelType);
    }
}
```



```

}

class Truck extends Vehicle {
    public Truck(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }

    @Override
    public double calculateFuelEfficiency() {
        return 8.0;
    }

    @Override
    public double calculateDistanceTraveled(double fuelUsed) {
        return fuelUsed * calculateFuelEfficiency();
    }

    @Override
    public double calculateMaxSpeed() {
        return 120.0;
    }
}

class Car extends Vehicle {
    public Car(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }

    @Override
    public double calculateFuelEfficiency() {
        return 15.0;
    }

    @Override
    public double calculateDistanceTraveled(double fuelUsed) {
        return fuelUsed * calculateFuelEfficiency();
    }

    @Override
    public double calculateMaxSpeed() {
        return 180.0;
    }
}

class Motorcycle extends Vehicle {
    public Motorcycle(String make, String model, int year, String fuelType) {
        super(make, model, year, fuelType);
    }
}

```

```

@Override
public double calculateFuelEfficiency() {
    return 40.0;
}

@Override
public double calculateDistanceTraveled(double fuelUsed) {
    return fuelUsed * calculateFuelEfficiency();
}

@Override
public double calculateMaxSpeed() {
    return 160.0;
}
}

public class VehicleHierarchy {
    public static void main(String[] args) {
        Truck truck = new Truck("Ford", "F-150", 2022, "Diesel");
        Car car = new Car("Toyota", "Camry", 2023, "Petrol");
        Motorcycle motorcycle = new Motorcycle("Yamaha", "YZF-R3", 2021, "Petrol");

        System.out.println("Truck Details:");
        truck.displayDetails();
        System.out.println("Fuel Efficiency: " + truck.calculateFuelEfficiency() + " km/l");
        System.out.println("Distance Traveled with 50 liters: " + truck.calculateDistanceTraveled(50) + " km");
        System.out.println("Max Speed: " + truck.calculateMaxSpeed() + " km/h");
        System.out.println();
        System.out.println("Car Details:");
        car.displayDetails();
        System.out.println("Fuel Efficiency: " + car.calculateFuelEfficiency() + " km/l");
        System.out.println("Distance Traveled with 50 liters: " + car.calculateDistanceTraveled(50) + " km");
        System.out.println("Max Speed: " + car.calculateMaxSpeed() + " km/h");
        System.out.println();
        System.out.println("Motorcycle Details:");
        motorcycle.displayDetails();
        System.out.println("Fuel Efficiency: " + motorcycle.calculateFuelEfficiency() + " km/l");
        System.out.println("Distance Traveled with 20 liters: " + motorcycle.calculateDistanceTraveled(20) + " km");
        System.out.println("Max Speed: " + motorcycle.calculateMaxSpeed() + " km/h");
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac VehicleHierarchy.java

D:\MIT\sem5\webtech\RECORD>java VehicleHierarchy
Truck Details:
Make: Ford
Model: F-150
Year: 2022
Fuel Type: Diesel
Fuel Efficiency: 8.0 km/l
Distance Traveled with 50 liters: 400.0 km
Max Speed: 120.0 km/h

Car Details:
Make: Toyota
Model: Camry
Year: 2023
Fuel Type: Petrol
Fuel Efficiency: 15.0 km/l
Distance Traveled with 50 liters: 750.0 km
Max Speed: 180.0 km/h

Motorcycle Details:
Make: Yamaha
Model: YZF-R3
Year: 2021
Fuel Type: Petrol
Fuel Efficiency: 40.0 km/l
Distance Traveled with 20 liters: 800.0 km
Max Speed: 160.0 km/h

```

7. AIM :

To create a Java program that models a bank account system. The system should have classes for Account, SavingsAccount, and CurrentAccount. Each account should have a balance, accountNumber, and interestRate(private). Account class should have methods calculate interest, validate deposit amount and updatebalance (protected).The SavingsAccount class should have an additional minimumBalance requirement. The CurrentAccount class should have an additional overdraftLimit.

SOURCE CODE :

```

class Account {
    private double balance, interestRate;
    private int accountNumber;
    Account(double b, double r, int an) {
        balance = b;
        interestRate = r;
        accountNumber = an;
    }
}

```

```

public double getBalance() {
    return balance;
}

public double getInterestRate() {
    return interestRate;
}

protected void interestCalc(int years) {
    double interest = (balance * interestRate * years) / 100;
    System.out.println("Interest after " + years + " year(s): " + interest);
}

protected void validateDeposit(double amount) {
    if (amount <= 0) {
        System.out.println("Invalid deposit amount.");
    } else {
        updateBalance(amount);
    }
}

protected void updateBalance(double amount) {
    balance += amount;
    System.out.println("Updated balance: " + balance);
}
}

class SavingsAccount extends Account {
    private double minBalance;
    SavingsAccount(double b, double r, int an, double mb) {
        super(b, r, an);
        minBalance = mb;
    }
    public void checkMinBalance() {
        if (getBalance() < minBalance) {
            System.out.println("Minimum balance reached.");
        } else {
            System.out.println("Sufficient balance.");
        }
    }
}

class CurrentAccount extends Account {
    private double overdraftLimit;

    CurrentAccount(double b, double r, int an, double od) {
        super(b, r, an);
        overdraftLimit = od;
    }
    public void checkOverdraftLimit() {

```

```

        if (getBalance() < -overdraftLimit) {
            System.out.println("Overdraft limit reached.");
        } else {
            System.out.println("Within overdraft limit.");    }  }}
public class BankAccSys {
    public static void main(String[] args) {
        SavingsAccount savings = new SavingsAccount(1000, 3, 101, 500);
        CurrentAccount current = new CurrentAccount(500, 2, 102, 1000);
        savings.interestCalc(1);
        savings.checkMinBalance();
        savings.validateDeposit(200);

        current.interestCalc(1);
        current.checkOverdraftLimit();
        current.validateDeposit(-200);
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac BankAccSys.java

D:\MIT\sem5\webtech\RECORD>java BankAccSys
Interest after 1 year(s): 30.0
Sufficient balance.
Updated balance: 1200.0
Interest after 1 year(s): 10.0
Within overdraft limit.
Invalid deposit amount.

```

8. AIM :

To write a Java programming to create a banking system with three classes - Bank, Account, SavingsAccount, and CurrentAccount. The bank should have a list of accounts and methods for adding them. Accounts should be an interface with methods to deposit, withdraw, calculate interest, and view balances. SavingsAccount and CurrentAccount should implement the Account interface and have their own unique methods.

SOURCE CODE :

```

import java.util.*;
interface Account {
    void deposit(double amount);
    void withdraw(double amount);
    double calculateInterest();
    double viewBalance();
}
class SavingsAccount implements Account {
    private int accountNumber;

```

```

private double balance;
private double interestRate;
private double minimumBalance;
public SavingsAccount(int accountNumber, double balance, double interestRate, double minimumBalance) {
    this.accountNumber = accountNumber;
    this.balance = balance;
    this.interestRate = interestRate;
    this.minimumBalance = minimumBalance;
}
@Override
public void deposit(double amount) {
    balance += amount;
    System.out.println("Deposited: " + amount);
}
@Override
public void withdraw(double amount) {
    if (balance - amount >= minimumBalance) {
        balance -= amount;
        System.out.println("Withdrawn: " + amount);
    } else {
        System.out.println("Withdrawal denied! Insufficient balance or below minimum balance.");
    }
}
@Override
public double calculateInterest() {
    return balance * interestRate / 100;
}
@Override
public double viewBalance() {
    return balance;
}
public void maintainMinimumBalance() {
    System.out.println("Minimum balance requirement is: " + minimumBalance);
}
}

class CurrentAccount implements Account {
    private int accountNumber;
    private double balance;
    private double overdraftLimit;
    public CurrentAccount(int accountNumber, double balance, double overdraftLimit) {
        this.accountNumber = accountNumber;
        this.balance = balance;
        this.overdraftLimit = overdraftLimit;
    }
    @Override
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount);
    }
}

```

```

    }
    @Override
    public void withdraw(double amount) {
        if (balance - amount >= -overdraftLimit) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
        } else {
            System.out.println("Withdrawal denied! Exceeded overdraft limit.");
        }
    }
    @Override
    public double calculateInterest() {
        return 0;
    }
    @Override
    public double viewBalance() {
        return balance;
    }
    public void checkOverdraftLimit() {
        System.out.println("Overdraft limit is: " + overdraftLimit);
    }
}

class Bank {
    private List<Account> accounts;
    public Bank() {
        accounts = new ArrayList<>();
    }
    public void addAccount(Account account) {
        accounts.add(account);
        System.out.println("Account added.");
    }
    public void viewAllBalances() {
        for (Account account : accounts) {
            System.out.println("Balance: " + account.viewBalance());
        }
    }
}

public class BankSystem {
    public static void main(String[] args) {
        Bank bank = new Bank();
        SavingsAccount savings = new SavingsAccount(101, 5000, 5.0, 1000);
        CurrentAccount current = new CurrentAccount(102, 2000, 500);
        bank.addAccount(savings);
        bank.addAccount(current);
        savings.deposit(1000);
        savings.withdraw(6000);
        System.out.println("Interest on savings: " + savings.calculateInterest());
        savings.maintainMinimumBalance();
        current.deposit(500);
        current.withdraw(3000);
    }
}

```

```

        current.checkOverdraftLimit();
        bank.viewAllBalances();
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac BankSystem.java

D:\MIT\sem5\webtech\RECORD>java BankSystem
Account added.
Account added.
Deposited: 1000.0
Withdrawal denied! Insufficient balance or below minimum balance.
Interest on savings: 300.0
Minimum balance requirement is: 1000.0
Deposited: 500.0
Withdrawn: 3000.0
Overdraft limit is: 500.0
Balance: 6000.0
Balance: -500.0

```

9. AIM :

We have to calculate the percentage of marks obtained in three subjects (each out of 100) by student A and in four subjects (each out of 100) by student B. Create an abstract class 'Marks' with an abstract method 'getPercentage'. It is inherited by two other classes 'A' and 'B' each having a method with the same name which returns the percentage of the students. The constructor of student A takes the marks in three subjects as its parameters and the marks in four subjects as its parameters for student B. Create an object for each of the two classes and print the percentage of marks for both the students.

SOURCE CODE :

```

class Marks {
    public void getPercentage() {
    }
}

class A extends Marks {
    double a, b, c;
    A(double aa, double bb, double cc) {
        a = aa;
        b = bb;
        c = cc;
    }
    public void getPercentage() {
        double percentage = (a + b + c) / 3.0;
        System.out.println("Percentage of marks for Student A: " + percentage + "%");
    }
}

```



