**ICS1312 – JAVA PROGRAMMING LABORATORY**

**DATE : 24.7.2025**

**ASSIGNMENT : 1C**

**TITLE : ARRAYS AND STRINGS**

**ROLLL NO : 3122247001017**

**LEARNING OBJECTIVE:**

* **To implement string manipulation tasks in java**
* **To work with arrays in java**
* **To perform matrix multiplication,pangram,binary search in java**



CLASS DIAGRAM :

A close-up of a list of data

AI-generated content may be incorrect.

A close-up of a list of data

AI-generated content may be incorrect.

CODE:

import java.util.Scanner;

class **data**{

    int value;

    int index;

**data**(int value,int index)

    {

            this.index=index;

            this.value=value;

    }

}

class **Search1**

{

public int[] elements;

int n;

Scanner sc = new **Scanner**(System.in);

public void **getelements**()

{

System.out.**println**("Enter number of elements : ");

n=sc.**nextInt**();

elements = new int[n];

for(int i=0;i<n;i++)

{

System.out.**println**("Enter element :" + (i+1));

elements[i]=sc.**nextInt**();

}

}

public int **linear**()

{

int s;

boolean status =false;

System.out.**println**(" Enter an element to search ");

s=sc.**nextInt**();

for(int i=0;i<n;i++)

{

if(elements[i]==s)

{

System.out.**println**("The element found in the index :"+ i);

status = true;

return(i);

}

}

if(!status)

System.out.**println**("No such element found");

return(-1);

}

public data[] **mergesort**(data[] array)

{

if(array.length <= 1)

{

    return(array);

}

int n=array.length;

int mid=n/2;

data[] L=new data[mid];

data[] R=new data[n-mid];

for(int i=0;i<mid;i++)

{

    L[i]=array[i];

}

for(int j=0;j<(n-mid);j++)

{

    R[j]=array[mid+j];

}

data[] left=**mergesort**(L);

data[] right=**mergesort**(R);

return(**merge**(left,right));

}

public data[] **merge**(data[] L,data[] R)

{

int i=0;

int j=0;

int k=0;

int n=L.length+R.length;

data[] result=new data[n];

while(i<L.length && j<R.length)

{

if(L[i].value < R[j].value)

{

  result[k++]=L[i++];

}

else

{

  result[k++]=R[j++];

}

}

while(i<L.length)

{

    result[k++]=L[i++];

}

while(j<R.length)

{

    result[k++]=R[j++];

}

return(result);

}

int count=0;

data[] arr;

data[] sortedelements;

public int **binarysearch**()

{

if(count++ == 0)

{

arr=new data[n];

sortedelements=new data[n];

for(int i=0;i<n;i++)

{

arr[i]=new **data**(elements[i],i);

}

sortedelements=**mergesort**(arr);

}

int s;

boolean status=false;

System.out.**println**("Enter an element to search");

s=sc.**nextInt**();

int low =0;

int high =elements.length-1;

int mid;

while(low <= high)

{

mid = low + (high - low) / 2;

if(sortedelements[mid].value == s)

{

System.out.**println**("The element found in the index:" + sortedelements[mid].index);

return(mid);

}

else if(s < sortedelements[mid].value)

{

high=mid-1;

}

else if(s > sortedelements[mid].value)

{

low = mid +1;

}

}

System.out.**println**("No such element found");

return(-1);

}

}

class **Search**

{

public static void **main**(String[] args)

{

int index;

Scanner sc = new **Scanner**(System.in);

Search1 s=new **Search1**();

s.**getelements**();

while(true)

{

System.out.**println**("------------------- MENU ------------------");

System.out.**println**("1. PRESS 1 : LINEAR SEARCH");

System.out.**println**("2. PRESS 2 : BINARY SEARCH");

System.out.**println**("3. PRESS 3 : EXIT         ");

