

No.	Practical	Date	Sign
1	Matrix multiplication n x m		
2	Rain water logging problem		
3	Staircase Problem		
4	Classroom Problem		
5	To find prime and twin prime with multithreading.		
6	Demonstrate wait() and notifyAll()		
7	Client Server using Java Sockets		
8	Object Oriented Concepts		
9	Inner class example		
10	Collection framework example.		

## Practical 1: Matrix multiplication $n \times m$

```
package matrixmul;
import java.util.Scanner;
/**
 *
 * @author Manasvi
 */
public class MatrixMultiplication {
    public void getInput(double matrix[][],String name){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter value for "+name);
        /**
         Here matrix1.length returns number of rows for the matrix
         and matrix1[0].length returns number of columns for matrix
        */
        for(int i=0;i<matrix.length;i++){
            for(int j=0;j<matrix[0].length;j++){
                matrix[i][j]=sc.nextDouble();
            }
        }
    }
    double multiplyMatrices(double [][]matrix1,double [][]matrix2, int row,int col){
        /**
         Here we compute value for each cell one at a time
         as it loops around matrix2 row and simultaneously
         multiplies each column of matrix1 to each row of matrix2
        */
        double value=0;
        for(int i=0;i<matrix2.length;i++){
            value+=matrix1[row][i] * matrix2[i][col];
        }
        return value;
    }
}
```

```

public static void main(String arg[]){

    MatrixMultiplication mat=new MatrixMultiplication();
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the matrix1 size");
    int r=sc.nextInt();
    int c=sc.nextInt();
    double matrix1[][]=new double[r][c];

    System.out.println("Enter the matrix2 size");
    r=sc.nextInt();
    c=sc.nextInt();
    double matrix2[][]=new double[r][c];

    //enter values for matrix1 and matrix2
    mat.getInput(matrix1, "Matrix 1");
    mat.getInput(matrix2, "Matrix 2");

    double [][]result=new double[matrix1.length][matrix2[0].length];
    for(int row=0;row<result.length;row++){
        for(int col=0;col<result[row].length;col++){
            result[row][col]=mat.multiplyMatrices(matrix1, matrix2, row, col);
            System.out.print(result[row][col]+" ");
        }
        System.out.println();
    }

}
}

```

#### Output - SyscsJavaProblems (run)

```

run:
Enter the matrix1 size
3 2
Enter the matrix2 size
2 3
Enter value for Matrix 1
1 10
2 20
3 30
Enter value for Matrix 2
3 5 7
2 4 6
23.0 45.0 67.0
46.0 90.0 134.0
69.0 135.0 201.0
BUILD SUCCESSFUL (total time: 1 minute 30 seconds)

```

## Practical 2: Rain water logging problem

```
package rainwater;

import java.util.Scanner;

/**
 *
 * @author Manasvi
 */
public class RainWaterStorage {
    //To find maximum between given array indexes
    int findmax(int array[], int start, int end ){
        int max=array[start];
        for(int i=start;i<=end;i++){
            if(max<array[i]){
                max=array[i];
            }
        }
        return max;
    }

    public static void main(String arg[]){

        RainWaterStorage rw=new RainWaterStorage();
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int pillar[]=new int[n];
        for(int i=0;i<n;i++){
            pillar[i]=sc.nextInt();
        }

        int totalwater=0,storeupto,w,lookleft,lookright;
```

```

int totalwater=0,storeupto,w,lookleft,lookright;

/*
the idea is to look left of the current element find the largest one
then look right for the same,

then find minimum between them
and compare it with current element

if greater than the difference between minimum and current element
must be the amount of water stored.

    |
    |
  | | | |
  | | |
  | _ | | |
  3 0 5 2 3

so if we are at 0
from left 3 is largest
from right 5 is largest
now the minimum from (3,5)==> 3
and since it is > 0 (current element)
difference = 3-0 = 3 (water stored)
*/

for(int i=0;i<n-1;i++){

    lookleft=rw.findmax(pillar, 0, i-1);
    lookright=rw.findmax(pillar, i+1, n-1);

    storeupto=Math.min(lookleft, lookright);
    if(storeupto>pillar[i]){
        w=storeupto-pillar[i];
        totalwater+=w;
    }
}

System.out.println("Total water stored "+totalwater);
}
}