```

    }
}
class B extends Marks {
    double a, b, c, d;
    B(double aa, double bb, double cc, double dd) {
        a = aa;
        b = bb;
        c = cc;
        d = dd;
    }
    public void getPercentage() {
        double percentage = (a + b + c + d) / 4.0;
        System.out.println("Percentage of marks for Student B: " + percentage + "%");
    }
}
public class MarksAB {
    public static void main(String[] args) {
        A studentA = new A(85, 90, 88);
        B studentB = new B(78, 82, 79, 91);
        studentA.getPercentage();
        studentB.getPercentage();
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac MarksAB.java

D:\MIT\sem5\webtech\RECORD>java MarksAB
Percentage of marks for Student A: 87.66666666666667%
Percentage of marks for Student B: 82.5%

```

10. AIM :

To write a program for onlinebookshop which sells and the add college books and story books as subclasses of the Book class which holds the details about book name, publisher, Published year, edition and price. Implement the polymorphic behavior through overriding methods. Explore the usage of Upcasting and Downcasting also.

SOURCE CODE :

```

public class Book {
    private String name;
    private String publisher;
    private int publishedYear;
    private String edition;
    private double price;
    public Book(String name, String publisher, int publishedYear, String edition, double price) {
        this.name = name;
        this.publisher = publisher;
        this.publishedYear = publishedYear;
    }
}

```

```

        this.edition = edition;
        this.price = price;
    }
    public String displayInfo() {
        return "Name: " + name + "\n" +
            "Publisher: " + publisher + "\n" +
            "Year: " + publishedYear + "\n" +
            "Edition: " + edition + "\n" +
            "Price: $" + String.format("%.2f", price);
    }
}

public class CollegeBook extends Book {
    private String courseCode;
    public CollegeBook(String name, String publisher, int publishedYear, String edition, double price, String courseCode) {
        super(name, publisher, publishedYear, edition, price);
        this.courseCode = courseCode;
    }
    @Override
    public String displayInfo() {
        return super.displayInfo() + "\nCourse Code: " + courseCode;
    }
}

public class StoryBook extends Book {
    private String genre;
    public StoryBook(String name, String publisher, int publishedYear, String edition, double price, String genre) {
        super(name, publisher, publishedYear, edition, price);
        this.genre = genre;
    }
    @Override
    public String displayInfo() {
        return super.displayInfo() + "\nGenre: " + genre;    }}

public class BookShop {
    public static void main(String[] args) {
        Book collegeBook = new CollegeBook("Advanced Calculus", "Math Publisher", 2023, "3rd Edition", 49.99,
"MATH101");
        Book storyBook = new StoryBook("The Great Adventure", "Fiction House", 2021, "1st Edition", 19.99, "Adventure");
        System.out.println("College Book Info:");
        System.out.println(collegeBook.displayInfo());
        System.out.println();
        System.out.println("Story Book Info:");
        System.out.println(storyBook.displayInfo());
        System.out.println();
        if (collegeBook instanceof CollegeBook) {
            CollegeBook castedCollegeBook = (CollegeBook) collegeBook;
            System.out.println("Downcasted College Book Info:");
            System.out.println(castedCollegeBook.displayInfo());}
        if (storyBook instanceof StoryBook) {

```

```
StoryBook castedStoryBook = (StoryBook) storyBook;  
System.out.println("Downcasted Story Book Info:");  
System.out.println(castedStoryBook.displayInfo());  
} }}
```

OUTPUT :

```
D:\MIT\sem5\webtech\RECORD>javac BookShop.java
```

```
D:\MIT\sem5\webtech\RECORD>java BookShop
```

College Book Info:

Name: Advanced Calculus

Publisher: Math Publisher

Year: 2023

Edition: 3rd Edition

Price: \$49.99

Course Code: MATH101

Story Book Info:

Name: The Great Adventure

Publisher: Fiction House

Year: 2021

Edition: 1st Edition

Price: \$19.99

Genre: Adventure

Downcasted College Book Info:

Name: Advanced Calculus

Publisher: Math Publisher

Year: 2023

Edition: 3rd Edition

Price: \$49.99

Course Code: MATH101

Downcasted Story Book Info:

Name: The Great Adventure

Publisher: Fiction House

Year: 2021

Edition: 1st Edition

Price: \$19.99

Genre: Adventure

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

RESULT:

Thus, the java programs using class and interface has been implemented successfully and the output has been verified.

EXP NO : 4	Java Programs Using Exception Handling and Regular Expression
DATE : 5/9/24	

1. AIM :

- Modify the factorial method to throw an `IllegalArgumentException` for negative integers, instead of returning 1.
- Modify the `main` method in Factorials to catch this exception and print an appropriate message, but continue with the loop.
- Also, handle large positive integers (>16) which cause overflow by throwing an `IllegalArgumentException`.

SOURCE CODE :

```
import java.util.Scanner;
public class Factorials {
```

```

public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    while (true) {
        System.out.print("Enter a positive integer (or a negative number to exit): ");
        int num = scan.nextInt();
        try {
            System.out.println("Factorial: " + MathUtils.factorial(num));
        } catch (IllegalArgumentException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}

class MathUtils {
    public static int factorial(int n) {
        if (n < 0) {
            throw new IllegalArgumentException("Factorial is not defined for negative integers.");
        } else if (n > 16) {
            throw new IllegalArgumentException("Factorial is too large to calculate for integers greater than 16.");
        }
        int result = 1;
        for (int i = 2; i <= n; i++) {
            result *= i;
        }
        return result;
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>java Factorials
Enter a positive integer (or a negative number to exit): 2
Factorial: 2
Enter a positive integer (or a negative number to exit): 4
Factorial: 24
Enter a positive integer (or a negative number to exit): -1
Error: Factorial is not defined for negative integers.

```

2. AIM :

- Read student data from a file and compute GPA.
- Write students with a GPA below a threshold to an output file.
- Handle exceptions like `FileNotFoundException`, `NumberFormatException`, and `IOException`, giving specific error messages.

SOURCE CODE :

```

import java.util.Scanner;

public class Warning {
    public static void main(String[] args) {
        Scanner scan = null;
        PrintWriter outFile = null;
        try {
            scan = new Scanner(new File("students.dat"));
            outFile = new PrintWriter(new FileWriter("warningList.txt"));
        }
    }
}

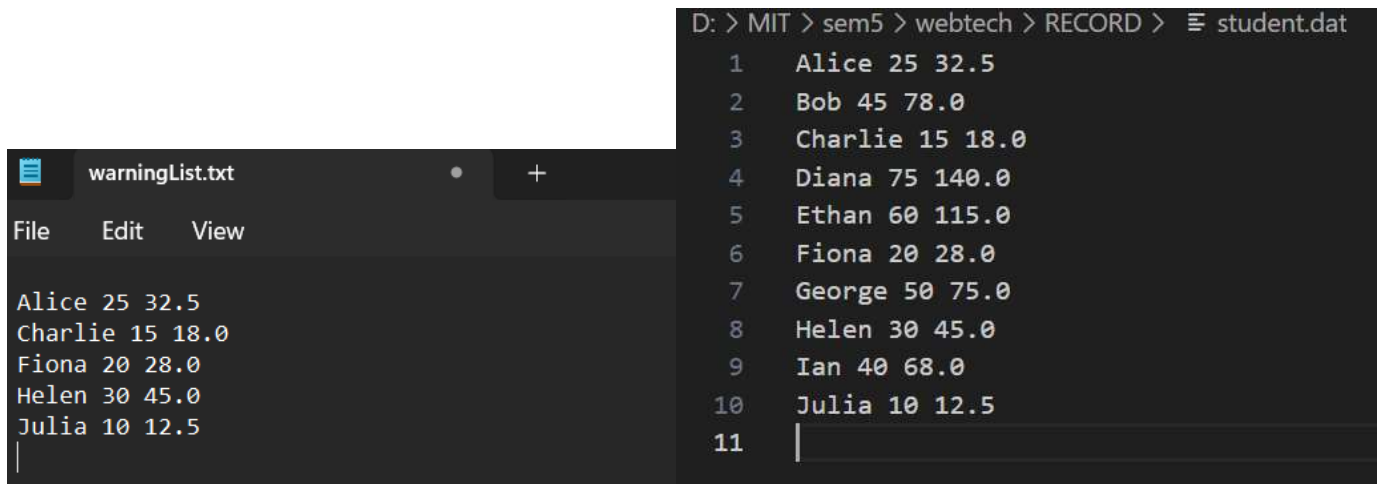
```

```

while (scan.hasNext()) {
    String name = scan.next();
    int creditHours = scan.nextInt();
    double qualityPoints = scan.nextDouble();
    double gpa = qualityPoints / creditHours;
    if ((creditHours < 30 && gpa < 1.5) ||
        (creditHours < 60 && gpa < 1.75) ||
        (gpa < 2.0)) {
        outFile.println(name + " " + creditHours + " " + String.format("%.2f", gpa));
    }
} catch (FileNotFoundException e) {
    System.out.println("Error: Input file not found.");
} catch (NumberFormatException e) {
    System.out.println("Error: Invalid number format in input file.");
} catch (IOException e) {
    System.out.println("Error: Problem with file input/output.");
} finally {
    if (scan != null) {
        scan.close();
    }
    if (outFile != null) {
        outFile.close();
    }
}

```

OUTPUT :



```

D: > MIT > sem5 > webtech > RECORD > student.dat
1 Alice 25 32.5
2 Bob 45 78.0
3 Charlie 15 18.0
4 Diana 75 140.0
5 Ethan 60 115.0
6 Fiona 20 28.0
7 George 50 75.0
8 Helen 30 45.0
9 Ian 40 68.0
10 Julia 10 12.5
11

```

3. AIM :

To ensure the name input consists of one or more characters followed by a space and another set of characters. If it doesn't match, display an "Incorrect format for name" message.

SOURCE CODE :

```

import java.util.regex.Pattern;
import java.util.regex.Matcher;
public class NameValidator {
    private static final String NAME_PATTERN = "^[A-Za-z]+ [A-Za-z]+$";
    public static void main(String[] args) {
        String name1 = "John Doe";
        String name2 = "John";
        System.out.println(validateName(name1));
        System.out.println(validateName(name2));
    }
}

```

```

    }
    public static String validateName(String name) {
        Pattern pattern = Pattern.compile(NAME_PATTERN);
        Matcher matcher = pattern.matcher(name);
        if (matcher.matches()) {
            return "Name is valid.";
        } else {
            return "Incorrect format for name.";
        }
    }
}

```

OUTPUT :

```

D:\MIT\sem5\webtech\RECORD>javac NameValidator.java

D:\MIT\sem5\webtech\RECORD>java NameValidator
Name is valid.
Incorrect format for name.

```

4. AIM :

- Use a regex to filter characters (a-f, A-F) from a coded answer key file and store them.
- Replace certain characters (`e -> b`, `E -> A`, etc.), convert to lowercase, and return the modified string for use on the exam.

SOURCE CODE :