System.out.**println**("-------------------------------------------");

int res =sc.**nextInt**();

if(res == 1)

index=s.**linear**();

else if(res == 2)

index=s.**binarysearch**();

else if(res == 3)

break;

else

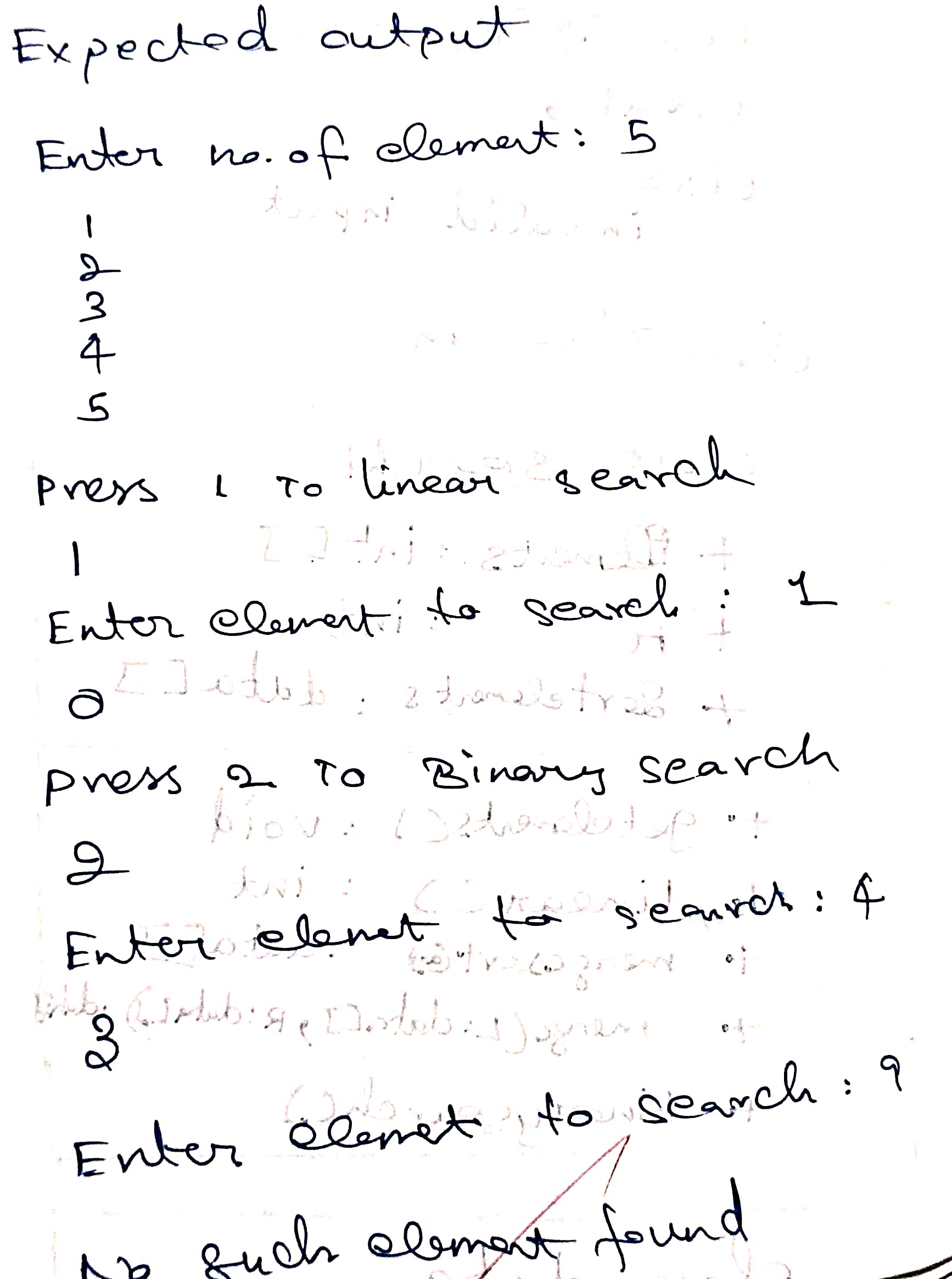
System.out.**println**("INVALID INPUT");

}

}

}

**OUTPUT:**



**CASE 1:**

A screenshot of a computer code

AI-generated content may be incorrect.