```

```

Output
Debugger Console X SycsJavaProblems (run) X
run:
5
3 0 5 2 3
Total water stored 4
BUILD SUCCESSFUL (total time: 17 seconds)

```

## Practical 3: Staircase Problem

```
package staircase;

import java.util.Scanner;

/**
 *
 * @author Manasvi
 */
public class StaircaseProblem {

    /**
     The problem here is given the number of stairs
     how many ways using either 1 or 2 steps at time we can reach from
     bottom to top.

     if stairs = 4
         |
         |   |
         |   |   |
         |   |   |   |
         |   |   |   |___|...
         |   |   |___|.....
         |   |___|.....
         |___|.....
         |.....
     then, considering 1 or 2 steps at a time,
     1,1,1,1
     1,1,2
     1,2,1
     2,1,1
     2,2
     which is 5 ways,
     Solving it is simple fibonacci series upto stairs+1
     */

    int numberOfWays(int n){

        if(n <= 1)
            return n;
        return numberOfWays(n-1)+numberOfWays(n-2);

    }

    public static void main(String []args){





        StaircaseProblem sp=new StaircaseProblem();
        Scanner sc=new Scanner(System.in);
        System.out.println("How many stairs ?? ");
        int stairs = sc.nextInt();
        System.out.println("Number of ways to reach top"
            + " with 1 or 2 step is "+sp.numberOfWays(stairs+1));

    }

}
```

Output

Debugger Console × SycsJavaProblems (run) ×

```
run:  
How many stairs ??  
4  
Number of ways to reach top with 1 or 2 step is 5  
BUILD SUCCESSFUL (total time: 2 seconds)
```

## Practical 4: Classroom Problem

```
package snapchat;

import java.util.Arrays;
import java.util.Scanner;

/**
 *
 * @author Manasvi
 */
public class ClassroomProblem {
    /**
     *
     * Given an array of time intervals (start, end) for classroom lectures (possibly overlapping),
     * find the minimum number of rooms required.
     * For example, given [(30, 75), (0, 50), (60, 150)], you should return 2.
     *
     * How to solve it ??
     * >> Take arrays for both all start time and all end time.
     * | start==>{30,0,60} end==>{75,50,150}
     * >> sort both of them and simply loop through both at same time.
     * >> keep a counter for classroom count
     * | 0 30 60 50 75 150
     * | if element is from start array increment classroom count
     * | or else decrement classroom count.
     * | Make sure to keep track of highest number of classroom count
     * | that would be the required solution.
     * | 0 +1 c=1 high=1
     * | 30 +1 c=2 high=2
     * | 50 -1 c=1 high=2
     * | 60 +1 c=2 high=2
     * | 75 -1 c=1 high=2
     * | 150 -1 c=0 high=2
     *
     * | Number of classroom required = 2
     */

    public static void main(String arg[]){

        Scanner sc=new Scanner(System.in);

        System.out.println("Enter number of lectures : ");
        int numberOfLecures=sc.nextInt();
        int start[]=new int[numberOfLecures];
        int end[]=new int[numberOfLecures];
```



```

System.out.println("Enter start time and respective end time of each lecture \n");
for(int i=0;i<numberOfLecures;i++){
    start[i]=sc.nextInt();
    end[i]=sc.nextInt();
}
//sortin arrays using inbuilt java function
Arrays.sort(start);
Arrays.sort(end);

//counter for classroom and highest among them
int c=0,high=c;

//loop counter for start and end loops
int i=0,j=0;
while(i<numberOfLecures && j<numberOfLecures){

    if(start[i]<end[j]){
        c++;
        i++;
        if(c>high){
            high=c;
        }
    }else if(start[i]>end[j]){
        c--;
        j++;
    }
}
System.out.println("Number of classrooms required = "+high);
}
}