```

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.HashMap;
import java.util.Map;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class AnswerKeyDecoder
{
    public static void main(String[] args) {
        String filePath = "key.txt"; // Update this path
        try {
            String decodedString = decodeAnswerKey(filePath);
            System.out.println(decodedString);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    public static String decodeAnswerKey(String filePath) throws IOException {
        String content = new String(Files.readAllBytes(Paths.get(filePath)));
        String filteredContent = content.replaceAll("[^a-zA-F]", "");
    }
}

```

```

        System.out.println(filteredContent);
Map<Character, Character> replacements = new HashMap<>();
    replacements.put('e', 'b');
    replacements.put('E', 'A');
    replacements.put('f', 'c');
    replacements.put('F', 'C');


StringBuilder sb = new StringBuilder();
    for (char c : filteredContent.toCharArray()) {
        sb.append(replacements.getOrDefault(c, c));
    }

    // Step 4: Convert to lowercase
    String finalContent = sb.toString().toLowerCase();

    return finalContent;
}
}

```

OUTPUT :



```

D: > MIT > sem5 > webtech
1  A
2  V
3  S
4  C
5  S
6  B
7  F
8  R
9  R
10 E
11 W
12 F

D:\MIT\sem5\webtech\RECORD>javac AnswerKeyDecoder.java

D:\MIT\sem5\webtech\RECORD>java AnswerKeyDecoder
ACBF EF
acbcac

```


Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

RESULT:

Thus, the java programs using exception handling and regular expression has been implemented successfully and the output has been verified.

EXP NO : 5	Java programs using multithreading
DATE : 26/09/24	

1.AIM :

To implement a Java program that creates three threads, each displaying different messages at specified time intervals.

SOURCE CODE :

```
public class mt1 extends Thread{
int i;String msg;
mt1(int n, String x){
```

```

i=n;
msg=x;}
public void run(){
while(true){
try{
Thread.sleep(i);
System.out.println(msg);}
catch (Exception e){
System.out.println(e);}}}
public static void main(String a[]){
mt1 t1=new mt1(1000,"Hello!");
mt1 t2=new mt1(2000,"Happy Holidays!");
mt1 t3=new mt1(5000,"Enjoy!");
t1.start();
t2.start();
t3.start();}}

```

OUTPUT :

```

Hello!
Happy Holidays!
Hello!
Hello!
Happy Holidays!
Hello!
Enjoy!
Hello!
Happy Holidays!
Hello!
Hello!
Happy Holidays!
Hello!
Hello!
Enjoy!
Happy Holidays!
Hello!
Hello!

```

2.AIM :

To implement a Java program that creates five threads with different priorities, checks their aliveness, and observes the effect of thread priority and sleep state.

SOURCE CODE :

```

public class mt2 extends Thread{
    mt2(String n){
        name=n;}
    public void run(){
        try{
            System.out.println("Inside run "+name+" "+Thread.currentThread().isAlive()+
Priority: "+Thread.currentThread().getPriority());
        }
        catch(Exception e){
            System.out.println(e);
        }}

```

```

public static void main(String args[]){
    mt2 t1=new mt2("t1");
    mt2 t2=new mt2("t2");
    mt2 t3=new mt2("t3");
    mt2 t4=new mt2("t4");
    mt2 t5=new mt2("t5");
    t1.setPriority(3);
    t2.setPriority(Thread.MAX_PRIORITY);
    t3.setPriority(Thread.MIN_PRIORITY);
    t4.setPriority(Thread.NORM_PRIORITY);
    t5.setPriority(Thread.MAX_PRIORITY);
    t1.start();
    t2.start();
    t3.start();
    t4.start();
    t5.start();
    try{
        t2.sleep(2000);
        t5.sleep(2000); }
    catch(Exception e){
        System.out.println(e);}System.out.println("t1 is alive: " + t1.isAlive());
    System.out.println("t2 is alive: " + t2.isAlive());
    System.out.println("t3 is alive: " + t3.isAlive());
    System.out.println("t4 is alive: " + t4.isAlive());
    System.out.println("t5 is alive: " + t5.isAlive());    }}

```

OUTPUT :

```

Inside run t5 true Priority: 10
Inside run t2 true Priority: 10
Inside run t4 true Priority: 5
Inside run t3 true Priority: 1
Inside run t1 true Priority: 3
t1 is alive: false
t2 is alive: false
t3 is alive: false
t4 is alive: false
t5 is alive: false

```

3.AIM :

To implement a Java program that creates a multi-threaded application with three threads where one generates random integers, another calculates squares for even values, and a third computes cubes for odd values.

SOURCE CODE :

```

class ranNum extends Thread
{
    int r; public void run()
    {
        try
        {
            for(int i=0;i<5;i++)
            {

```

```

        r=(int)(Math.random()*100);
        System.out.println("Random number: "+r);
        synchronized(this)
        {
            notifyAll();
        }
        Thread.sleep(1000);
    }
}
catch(Exception e)
{
    System.out.println(e);
}
}
public synchronized int getRan()throws InterruptedException
{
    wait();
    return r;}}

class oddSq extends Thread
{
    final ranNum a;
    oddSq(ranNum b)    {
        a=b;    }
    public void run()    {
        try    {
            for(int i=0;i<5;i++)
            {
                int v=a.getRan();
                if(v%2!=0)
                {
                    System.out.println("cube: "+v*v*v);
                }
            }
        }
        catch(Exception e)    {
            System.out.println(e);
        }
    }
}

class evenSq extends Thread
{
    final ranNum a;
    evenSq(ranNum b)
    {
        a=b;    }
    public void run()    {
        try    {
            for(int i=0;i<5;i++)
            {
                int v=a.getRan();
                if(v%2==0)
                {
                    System.out.println("square: "+v*v);
                }
            }
        }
        catch(Exception e)    {
            System.out.println(e);
        }
    }
}

public class mt3 {

    public static void main(String a[])    {
        ranNum x=new ranNum();

```

```

        evenSq e=new evenSq(x);
        oddSq o=new oddSq(x);
        x.start();
        e.start();
        o.start();    }}

```

OUTPUT :

```

Random number: 33
cube: 35937
Random number: 68
square: 4624
Random number: 29
cube: 24389
Random number: 51
cube: 132651
Random number: 71
cube: 357911

```

4.AIM :

To implement a Java program that uses two synchronized threads to display alphabets in opposite directions, ensuring one thread waits for the other to finish.

SOURCE CODE :

```

class Alpha extends Thread {
    public void run() {
        try {
            for (char c = 'A'; c <= 'Z'; ++c) {
                System.out.print(c + " ");
                Thread.sleep(1000);
            }
            System.out.print("\n");
        } catch (Exception e) {
            System.out.println(e);    }    }}
class AlphaReverse extends Thread {
    public void run() {
        try {
            for (char c = 'Z'; c >= 'A'; --c) {
                System.out.print(c + " ");
                Thread.sleep(2000);
            }
        } catch (Exception e) {
            System.out.println(e);    }    }}
public class mt4 {
    public static void main(String[] args) {
        Alpha a = new Alpha();
        AlphaReverse ar = new AlphaReverse();
        a.start();
        try {
            a.join();
        } catch (Exception e) {
            System.out.println(e);    }
    }
}

```

```

ar.start();
try {
    ar.join();
} catch (Exception e) {
    System.out.println(e);    }  }}

```

OUTPUT :



```

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Z Y X W V U T S R Q P O N M L K J I H G F E D C B A

```

5.AIM :

To implement a Java program that simulates a shared bank account, using thread synchronization to prevent race conditions and ensure correct balance updates during concurrent deposits and withdrawals.

SOURCE CODE :

```

class Bank extends Thread {
    double bal;
    public Bank() {
        bal = 0.0;
    }

    public synchronized void deposit(double amt) {
        bal += amt;
        System.out.println("Amount " + amt + " deposited \nTotal Balance: " + bal);
        notify();    }
    public synchronized void withdraw(double amt) {
        if (bal < amt) {
            System.out.println("Insufficient balance for withdrawal of " + amt);
        } else {
            bal -= amt;
            System.out.println("Amount " + amt + " withdrawn \nTotal Balance: " + bal);
        }
        notify();    }
    public synchronized double getBal() throws InterruptedException {
        wait();
        return bal;    }}

class AccOp extends Thread {
    Bank bank;
    char operation;
    double amount;
    public AccOp(Bank bank, char operation, double amount) {
        this.bank = bank;
        this.operation = operation;
        this.amount = amount;    }
    public void run() {
        if (operation == 'd') {
            bank.deposit(amount);
        } else if (operation == 'w') {
            bank.withdraw(amount);    }}
}

public class mt5 {
    public static void main(String args[]) {

```

```

Bank bank = new Bank();
AccOp u1 = new AccOp(bank, 'd', 200.0);
AccOp u2 = new AccOp(bank, 'w', 150.0);
u1.start();
u2.start();  }}

```

OUTPUT :

```

D:\software\abi\java_practice>java mt5
Amount 200.0 deposited
Total Balance: 200.0
Amount 150.0 withdrawn
Total Balance: 50.0

```

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Hence, basic Java programs to explore fundamental topics using multithreading is successfully implemented.

EXP NO : 6	I/O Streams and object serialization
DATE : 03/10/24	

Processing Large Log Files

1.AIM :

To implement a Java program that reads a large server log file line by line, filters lines containing the keyword 'ERROR,' writes them to a new file, and reports the total number of such lines efficiently.

SOURCE CODE :

server_log.txt:

```
20-10-2024 INFO : ERROR - Unable to create an instance.
20-10-2024 INFO : OS booted.
20-10-2024 INFO : Cleared cache.
20-10-2024 INFO : ERROR - DNS request failed.
20-10-2024 INFO : ERROR - 404:Server not found.
20-10-2024 INFO : Key created.
```

```
import java.io.*;
public class io1{
    public static void main(String args[]){
        int c=0;
        try{
            FileInputStream f1=new FileInputStream("server_log.txt");
            InputStreamReader x1= new InputStreamReader(f1);
            BufferedReader r1=new BufferedReader(x1);
            FileOutputStream f2=new FileOutputStream("error.txt");
            OutputStreamWriter x2= new OutputStreamWriter(f2);
            BufferedWriter r2=new BufferedWriter(x2);
            String s;
            while((s=r1.readLine())!=null){
                if(s.contains("ERROR")){
                    r2.write(s);
                    r2.newLine();
                    c++;
                }
            }
            r1.close();
            r2.close();
            System.out.println("Total errors: "+c);}
        catch(Exception e){
            System.out.println(e);}}}
}
```

OUTPUT :