ALL POSSIBLE CASSES:

A screenshot of a computer

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

CLASS DIAGRAM:

A close-up of a paper

AI-generated content may be incorrect.

A close-up of a diagram

AI-generated content may be incorrect.

CODE:

import java.util.Arrays;

import java.util.Scanner;

class Matrixcreate

{

int rows;

int cols;

Scanner sc = new Scanner(System.in);

int[][] mat;

public void create()

{

System.out.println("Enter number of rows :");

rows=sc.nextInt();

System.out.println("Enter number of columns :");

cols=sc.nextInt();

mat=new int[rows][cols];

for(int i=0;i<rows;i++)

{

for(int j=0;j<cols;j++)

{

    System.out.println("Enter Element row[" + i + "] column[" + j + "] :");

    mat[i][j]=sc.nextInt();

}}

System.out.println(Arrays.deepToString(mat));

}

public int[][] getmatrix(){

    return mat;

}}

class Matrixoperation

{

public int[][] Addition(int[][] a,int[][] b)

{

int row1len =a.length;

int column1len=a[0].length;

int row2len=b.length;

int column2len=b[0].length;

if(row1len != row2len || column1len != column2len)

{

System.out.println("The order of the matrix is different : Addition not possible");

return(null);

}

int[][] c =new int[row1len][column1len];

for(int i=0;i<row1len;i++)

{

    for(int j=0;j<column1len;j++)

    {

        c[i][j] = a[i][j] + b[i][j];

    }

}

return(c);}

public int[][] subtraction(int[][] a,int[][] b)

{

int row1len =a.length;

int column1len=a[0].length;

int row2len=b.length;

int column2len=b[0].length;

if(row1len != row2len || column1len != column2len)

{

System.out.println("The order of the matrix is different : Subtraction not possible");

return(null);

}

int[][] c =new int[row1len][column1len];

for(int i=0;i<row1len;i++)

{

    for(int j=0;j<column1len;j++)

    {

        c[i][j] = a[i][j] - b[i][j];

    }

}

return(c);

}

public int[][] multiplication(int[][] a,int[][] b)

{

int row1len =a.length;

int column1len=a[0].length;

int row2len=b.length;

int column2len=b[0].length;

if(column1len != row2len )

{

System.out.println("Multiplication not possible for This  order of the matrix ");

return(null);

}

int[][] c =new int[row1len][column2len];

for(int i=0;i<row1len;i++)

{

    for(int j=0;j<column2len;j++)

    {

        for(int k=0;k<row2len;k++)

        {

        c[i][j] += a[i][k]  \* b[k][j];

        }

    }

}

return(c);

}

}

class Matrix{

    public static void main(String[] args)

    {

       Matrixcreate A=new Matrixcreate();

       Matrixcreate B=new Matrixcreate();

       System.out.println("Matrix A:");

       A.create();

       System.out.println("Matrix B:");

       B.create();

       Matrixoperation obj=new Matrixoperation();

       int[][] result;

       result=obj.Addition(A.getmatrix(),B.getmatrix());

       System.out.println("Addition result :  " + Arrays.deepToString(result));

       result=obj.subtraction(A.getmatrix(),B.getmatrix());

       System.out.println("Subtraction result  : " + Arrays.deepToString(result));

       result=obj.multiplication(A.getmatrix(),B.getmatrix());

       System.out.println("Multiplication result  : " + Arrays.deepToString(result));

    }

}

**OUTPUT:**

A close-up of a paper

AI-generated content may be incorrect.

A close-up of a paper

AI-generated content may be incorrect.

CASE 1:

A screenshot of a computer program

AI-generated content may be incorrect.

CASE 2:

A screenshot of a computer program

AI-generated content may be incorrect.