```

## Output

Debugger Console x SycsJavaProblems (run) x

```

run:
Enter number of lectures :
3
Enter start time and respective end time of each lecture

30 70
0 50
60 150
Number of classrooms required = 2
BUILD SUCCESSFUL (total time: 17 seconds)

```

## Practical 5: To find prime and twin prime with multithreading.

```
public class Multithread {  
  
    static int num;  
    static class Prime extends Thread{  
        synchronized boolean prime(int n){  
            int h=n/2;  
            int i=2,c=0;  
            while(i<=h){  
                if(n%i==0){  
                    c++;  
                }  
                if(c>0){  
                    return false;  
                }  
                i++;  
            }  
            return true;  
        }  
    }  
    public void run(){  
        if(prime(num)){  
            System.out.println(num+" is prime");  
        }else{  
            System.out.println(num+" is not a prime");  
        }  
    }  
}
```

```

static class Twinprime extends Thread{
    String checkForTwinPrime(){
        Prime p=new Prime();
        if(p.prime(num)){
            if(p.prime(num+2)){
                return (num+2)+" is its twin prime";
            }else if(p.prime(num-2)){
                return (num-2)+" is its twin prime";
            }
        }
        return num+" does not have a twin prime";
    }
    public void run(){
        System.out.println(checkForTwinPrime());
    }
}

public static void main(String arg[]){
    Scanner sc =new Scanner(System.in);
    num=sc.nextInt();

    /*
    Here we will check for prime and its win prime if there exists one
    at the same time using multithreading.

    Twin prime is prime number at difference of two
    so if 73 is prime than + or - 2 is 71 and 75,
    where 71 is prime so 71 ,73 are twin primes

    */
    Prime p=new Prime();
    p.start();
    Twinprime t=new Twinprime();
    t.start();
}
}

```

```

Output - SycsJavaProblems (run)
run:
71
71 is prime
73 is its twin prime
BUILD SUCCESSFUL (total time: 4 seconds)

```

```

Output - SycsJavaProblems (run)
run:
75
75 is not a prime
75 does not have a twin prime
BUILD SUCCESSFUL (total time: 4 seconds)

```

## Practical 6: Demonstrate wait() and notifyAll()

```
public class Data {
    private String packet;

    // True if receiver should wait
    // False if sender should wait
    private boolean transfer = true;
    public synchronized void send(String packet) {
        while (!transfer) {
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted"+e.getMessage());
            }
        }
        transfer = false;
        this.packet = packet;
        notifyAll();
    }
    public synchronized String receive() {
        while (transfer) {
            try {
                wait();
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted"+e.getMessage());
            }
        }
        transfer = true;
        notifyAll();
        return packet;
    }
}
```

```

static class Sender implements Runnable {
    private Data data;
    // standard constructors
    Sender(Data data){
        this.data=data;
    }
    public void run() {
        String packets[] = {
            "First packet",
            "Second packet",
            "Third packet",
            "Fourth packet",
            "End"
        };

        for (String packet : packets) {
            data.send(packet);
            // Thread.sleep() to mimic heavy server-side processing
            try {
                Thread.sleep(ThreadLocalRandom.current().nextInt(1000, 5000));
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted"+e.getMessage());
            }
        }
    }
}

```

```

static class Receiver implements Runnable {
    private Data load;

    Receiver(Data load) {
        this.load=load;
    }

    public void run() {
        for(String receivedMessage = load.receive();
            !"End".equals(receivedMessage);
            receivedMessage = load.receive())
        {
            System.out.println(receivedMessage);
            try {
                Thread.sleep(ThreadLocalRandom.current().nextInt(1000, 5000));
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted"+e.getMessage());
            }
        }
    }
}

public static void main(String[] args) {
    Data data = new Data();
    Thread sender = new Thread(new Sender(data));
    Thread receiver = new Thread(new Receiver(data));

    sender.start();
    receiver.start();
}
}