```
Total errors: 3
```

error.txt

```
20-10-2024 INFO : ERROR - Unable to create an instance.
20-10-2024 INFO : ERROR - DNS request failed.
20-10-2024 INFO : ERROR - 404:Server not found.
```

Deserializing and Filtering Network Data Packets

2. AIM :

To implement a Java program that reads serialized sensor data packets from a network, deserializes them into objects, filters packets based on temperature readings, and stores the filtered data for further analysis.

SOURCE CODE :

```
import java.io.*;
import java.util.*;
class data implements Serializable {
    double temp, hum, pres;
    public data(double t, double h, double p) {
```



```

        temp = t;
        hum = h;
        pres = p;
    }
    public double getTem() {
        return temp;    }
}
public class io2 {
    public static byte[] serialize(data d) {
        try (ByteArrayOutputStream b = new ByteArrayOutputStream();
             ObjectOutputStream out = new ObjectOutputStream(b)) {
            out.writeObject(d);
            return b.toByteArray();
        } catch (IOException e) {
            e.printStackTrace();    }
        return null;    }
    public static void deserialize(byte[] dataBytes, double threshold, List<data> l) {
        try (ByteArrayInputStream bin = new ByteArrayInputStream(dataBytes);
             ObjectInputStream oin = new ObjectInputStream(bin)) {
            while (true) {
                try {
                    data d = (data) oin.readObject();
                    if (d.getTem() > threshold) {
                        l.add(d);
                    }
                } catch (Exception e) {
                    break;    }
            }
        } catch (Exception e) {
            e.printStackTrace();    }
    }
    public static void main(String[] args) {
        List<data> l = new ArrayList<>();
        data d1 = new data(30.0, 70.0, 900);
        data d2 = new data(35.5, 60.0, 1000);
        data d3 = new data(40.0, 75.0, 950);
        byte[] p1 = serialize(d1);
        byte[] p2 = serialize(d2);
        byte[] p3 = serialize(d3);
        deserialize(p1, 30.0, l);
        deserialize(p2, 30.0, l);
        deserialize(p3, 30.0, l);
        System.out.println("> 30°C");
        for (data d : l) {
            System.out.println("Temperature: " + d.getTem() + "°C");    }    }
}

```

OUTPUT :

```

E:\java_practice>java io2
> 30°C
Temperature: 35.5°C
Temperature: 40.0°C

```

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Hence, Java programs to efficiently process large server log files using filtering and file handling, and deserialize network data packets with object-oriented principles for analysis is successfully implemented.

EXP NO : 7	Java networking programs
DATE : 10/10/24	

Real-Time Weather Update System (UDP Protocol)

1.AIM :

To develop a real-time weather update system that broadcasts weather information from a weather station to multiple devices on a local network using the UDP protocol.

SOURCE CODE :

Server:

```
import java.net.*;
import java.net.InetAddress;
public class WeatherInfo {
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket();
        InetAddress ia = InetAddress.getByName("255.255.255.255");
        int port = 1234;
        String info = "Temperature: 25°C, Humidity: 60%";
        while (true) {
            DatagramPacket pac = new DatagramPacket(info.getBytes(), info.length(), ia, port);
            ds.send(pac);
            Thread.sleep(1000);    }  }}
import java.net.*;
import java.net.InetAddress;
public class device {
    public static void main(String[] args) throws Exception {
        DatagramSocket ds = new DatagramSocket(1234);
        byte[] buf = new byte[1024];
        DatagramPacket pac = new DatagramPacket(buf, buf.length);
        ds.receive(pac);
        String info = new String(pac.getData(), 0, pac.getLength());
        System.out.println("Received weather update: " + info);}}
```

OUTPUT :

Server:

```
E:\java_practice>javac WeatherInfo.java
E:\java_practice>java WeatherInfo
```

Client:

```
E:\java_practice>java device
Received weather update: Temperature: 25°C, Humidity: 60
```

TCP-Based Multi-Client Auction System

2. AIM :

To implement a TCP-based multi-client auction system where clients can place bids, the server tracks the highest bid, and all clients are notified of the auction's current status in real-time.

SOURCE CODE :

```
import java.net.*;
import java.io.*;
import java.util.*;
public class AuctionServer {
```

```

public static void main(String[] args) throws Exception {
    ServerSocket server = new ServerSocket(8000);
    System.out.println("Auction Server started on port 8000");
    double h = 0.0;
    String a = "";
    List<Socket> l = new ArrayList<>();
    while (true) {
        Socket cli = server.accept();
        System.out.println("Client connected");
        l.add(cli);
        Thread cth = new Thread(new ClientHandler(cli, l, h, a));
        cth.start();
    }
}

class ClientHandler implements Runnable {
    private Socket c;
    private List<Socket> l;
    private double h;
    private String a;
    public ClientHandler(Socket x, List<Socket> y, double z, String b) {
        this.c = x;
        this.l = y;
        this.h = z;
        this.a = b;
    }
    public void run() {
        try {
            BufferedReader in = new BufferedReader(new InputStreamReader(c.getInputStream()));
            PrintWriter out = new PrintWriter(c.getOutputStream(), true);
            while (true) {
                String message = in.readLine();
                if (message == null) break;
                try {
                    double bid = Double.parseDouble(message);
                    synchronized (l) {
                        if (bid > h) {
                            h = bid;
                            a = c.getInetAddress().getHostName();
                            broadcast("New highest bid: " + a + " - " + h);
                        } else {
                            out.println("Your bid is too low. Current highest bid: " + a + " - " +
h);
                        }
                    }
                } catch (NumberFormatException e) {
                    out.println("Invalid bid. Please enter a valid number.");
                }
                l.remove(c);
                c.close();
            }
        } catch (Exception e) {
            System.out.println("Error handling client: " + e.getMessage());
        }
    }
    private void broadcast(String message) {
        synchronized (l) {
            for (Socket client : l) {
                try {
                    PrintWriter out = new PrintWriter(client.getOutputStream(), true);
                    out.println(message);
                } catch (Exception e) {
                    System.out.println("Error broadcasting to client: " + e.getMessage());
                }
            }
        }
    }
}

import java.net.*;

```

```

import java.io.*;
public class AuctionClient {
    public static void main(String[] args) throws Exception {
        Socket client = new Socket("localhost", 8000);
        System.out.println("Connected to auction server");
        BufferedReader in = new BufferedReader(new InputStreamReader(client.getInputStream()));
        PrintWriter out = new PrintWriter(client.getOutputStream(), true);
        BufferedReader userInput = new BufferedReader(new InputStreamReader(System.in));
        Thread responseThread = new Thread(new ResponseHandler(in));
        responseThread.start();
        while (true) {
            System.out.print("Enter");
            String message = userInput.readLine();
            out.println(message);    }  }}
class ResponseHandler implements Runnable {
    private BufferedReader in;
    public ResponseHandler(BufferedReader in) {
        this.in = in;    }
    public void run() {
        try {
            while (true) {
                String response = in.readLine();
                if (response == null) break;
                System.out.println("Server: " + response);    }    } catch (Exception e) {
                System.out.println("Error handling server response: " + e.getMessage());    }    }}

```

OUTPUT :

Server:

```

Auction Server started on port 8000
Client connected
Client connected

```

Client:

```

E:\java_practice>java AuctionClient
Connected to auction server
Enter your bid: 100
Enter your bid: Server: New highest bid: 127.0.0.1 - 100.0
Server: New highest bid: 127.0.0.1 - 200.0
300
Enter your bid: Server: New highest bid: 127.0.0.1 - 300.0
Server: New highest bid: 127.0.0.1 - 400.0
100

```

```

E:\java_practice>java AuctionClient
Connected to auction server
Enter your bid: Server: New highest bid: 127.0.0.1 - 100.0
200
Enter your bid: Server: New highest bid: 127.0.0.1 - 200.0
Server: New highest bid: 127.0.0.1 - 300.0
100
Enter your bid: Server: Your bid is too low. Current highest bid: 127.0.0.1 - 200.0
400

```

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Developed a real-time weather update system using the UDP protocol to broadcast weather information from a weather station to multiple devices, and implement a TCP-based multi-client auction system where clients can place bids, the server tracks the highest bid, and notifies all clients about the auction's current status.

EXP NO : 8	Java swing programs
DATE : 17/11/24	

1.AIM :

To develop a Java Swing program that simulates raindrops falling from the top of the window, with each raindrop moving vertically at a random speed and disappearing once it reaches the bottom.

SOURCE CODE :

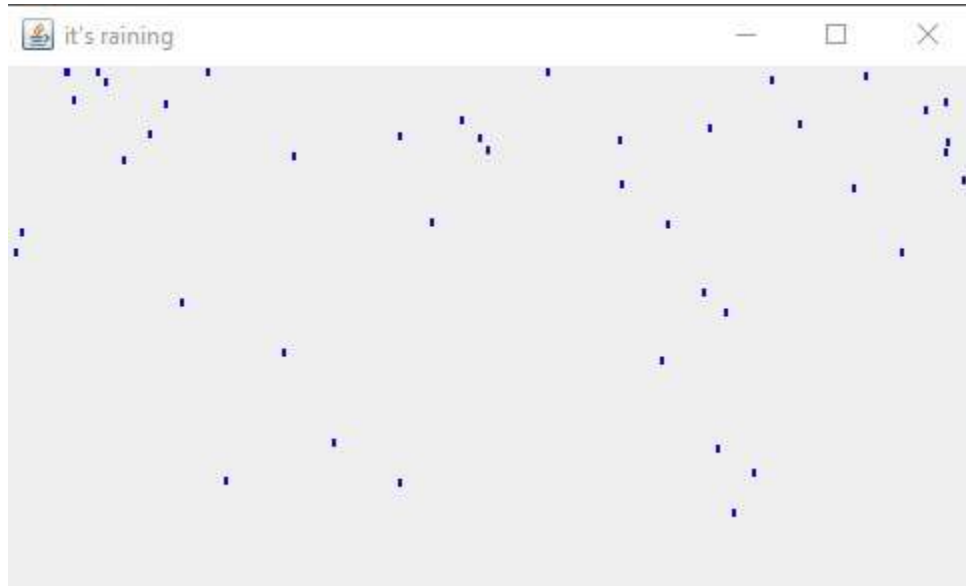
```

import javax.swing.*;
import javax.swing.Timer;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.List;
import java.util.ArrayList;
import java.util.Random;
import java.awt.Graphics;
import java.awt.Color;
public class rainy extends JPanel implements ActionListener {
    class drop{
        int x,y,s;
        drop(int a,int b,int c){
            x=a;
            y=b;
            s=c;
        }
    }
    List<drop>l=new ArrayList<>();
    Timer t;
    Random ran=new Random();
    rainy(){
        t=new Timer(30,this);
        t.start();
    }
    public void dropsCreate(){
        int yVal=0,xVal=ran.nextInt(500);
        int sp=ran.nextInt(6);
        l.add(new drop(xVal,yVal,sp));
    }
    public void actionPerformed(ActionEvent e){
        for(int i=0;i<l.size();i++){
            drop d=l.get(i);
            d.y=d.y+d.s;
            if(d.y>getHeight()){
                l.remove(i);
                i--;
            }
        }
        if(ran.nextInt(10)>3){
            dropsCreate();
        }
        repaint();
    }
    public void paintComponent(Graphics g){
        super.paintComponent(g);
        g.setColor(Color.BLUE);
        for(drop i:l){
            g.fillOval(i.x,i.y,3,5);}
    }
    public static void main(String args[]){
        rainy r=new rainy();
        JFrame f=new JFrame("it's raining");
    }
}

```