```

#### Output - SyscsJavaProblems (run)

```

run:
First packet
Second packet
Third packet
Fourth packet
BUILD SUCCESSFUL (total time: 17 seconds)

```

## Practical 7: Client Server using Java Sockets

```
import java.io.*;
import java.net.*;

/**
 *
 * @author Manasvi
 */
public class Server {
    public static void main(String args[]) throws Exception{
        ServerSocket ss=new ServerSocket(9999);
        Socket s=ss.accept();
        DataInputStream din=new DataInputStream(s.getInputStream());
        DataOutputStream dout=new DataOutputStream(s.getOutputStream());
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        String str="",str2="";
        while(!str.equals("stop")){
            str=din.readUTF();
            System.out.println("client says: "+str);
            str2=br.readLine();
            dout.writeUTF(str2);
            dout.flush();
        }
        din.close();
        s.close();
        ss.close();
    }
}

import java.net.*;
import java.io.*;
class Client{
    public static void main(String args[]) throws Exception{
        Socket s=new Socket("localhost",9999);
        DataInputStream din=new DataInputStream(s.getInputStream());
        DataOutputStream dout=new DataOutputStream(s.getOutputStream());
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        String str="",str2="";
        while(!str.equals("stop")){
            str=br.readLine();
            dout.writeUTF(str);
            dout.flush();
            str2=din.readUTF();
            System.out.println("Server says: "+str2);
        }
        dout.close();
        s.close();
    }
}
```

Client starts the conversation and then server replies

```
Output
SycsJavaProblems (run) × SycsJavaProblems (run) #2 ×
run:
hi when is java practical ?
```

```
Output
SycsJavaProblems (run) × SycsJavaProblems (run) #2 ×
run:
client says: hi when is java practical ?
4th november
```

```
Output
SycsJavaProblems (run) × SycsJavaProblems (run) #2 ×
run:
hi when is java practical ?
Server says: 4th november
stop
```

```
Output
SycsJavaProblems (run) × SycsJavaProblems (run) #2 ×
run:
client says: hi when is java practical ?
4th november
client says: stop
stop
BUILD SUCCESSFUL (total time: 1 minute 43 seconds)
```

```
Output
SycsJavaProblems (run) × SycsJavaProblems (run) #2 ×
run:
hi when is java practical ?
Server says: 4th november
stop
Server says: stop
BUILD SUCCESSFUL (total time: 1 minute 36 seconds)
```



## Practical 8: Object Oriented Concepts

```
interface Calculator{
    double add(double a, double b);
    double sub(double a, double b);
    double div(double a, double b);
    double mul(double a, double b);
}

abstract class MoreToCalculate{
    abstract double modulo(int a ,int b);
    double percentage(double value,double outof){
        double r=value/outof;
        return (r*100);
    }
}

class Logic extends MoreToCalculate implements Calculator{
    double modulo(int a, int b) {
        return a%b;
    }
    public double add(double a, double b) {
        return a+b;
    }
    public double sub(double a, double b) {
        return a-b;
    }
    public double div(double a, double b) {
        return a/b;
    }
    public double mul(double a, double b) {
        return a*b;
    }
}
```

```

public class ObjectOriented extends Logic {

    static int a,b,c;

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

    public double add(int a,int b,int c){
        return a+b+c;
    }

    public double modulo(int a, int b){
        return b%a;
    }

    double callSuperModulo(int a,int b){
        return super.modulo(a, b);
    }

    public static void main(String arg[]){
        ObjectOriented obj=new ObjectOriented();
        new ObjectOriented(450,700,850);
        System.out.println("Add two numbers"+a+" "+b+" = "+obj.add(a, b));
        System.out.println("Add three numbers"+a+" "+b+" "+c+" = "+obj.add(a, b));
        System.out.println("Subtract two numbers"+a+" "+b+" = "+obj.sub(a, b));
        System.out.println("Multiply two numbers"+a+" "+b+" = "+obj.mul(a, b));
        System.out.println("Divide two numbers"+a+" "+b+" = "+obj.div(a, b));
        System.out.println("Modulo two numbers"+b+" "+a+" = "+obj.modulo(a, b));
        System.out.println("Modulo two numbers"+a+" "+b+" = "+obj.callSuperModulo(a, b));
        System.out.println("Percentage of two numbers"+a+" "+b+" = "+obj.percentage(a, b));
    }
}