```
f.add(r);
f.setSize(500,500);
f.setVisible(true);
}}
```

OUTPUT:



2.AIM :

To design a currency converter in Java Swing, where users can input an amount, select currencies to convert from and to, and display the converted amount based on predefined exchange rates.

SOURCE CODE :

```
import javax.swing.*.*;
import java.awt.*.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class CurrencyConverter extends JFrame {
    private double exchangeRates[][] = {
        {1, 0.88, 0.76},
        {1.14, 1, 0.86},
        {1.32, 1.16, 1}
    };
};
private String[] currencies = {"USD", "EUR", "GBP"};
private JTextField amountField;
private JComboBox<String> fromComboBox;
private JComboBox<String> toComboBox;
private JLabel resultLabel;
public CurrencyConverter() {
    super("Currency Converter");
    setLayout(new FlowLayout());
    amountField = new JTextField(10);
    fromComboBox = new JComboBox<>(currencies);
    toComboBox = new JComboBox<>(currencies);
```



```

resultLabel = new JLabel("Result: ");
add(new JLabel("Amount:"));
add(amountField);
add(new JLabel("From:"));
add(fromComboBox);
add(new JLabel("To:"));
add(toComboBox);
add(new JButton(new ConvertAction()));
add(resultLabel);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setSize(300, 100);
setVisible(true);
}
private class ConvertAction extends AbstractAction {
    public ConvertAction() {
        super("Convert");
    }
    public void actionPerformed(ActionEvent e) {
        try {
            double amount = Double.parseDouble(amountField.getText());
            int fromIndex = fromComboBox.getSelectedIndex();
            int toIndex = toComboBox.getSelectedIndex();
            double result = amount * exchangeRates[fromIndex][toIndex];
            resultLabel.setText(String.format("Result: %.2f %s", result, currencies[toIndex]));
        } catch (NumberFormatException ex) {
            resultLabel.setText("Invalid amount");
        }
    }
}
public static void main(String[] args) {
    new CurrencyConverter();
}

```

OUTPUT:



3.AIM :

To develop a JavaScript code that validates a form consisting of fields like Name, Age, Address, EmailId, Hobby (checkbox), Gender (radio box), and Country (dropdown menu).

SOURCE CODE :

```

<html >

<head>

    <meta charset="UTF-8">

```

```

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Form Validation</title>
<script>
function validateForm() {
    let name = document.getElementById("name").value;
    if (name === "") {
        alert("empty");
        return false;
    }
    let age = document.getElementById("age").value;
    if (age === "" || age <= 0 || age > 120) {
        alert("invalid age");
        return false;
    }
    let address = document.getElementById("address").value;
    if (address === "") {
        alert("empty");
        return false;
    }
    let email = document.getElementById("email").value;
    let emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/; //use matches check
    if (!emailPattern.test(email)) {
        alert("invalid mail id");
        return false;
    }
    let hobbySelected = false;
    let hobbies = document.getElementsByName("hobby");
    for (let i = 0; i < hobbies.length; i++) {
        if (hobbies[i].checked) {
            hobbySelected = true;
            break;
        }
    }
}

```

```

    }
    if (!hobbySelected) {
        alert("Please select at least one hobby.");
        return false;
    }
    let genderSelected = false;
    let genders = document.getElementsByName("gender");
    for (let i = 0; i < genders.length; i++) {
        if (genders[i].checked) {
            genderSelected = true;
            break;
        }
    }
    if (!genderSelected) {
        alert("Please select a gender.");
        return false;
    }
    let country = document.getElementById("country").value;
    if (country === "") {
        alert("Please select a country.");
        return false;
    }
    alert("Form submitted successfully!");
    return true;
}
</script>
</head>
<body>
    <form name="myForm" onsubmit="return validateForm()">
        Name:
        <input type="text" id="name" name="name"><br><br>
        <label for="age">Age:</label>

```

```

<input type="text" id="age" name="age"><br><br>
<label for="address">Address:</label>
<input type="text" id="address" name="address"><br><br>
<label for="email">Email ID:</label>
<input type="text" id="email" name="email"><br><br>
<label>Hobby:</label><br>
<input type="checkbox" id="hobby1" name="hobby" value="Reading">
Reading<br>
<input type="checkbox" id="hobby2" name="hobby" value="Sports">
<label for="hobby2">Sports</label><br>
<input type="checkbox" id="hobby3" name="hobby" value="Traveling">
<label for="hobby3">Traveling</label><br><br>
<label>Gender:</label><br>
<input type="radio" id="male" name="gender" value="Male">
<label for="male">Male</label><br>
<input type="radio" id="female" name="gender" value="Female">
<label for="female">Female</label><br><br>
<label for="country">Country:</label>
<select id="country" name="country">
  <option value="">Select</option>
  <option value="USA">USA</option>
  <option value="India">India</option>
  <option value="UK">UK</option>
  <option value="Australia">Australia</option>
</select><br><br>
<input type="submit" value="Submit">
</form>
</body>
</html>

```

OUTPUT:

← → ↻ ⓘ File E:/java_practice/exp83.html ☆ 🌐 📁

☰ | 📧 Gm

This page says
empty

OK

Name:

Age:

Address:

Email ID:

Hobby:

☐ Reading

☐ Sports

☐ Traveling

Gender:

☐ Male

☐ Female

Country: ▼

☰ | 📧 Gm

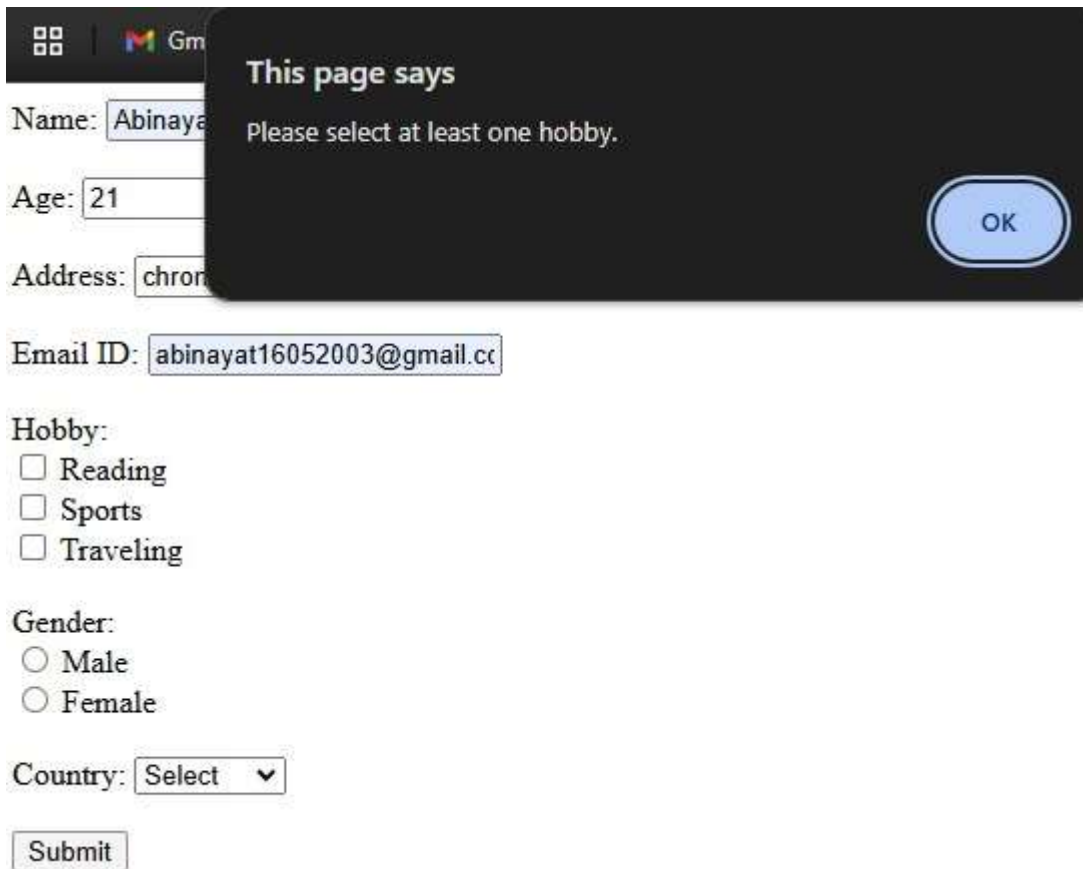
This page says
invalid age

OK

Name:

Age:

Address:



The image shows a web form with the following fields: Name (Abinaya), Age (21), Address (chrome), Email ID (abinayat16052003@gmail.co), and Hobby (Reading, Sports, Traveling). The Gender field has radio buttons for Male and Female. The Country field is a dropdown menu with 'Select' as the current value. A 'Submit' button is at the bottom. A dark overlay box with the title 'This page says' and the message 'Please select at least one hobby.' is displayed over the form, with an 'OK' button.

Name: Abinaya

Age: 21

Address: chrome

Email ID: abinayat16052003@gmail.co

Hobby:

☐ Reading

☐ Sports

☐ Traveling

Gender:

☐ Male

☐ Female

Country: Select

Submit

This page says

Please select at least one hobby.

OK

4.AIM :

To develop a JavaScript code that displays a custom context menu when the user right-clicks anywhere in the document.

SOURCE CODE :

```
<html>
<head>
  <title>Context Menu Example</title>
</head>
<body>
  <div id="context-menu">
    <ul>
      <li>Option 1</li>
      <li>Option 2</li>
      <li>Option 3</li>
    </ul>
  </div>
```

```

<script>
const contextMenu = document.getElementById('context-menu');
document.addEventListener('contextmenu', (event) => {
    event.preventDefault();
    contextMenu.style.top = event.clientY + 'px';
    contextMenu.style.left = event.clientX + 'px';
    contextMenu.style.display = 'block';
});
document.addEventListener('click', () => contextMenu.style.display = 'none');
</script>
</body>
</html>

```

OUTPUT:

- Option 1
- Option 2
- Option 3

5.AIM :

To write a simple jQuery function that increases the width and height of a <div> or button by 50 pixels when clicked.

SOURCE CODE :