```

#### Output - SycsJavaProblems (run)

```

run:
Add two numbers450 700 = 1150.0
Add three numbers450 700 850 = 1150.0
Subtract two numbers450 700 = -250.0
Multiply two numbers450 700 = 315000.0
Divide two numbers450 700 = 0.6428571428571429
Modulo two numbers700 450 = 250.0
Modulo two numbers450 700 = 450.0
Percentage of two numbers450 700 = 64.28571428571429
BUILD SUCCESSFUL (total time: 0 seconds)

```

## Practical 9: Inner class example

```
public class InnerClass {  
    /*  
     * Creating Inner class in Java  
     */  
    private class Inner{  
        public void name(){  
            System.out.println("Inner class example in java");  
        }  
    }  
    public static void main(String args[]) {  
        //creating local inner class inside method  
        class Local {  
            public void name() {  
                System.out.println("Example of Local class in Java");  
            }  
        }  
        //creating instance of local inner class  
        Local local = new Local();  
        local.name(); //calling method from local inner class  
        //Creating anonymous inner class in java for implementing thread  
        Thread anonymous = new Thread(){  
            @Override  
            public void run(){  
                System.out.println("Anonymous class example in java");  
            }  
        };  
        anonymous.start();  
        //example of creating instance of inner class  
        InnerClass test = new InnerClass();  
        InnerClass.Inner inner = test.new Inner();  
        inner.name(); //calling method of inner class  
    }  
}
```

### Output - SyscsJavaProblems (run)

```
run:  
Example of Local class in Java  
Anonymous class example in java  
Inner class example in java  
BUILD SUCCESSFUL (total time: 3 seconds)
```

## Practical 10: Collection framework example.

```
import java.util.*;

/**
 *
 * @author Manasvi
 */
public class Collection {
    public static void main(String args[]) {
        ArrayList<String> alist=new ArrayList<String>();
        alist.add("Apple");
        alist.add("Mango");
        alist.add("Pineapple");
        alist.add("Grape");
        alist.add("Banana");
        alist.add("Strawberry");
        System.out.println("Arraylist using iterator ");
        Iterator<String> itr=alist.iterator();
        while(itr.hasNext()){
            System.out.println(itr.next());
        }
        //displaying elements
        System.out.println(alist);
        //Adding "Apple" at the fourth position
        alist.add(3, "Apple");
        //displaying elements
        System.out.println(alist);
        //Removing "Apple" and "Grape"
        alist.remove("Apple");
        alist.remove("Grape");
        //displaying elements
        System.out.println(alist);
        //Removing 3rd element
        alist.remove(2);
        //displaying elements
        System.out.println(alist);
        //iterating ArrayList
        for(String str:alist)
            System.out.println(str);
    }
}
```

Output - SycsJavaProblems (run)



```
run:
Arraylist using iterator
Apple
Mango
Pineapple
Grape
Banana
Strawberry
[Apple, Mango, Pineapple, Grape, Banana, Strawberry]
[Apple, Mango, Pineapple, Apple, Grape, Banana, Strawberry]
[Mango, Pineapple, Apple, Banana, Strawberry]
[Mango, Pineapple, Banana, Strawberry]
Mango
Pineapple
Banana
Strawberry
BUILD SUCCESSFUL (total time: 2 seconds)
.
```