```

<html >
<head>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <style>
        div, button {
            display: inline-block;
            width: 100px;
            height: 100px;
            margin: 10px;
            background-color: blue;

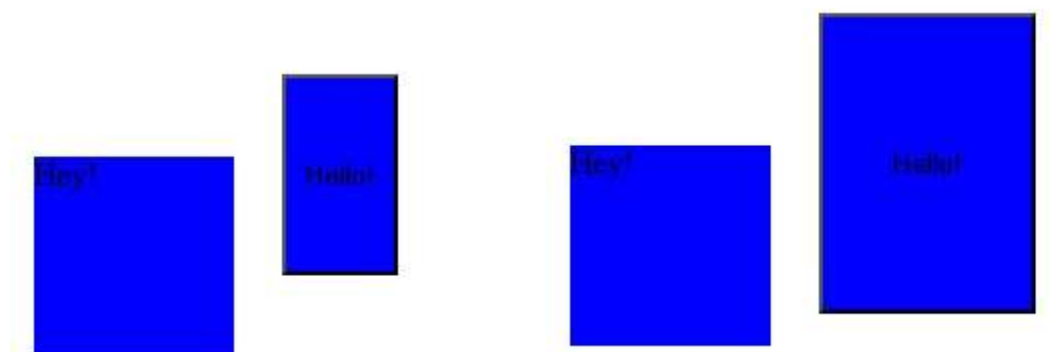
```

```

    }
    button {
        width: auto;
        padding: 10px;
    }
</style>
<script>
    $(document).ready(function() {
        // Increase size of div or button by 50 pixels when clicked
        $('div, button').click(function() {
            $(this).width($(this).width() + 50);
            $(this).height($(this).height() + 50);
        });
    });
</script>
</head>
<body>
    <div>Hey!</div>
    <button>Hello!</button>
</body>
</html>

```

OUTPUT:



6.AIM :

To write jQuery code that selects a paragraph with the class .intro, changes its text color to red, hides it, and fades it in over 2 seconds.

SOURCE CODE :

```
<html>

<head>

<title>jQuery Example</title>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<style>

    body {

        font-family: Arial, sans-serif;

    }

</style>

</head>

<body>

<p class="intro">Hello World!</p>

<script>

$(document).ready(function(){

    $('.intro')

    .css('color','red')

    .hide()

    .fadeIn(2000);

});

</script>

</body>

</html>
```

OUTPUT:

Hello World!



Hello World!

7.AIM :

To create a form with a text input and submit button, using jQuery to prevent the form from submitting if the input field is empty and displaying a message below the form

SOURCE CODE :

```
<html>
<head>
  <title>Form Validation</title>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
  <form id="myForm">
    <input type="text" id="textInput" placeholder="Enter some text">
    <button type="submit">Submit</button>
    <div id="errorMessage"></div>
  </form>
  <script>
    $(document).ready(function() {
      $('#myForm').submit(function(event) {
        var textInput = $('#textInput').val();
        if (textInput.trim() === "") {
          $('#errorMessage').text('Please enter some text');
          event.preventDefault();
        } else {
          $('#errorMessage').text("");
        }
      });
    });
  </script>
</body>
</html>
```

OUTPUT:

Please enter some text

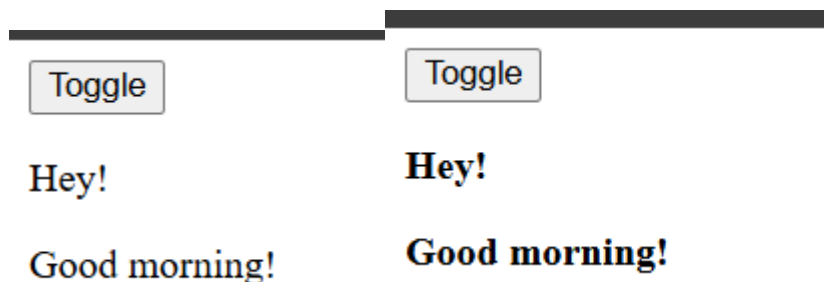
8.AIM :

To write jQuery code that toggles the font-weight of all paragraphs between normal and bold when a button is clicked.

SOURCE CODE :

```
<!DOCTYPE html>
<html>
<head>
  <title>Toggle Font Weight</title>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <style> .bold { font-weight: bold; }
</style>
</head>
<body>
  <button id="toggleButton">Toggle Font Weight</button>
  <p>This is a paragraph of text.</p>
  <p>This is another paragraph of text.</p>
  <p>This is yet another paragraph of text.</p>
  <script>
    $(document).ready(function() {
      $('#toggleButton').click(function() {
        $('p').toggleClass('bold');    });    });
  </script>
</body>
</html>
```

OUTPUT:



9.AIM :

To write jQuery code using AJAX to fetch user data from a data.json file, where each user object contains properties like id, name, and email, and logs the response to the console.

SOURCE CODE :

```
<html>

<head>

  <title>AJAX Example</title>

  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

  <h1>AJAX Example</h1>

  <div id="output"></div>

  <script>

    $.ajax({

      url: 'data.json',

      dataType: 'json',

      success: function(data) {

        console.log(data);

        $('#output').html(JSON.stringify(data));

      }

    });

  </script>

</body>

</html>
```

DATA.JSON:[

```
{

  "id": 1,

  "name": "Abinaya",

  "email": "abi@gmail.com"

},

{

  "id": 2,
```

```
"name": "Ahilan",  
"email": "ahilan@gmail.com"  
}  
]
```

OUTPUT:

User Data

Load User Data

Data fetched successfully:

```
[ { "id": 1, "name": "Abinaya", "email": "abi@gmail.com" }, { "id": 2, "name": "Ahilan", "email": "ahilan@gmail.com" } ]
```

10.AIM :

To create a dynamic product page for an e-commerce website, where users can filter products by categories (e.g., electronics, clothing, home goods) using AJAX requests without reloading the page.

SOURCE CODE :

```
<html>  
<head>  
  <title>Dynamic Product Page</title>  
</head>  
<body>  
  <h1>Products</h1>  
  <select id="category">  
    <option value="">All</option>  
    <option value="mobiles">Mobiles</option>  
    <option value="clothing">Clothing</option>  
    <option value="groceries">Groceries</option>  
  </select>  
  <div id="products"></div>  
  
  <script>  
    const products = [  
      {id: 1, name: "Vivo V40", category: "electronics"},
```

```

{id: 2, name: "Samsung TV", category: "electronics"},
{id: 3, name: "Nike Shoes", category: "clothing"},
{id: 4, name: "Corn Flour", category: "groceries"}
];

const categorySelect = document.getElementById("category");
const productsDiv = document.getElementById("products");

categorySelect.addEventListener("change", () => {
  const selectedCategory = categorySelect.value;
  const filteredProducts = products.filter(product => product.category === selectedCategory || selectedCategory === "");
  let productHtml = "";
  filteredProducts.forEach(product => {
    productHtml += `<p>${product.name}</p>`;
  });
  productsDiv.innerHTML = productHtml;
});
categorySelect.dispatchEvent(new Event("change"));
</script>
</body>
</html>

```

OUTPUT:

All ▼

Vivo V40
Samsung TV
Nike Shoes
Corn Flour

Groceries ▼
Corn Flour

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Implemented a Java Swing program to efficiently process large server log files with a user interface for filtering and file handling, and deserialize network data packets using object-oriented principles for analysis.

EXP NO : 9	Java server based web applications
DATE : 17/11/24	

1.AIM :

To Develop a code for creating cookies with the name of the person and secret code at the server, after getting these details from the client using HTML form. Use another servlet program to get the information stored in the cookie and display the details.

SOURCE CODE :

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>User Form</title>
</head>
<body>
  <fieldset>
    <legend>User details</legend>
    <form action = "create" method = "POST">
      <label for="name">Name:
      <input type="text" name= "name" id="name">
      </label>
      <br><br>
      <label for = "secret">Secret Code:
        <input type ="text" name = "secret" id = "secret">
      </label>
      <br>
      <br>
      <input type = "submit" value ="Create Cookie">
    </form>
  </fieldset>
</body>
</html>
```

Web.xml

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0">
  <!-- Servlet1 Mapping -->
  <servlet>
    <servlet-name>Servlet1</servlet-name>
    <servlet-class>Servlet1</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Servlet1</servlet-name>
    <url-pattern>/create</url-pattern>
  </servlet-mapping>
```



```

<!-- DisplayServlet1 Mapping -->
<servlet>
    <servlet-name>DisplayServlet1</servlet-name>
    <servlet-class>DisplayServlet1</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>DisplayServlet1</servlet-name>
    <url-pattern>/display</url-pattern>
</servlet-mapping>
</web-app>

```

Servlet1.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
public class Servlet1 extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException {
        String name = request.getParameter("name");
        String secret = request.getParameter("secret");
        Cookie cookie1 = new Cookie("name",name);
        Cookie cookie2 = new Cookie("secret", secret);
        cookie1.setMaxAge(60*60);
        cookie2.setMaxAge(60*60);
        response.addCookie(cookie1);
        response.addCookie(cookie2);
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");
        out.println("<html>");
        out.println("<body>");
        out.println("<p>Cookies have been set successfully name : " +name + " </p>");
        out.println("<a href = '/DEMO/display'>Display Cookies</a>");
        out.println("</body>");
        out.println("</html>");
    }
}

```

DisplayServlet.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

public class DisplayServlet1 extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {

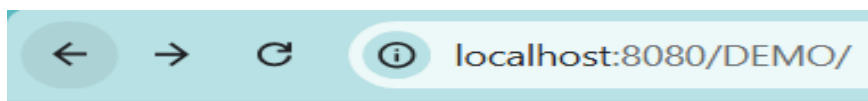
```

```

Cookie[] cookies = request.getCookies();
PrintWriter out = response.getWriter();
String name = null, secret = null;
for(Cookie cookie : cookies) {
    if(cookie.getName().equals("name")){
        name = cookie.getValue();    }
    else if(cookie.getName().equals("secret")){
        secret = cookie.getValue();    }    }
out.println("<html>");
out.println("<body>");
out.println("<h1> Here are the cookies! </h1>");
out.println("<p> Name: " + name + "</p>");
out.println("<p> Secret: " + secret + "</p>");
out.println("</body>");
out.println("</html>");    }}

```

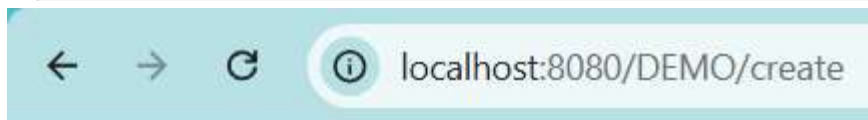
OUTPUT:



User details

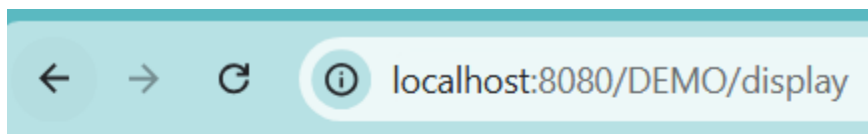
Name:

Secret Code:



Cookies have been set successfully name : babu

[Display Cookies](#)



Here are the cookies!

Name: babu

Secret: Hello

2.AIM :

To Develop a simple User Login system using Java Servlets. Design a registration page where users can enter their name, password, and email. Upon submission, the data will be stored in a database. After successful registration, it redirects to a login page where the users can enter their username and password which will be checked against the data in the database. If the details are valid, a session is created for the user and then the user's profile page will be displayed. When the user clicks the logout button, the user's session will be invalidated and redirected to the login page.

SOURCE CODE:

Web.xml:

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    version="3.0">
    <!-- SignupServlet Mapping -->
    <servlet>
        <servlet-name>DatabaseConnection</servlet-name>
        <servlet-class>DatabaseConnection</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>DatabaseConnection</servlet-name>
        <url-pattern>/babu</url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>SignupServlet</servlet-name>
        <servlet-class>SignupServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>SignupServlet</servlet-name>
        <url-pattern>/signup</url-pattern>
    </servlet-mapping>
    <!-- Login Mapping -->
    <servlet>
        <servlet-name>Login</servlet-name>
        <servlet-class>Login</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>Login</servlet-name>
        <url-pattern>/login</url-pattern>
    </servlet-mapping>
    <!-- UserProfile Mapping -->
    <servlet>
        <servlet-name>UserProfile</servlet-name>
        <servlet-class>UserProfile</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>UserProfile</servlet-name>
        <url-pattern>/userdetails</url-pattern>
```

```
</servlet-mapping>
</web-app>
```

Signup.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>User Registration</title>
</head>
<body>
  <fieldset>
    <legend>User Registration form</legend>
    <form action = "signup" method = "POST">
      <label for = "username"> Username:
        <input type = "text" id = "username" name = "username">
      </label> <br><br>
      <label for = "password"> Password:
        <input type = "password" id = "password" name = "password">
      </label> <br><br>
      <label for = "mail"> Email:
        <input type = "email" id = "mail" name = "mail">
      </label> <br><br>
      <input type = "submit" value = "Register">
    </form>
  </fieldset>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Login</title>
</head>
<body>
  <fieldset>
    <legend>Login</legend>
    <form action = "login" method = "POST">
      <label for = "username"> Username:
        <input type = "text" id = "username" name = "username">
      </label> <br><br>
      <label for = "password"> Password:
        <input type = "password" id = "password" name = "password">
      </label> <br><br>
      <input type = "submit" value = "Login">
    </form>
  </fieldset>
</body>
</html>
```

```
</form>
</fieldset>
</body>
</html>
```

Login.java

```
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.*;

public class Login extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException {
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        Connection con = null;
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            PrintWriter out = response.getWriter();
            response.setContentType("text/html");
            con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers?useSSL=false&allowPublicKeyRetrieval=true&se
rverTimezone=UTC","root","");
            String sql = "select * from users where username = ? and password = ?";
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1, username);
            ps.setString(2, password);
            ResultSet rs = ps.executeQuery();
            if(rs.next()){
                HttpSession session = request.getSession();
                session.setAttribute("username", username);
                out.println("Login Successful<br>");
                out.println("Go to <a href = '/P2/userdetails'>User Profile</a> ");
            }
            else{
                out.println("Login Credentials does not exist");
            }
        }catch (SQLException e) {
            response.getWriter().println("Database error: " + e.getMessage());
            e.printStackTrace();
        } catch (Exception e) {
            response.getWriter().println("Error: " + e.getMessage());
            e.printStackTrace();    } }}
}
```

UserProfile.java

```
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.*;

public class UserProfile extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
        HttpSession session = request.getSession();
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String username = (String) session.getAttribute("username");
        Connection con = null;
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC","root","");
            String sql = "select * from users where username=?";
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1,username);
            ResultSet rs = ps.executeQuery();
            String email = "";
            if(rs.next()){
                email = rs.getString("email");
            }
            out.println("<div border = 'black 1px'>");
            out.println("<h1>"+ "User Details" + "</h1>");
            out.println("Username: "+username);
            out.println("<br><br>Email: "+email);
            // out.println("<br> Go to <a href = '/Servlet/products'>Products</a>");
            out.println("</div>");
        }catch (SQLException e) {
            response.getWriter().println("Database error: " + e.getMessage());
            e.printStackTrace();
        } catch (Exception e) {
            response.getWriter().println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}
```

SignupServlet.java

```
import java.io.Console;
```

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
public class SignupServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException {
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        String email = request.getParameter("mail");
        response.getWriter().println("Page Loaded");
        Connection con = null;
        try{
            response.setContentType("text/html");
            Class.forName("com.mysql.cj.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers?useSSL=false&allowPublicKeyRetrieval=true&
serverTimezone=UTC","root","");
            response.getWriter().println("Connection Established!");
            String insertSql = "insert into users values(?,?,?)";
            PreparedStatement ps = con.prepareStatement(insertSql);
            ps.setString(1, username);
            ps.setString(2, password);
            ps.setString(3, email);
            ps.executeUpdate();
            response.getWriter().println("<p color='green'>User Registered Successfully</p><br><a href = '/P2/login.html'>Go
to Login</a><br>");
            con.close();
        }catch (SQLException e) {
            response.getWriter().println("Database error: " + e.getMessage());
            e.printStackTrace();
        } catch (Exception e) {
            response.getWriter().println("Error: " + e.getMessage());
            e.printStackTrace();
        }
    }
}

```

OUTPUT:

Two browser screenshots are shown side-by-side. The left browser window displays the 'User Registration form' at 'localhost:8080/P2/signup.html'. It contains three input fields: 'Username' with the value 'Babu_Mohan', 'Password' with masked characters '.....', and 'Email' with the value 'babu@gmail'. Below these fields is a 'Register' button. The right browser window displays the 'Login' form at 'localhost:8080/P2/login.html'. It contains two input fields: 'Username' with the value 'Babu Mohan' (underlined in red) and 'Password' with masked characters '.....'. Below these fields is a 'Login' button.

Page Loaded Connection Established!

User Registered Successfully

[Go to Login](#)

User Details

Username: Babu_Mohan

Email: babu@gmail

3.AIM :

To Create a Shopping Cart System using Java Servlets that allows users to:

View products from a product catalog.

Add products to their shopping cart.

View and modify their cart (update quantity or remove items).

Checkout by providing their details, which will be stored in the database.

SOURCE CODE:

Web.xml

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0">
```

```
<servlet>
```



```

    <servlet-name>SignupServlet</servlet-name>
    <servlet-class>SignupServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>SignupServlet</servlet-name>
    <url-pattern>/signup</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>Login</servlet-name>
    <servlet-class>Login</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Login</servlet-name>
    <url-pattern>/login</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>UserProfile</servlet-name>
    <servlet-class>UserProfile</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>UserProfile</servlet-name>
    <url-pattern>/userdetails</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>ProductServlet</servlet-name>
    <servlet-class>ProductServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>ProductServlet</servlet-name>
    <url-pattern>/products</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>CartServlet</servlet-name>
    <servlet-class>CartServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>CartServlet</servlet-name>
    <url-pattern>/cart</url-pattern>
</servlet-mapping>
<servlet>
    <servlet-name>ViewCartServlet</servlet-name>
    <servlet-class>ViewCartServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>ViewCartServlet</servlet-name>
    <url-pattern>/viewcart</url-pattern>
</servlet-mapping>

```

```

<servlet>
  <servlet-name>CheckoutServlet</servlet-name>
  <servlet-class>CheckoutServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>CheckoutServlet</servlet-name>
  <url-pattern>/checkout</url-pattern>
</servlet-mapping>
<servlet>
  <servlet-name>ConfirmationServlet</servlet-name>
  <servlet-class>ConfirmationServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>ConfirmationServlet</servlet-name>
  <url-pattern>/confirmation</url-pattern>
</servlet-mapping>
<servlet>
  <servlet-name>EditCartServlet</servlet-name>
  <servlet-class>EditCartServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>EditCartServlet</servlet-name>
  <url-pattern>/editcart</url-pattern>
</servlet-mapping>
</web-app>

```

CartServlet.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class CartServlet extends HttpServlet {
  protected void doPost(HttpServletRequest request, HttpServletResponse response){
    String username= request.getSession().getAttribute("username").toString();
    Connection con = null;
    response.setContentType("text/html");
    int count = Integer.parseInt(request.getParameter("count"));
    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers","root","");
      String sql = "insert into cart values(?, ?, ?)";
      PreparedStatement ps = con.prepareStatement(sql);

```

```

for (int i = 1; i <= count; i++) {
    String quantity = request.getParameter("quantity"+i);
    if("0".equals(quantity)){
        continue;
    }

    String pid = request.getParameter("pid"+i);
    PreparedStatement psq = con.prepareStatement("select * from cart where pid = ? and username = ?");
    psq.setInt(1, Integer.parseInt(pid));
    psq.setString(2, username);
    ResultSet rs = psq.executeQuery();
    if(rs.next()) {
        continue;
    }
    String price = request.getParameter("price"+i);
    ps.setString(1, username);
    ps.setInt(2, Integer.parseInt(pid));
    ps.setInt(3, Integer.parseInt(quantity));
    ps.executeUpdate();
}
response.getWriter().println("Items added to the card successfully");
response.getWriter().println("Go to <a href = '/DEMO/viewcart'>Cart</a>");
}catch (Exception e){}
}
}

```

CheckoutServlet.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.*;

public class CheckoutServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        HttpSession session = request.getSession();
        String username = (String) session.getAttribute("username");

        try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers", "root", "")) {
            String query = "SELECT p.name, p.description, p.price, c.quantity, (p.price * c.quantity) AS total_price " +
                "FROM products p INNER JOIN cart c ON p.pid = c.pid WHERE c.username = ?";

```

```

        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, username);
        ResultSet rs = ps.executeQuery();
        out.println("<h2>Checkout Page</h2>");
        out.println("<table
border='1'><tr><th>Product</th><th>Description</th><th>Price</th><th>Quantity</th><th>Total</th></tr>");
        int grandTotal = 0;
        while (rs.next()) {
            String name = rs.getString("name");
            String description = rs.getString("description");
            int price = rs.getInt("price");
            int quantity = rs.getInt("quantity");
            int totalPrice = rs.getInt("total_price");

            out.println("<tr><td>" + name + "</td><td>" + description + "</td><td>" + price + "</td><td>" + quantity +
"</td><td>" + totalPrice + "</td></tr>");
            grandTotal += totalPrice;
        }

        out.println("</table>");
        out.println("<h3>Grand Total: Rs." + grandTotal + "</h3>");

        out.println("<form action='confirmation' method='post'>");
        out.println("Full Name: <input type='text' name='fullName' required><br>");
        out.println("Address: <input type='text' name='address' required><br>");
        out.println("Email: <input type='email' name='email' required><br>");
        out.println("<input type='hidden' name='grandTotal' value='" + grandTotal + "'>");
        out.println("<input type='submit' value='Confirm and Checkout'>");
        out.println("</form>");

    } catch (SQLException e) {
        e.printStackTrace(out);
    }
}
}

```

Confirmation.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

public class ConfirmationServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException {
        response.setContentType("text/html");
    }
}

```

```

    PrintWriter out = response.getWriter();

    String fullName = request.getParameter("fullName");
    String address = request.getParameter("address");
    String email = request.getParameter("email");
    int grandTotal = Integer.parseInt(request.getParameter("grandTotal"));

    out.println("<h2>Order Confirmation</h2>");
    out.println("<p>Thank you, <b>" + fullName + "</b>, for your order!</p>");
    out.println("<p>Shipping to: " + address + "</p>");
    out.println("<p>Email: " + email + "</p>");
    out.println("<p><b>Total Amount Paid: Rs." + grandTotal + "</b></p>");
    out.println("<p>Your order has been confirmed and will be processed shortly.</p>");
}
}

```

EditCartServlet.java

```

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class EditCartServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
        HttpSession session = request.getSession();
        String username = (String) session.getAttribute("username");

        Connection con = null;
        response.setContentType("text/html");
        response.getWriter().println("<h1>Edit Cart</h1>");
        response.getWriter().println("<form action='editcart' method='post'>");
        response.getWriter().println("<table border='1'>");
        response.getWriter().println("<tr><th>Product  
Name</th><th>Description</th><th>Price</th><th>Quantity</th><th>Total</th><th>Actions</th></tr>");

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers", "root", "");

```

```

String sql = "SELECT p.pid, p.name, p.description, p.price, c.quantity, (p.price * c.quantity) AS total " +
    "FROM cart c JOIN products p ON c.pid = p.pid WHERE c.username = ?";
PreparedStatement ps = con.prepareStatement(sql);
ps.setString(1, username);
ResultSet rs = ps.executeQuery();

while (rs.next()) {
    int pid = rs.getInt("pid");
    String name = rs.getString("name");
    String description = rs.getString("description");
    double price = rs.getDouble("price");
    int quantity = rs.getInt("quantity");
    double productTotal = rs.getDouble("total");

    response.getWriter().println("<tr>");
    response.getWriter().println("<td>" + name + "</td>");
    response.getWriter().println("<td>" + description + "</td>");
    response.getWriter().println("<td>" + price + "</td>");
    response.getWriter().println("<td><input type='number' name='quantity_' + pid + "' value='" + quantity + '"
min='0'></td>");
    response.getWriter().println("<td>" + productTotal + "</td>");
    response.getWriter().println("<td><button type='submit' name='remove' value='" + pid +
">Remove</button></td>");
    response.getWriter().println("</tr>");
}

response.getWriter().println("</table>");
response.getWriter().println("<button type='submit' name='update' value='true'>Update Cart</button>");
response.getWriter().println("</form>");
} catch (Exception e) {
    e.printStackTrace();
    response.getWriter().println("<p>Error loading cart.</p>");
}

}

```

```

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {

```

```

    HttpSession session = request.getSession();
    String username = (String) session.getAttribute("username");

    Connection con = null;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers", "root", "");

        if (request.getParameter("remove") != null) {

```

```

        int pidToRemove = Integer.parseInt(request.getParameter("remove"));
        String deleteSql = "DELETE FROM cart WHERE username = ? AND pid = ?";
        PreparedStatement deleteStmt = con.prepareStatement(deleteSql);
        deleteStmt.setString(1, username);
        deleteStmt.setInt(2, pidToRemove);
        deleteStmt.executeUpdate();
    } else if ("true".equals(request.getParameter("update"))) {
        String getPidsSql = "SELECT pid FROM cart WHERE username = ?";
        PreparedStatement getPidsStmt = con.prepareStatement(getPidsSql);
        getPidsStmt.setString(1, username);
        ResultSet pidsResult = getPidsStmt.executeQuery();

        while (pidsResult.next()) {
            int pid = pidsResult.getInt("pid");
            String quantityParam = request.getParameter("quantity_" + pid);
            if (quantityParam != null) {
                int newQuantity = Integer.parseInt(quantityParam);
                String updateSql = "UPDATE cart SET quantity = ? WHERE username = ? AND pid = ?";
                PreparedStatement updateStmt = con.prepareStatement(updateSql);
                updateStmt.setInt(1, newQuantity);
                updateStmt.setString(2, username);
                updateStmt.setInt(3, pid);
                updateStmt.executeUpdate();
            }
        }
    }

    response.sendRedirect("/DEMO/viewcart");
} catch (Exception e) {
    e.printStackTrace();
    response.getWriter().println("<p>Error updating cart.</p>");
}
}
}

```

ProductServlet.java

```

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

```

```

import java.sql.ResultSet;

public class ProductServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html><head><title>Product Page</title></head><body>");
        out.println("<h1>Product List</h1>");
        out.println("<form action='cart' method='post'>");
        out.println("<table border='1'><tr><th>Product
ID</th><th>Name</th><th>Description</th><th>Price</th><th>Quantity</th></tr>");

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers", "root", "");
            String query = "SELECT * FROM products";
            PreparedStatement ps = con.prepareStatement(query);
            ResultSet rs = ps.executeQuery();
            int i = 1;
            while (rs.next()) {
                int pid = rs.getInt("pid");
                String name = rs.getString("name");
                String description = rs.getString("description");
                int price = rs.getInt("price");
                out.println("<tr>");
                out.println("<td>" + pid + "</td>");
                out.println("<td>" + name + "</td>");
                out.println("<td>" + description + "</td>");
                out.println("<td>" + price + "</td>");
                out.println("<td>");

                out.println("<input type='hidden' name='pid' + i+'' value='' + pid + ''>");
                out.println("<input type='hidden' name='price' + i +'' value='' + price + ''>");
                out.println("<input type='number' name='quantity' + i+'' value='0' min='0'>");

                out.println("</td>");
                out.println("</tr>");
                i++;
            }
            out.println("</table>");
            i = i-1;
            out.println("<input type = 'hidden' name = 'count' value = '' + i + ''>");
            out.println("<input type = 'submit' name = 'submit' value='Add to Cart'>");
            out.println("</form>");
            out.println("</body></html>");
        } catch (Exception e) {

```



```

        e.printStackTrace();
        out.println("<p>Error retrieving products.</p>");
    } finally {
        out.close();
    }
}
}

```

viewCartServlet.java

```

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class ViewCartServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException {
        Connection con = null;
        double totalCartPrice = 0;
        PrintWriter out = response.getWriter();
        response.setContentType("text/html");
        out.println("<h1>Shopping Cart</h1>");
        out.println("<table border='1'>");
        out.println("<tr><th>Product  
Name</th><th>Description</th><th>Price</th><th>Quantity</th><th>Total</th></tr>");

        String username = request.getSession().getAttribute("username").toString();
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/servletUsers", "root", "");

            String sql = "SELECT p.name, p.description, p.price, c.quantity, (p.price * c.quantity) AS total " +
                "FROM cart c JOIN products p ON c.pid = p.pid WHERE c.username = ?";
            PreparedStatement ps = con.prepareStatement(sql);
            ps.setString(1, username);

            ResultSet rs = ps.executeQuery();

            while (rs.next()) {
                String name = rs.getString("name");
                String description = rs.getString("description");
                double price = rs.getDouble("price");
                int quantity = rs.getInt("quantity");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

```

double productTotal = rs.getDouble("total");

totalCartPrice += productTotal;

response.getWriter().println("<tr>");
response.getWriter().println("<td>" + name + "</td>");
response.getWriter().println("<td>" + description + "</td>");
response.getWriter().println("<td>" + price + "</td>");
response.getWriter().println("<td>" + quantity + "</td>");
response.getWriter().println("<td>" + productTotal + "</td>");
response.getWriter().println("</tr>");
}

response.getWriter().println("</table>");
response.getWriter().println("<h3>Total Cart Price: " + totalCartPrice + "</h3>");
out.println("<a href = '/DEMO/editcart'>Edit Cart</a>");
out.println("<a href = '/DEMO/checkout'>Checkout</a>");

} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

OUTPUT:

User Details

Username: Bba

Email: bb@sdsd.sdf

Go to [Products](#)

Product List

Product ID	Name	Description	Price	Quantity
1	pen	INK_PEN	30	0
1	pen	INK_PEN	30	0
2	Pencil	Lead_pencil	10	0
3	Eraser	--	5	0
4	Sharpner	For_pencil	5	0
5	Scale	--	15	0

Add to Cart

Items added to the card successfully [Go to Cart](#)

Shopping Cart

Product Name	Description	Price	Quantity	Total
pen	INK_PEN	30.0	1	30.0
pen	INK_PEN	30.0	1	30.0
Pencil	Lead_pencil	10.0	2	20.0
Eraser	--	5.0	2	10.0
Scale	--	15.0	4	60.0

Total Cart Price: 150.0

[Edit Cart](#) [Checkout](#)

Edit Cart

Product Name	Description	Price	Quantity	Total	Actions
Pencil	Lead_pencil	10.0	<input type="text" value="5"/>	20.0	<button>Remove</button>
Eraser	--	5.0	<input type="text" value="2"/>	10.0	<button>Remove</button>
Scale	--	15.0	<input type="text" value="4"/>	60.0	<button>Remove</button>

Update Cart

Checkout Page

Product	Description	Price	Quantity	Total
pen	INK_PEN	30	1	30
pen	INK_PEN	30	1	30
Pencil	Lead_pencil	10	2	20
Eraser	--	5	2	10
Scale	--	15	4	60

Grand Total: Rs.150

Full Name:

Address:

Email:

Order Confirmation

Thank you, **Babu M**, for your order!

Shipping to: 3H/2 Subramaniapuram

Email: babum@gmail.com

Total Amount Paid: Rs.150

Your order has been confirmed and will be processed shortly.

Course outcomes	Observation	Record
CO1 : Ability for problem definition and realization	NA/10
CO2: Ability to design and analysis	NA/10
CO3: Ability to implement and validate	NA/10

Result :

Hence Java server based web applications implemented successfully.