CASE 3:

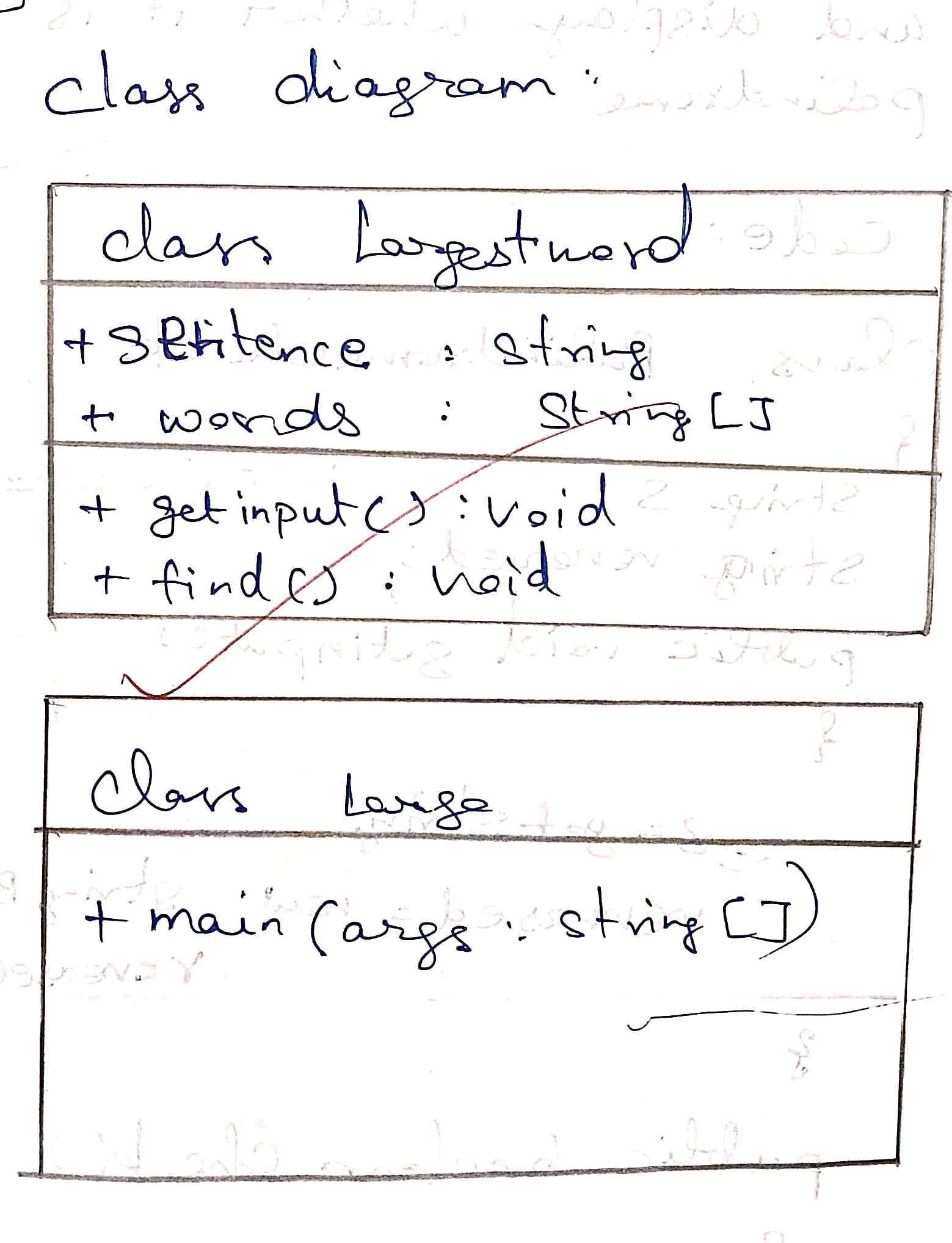
A screenshot of a computer program

AI-generated content may be incorrect.

A black text on a white background

AI-generated content may be incorrect.

CLASS DIAGRAM :



CODE:

import java.util.Scanner;

class Largestword

{

String sentence;

String[] words;

public void getinput()

{

System.out.println(" Enter an Sentence : ");

Scanner sc=new Scanner(System.in);

sentence=sc.nextLine();

words=sentence.split(" ");

}

public void find()

{

int max=words[0].length();

int index=0;

int lenplus=0;

String large=words[0];

for(String i:words)

{

    System.out.println(i);

    if(i.length() > max)

    {

        max=i.length();

        large=i;

        index=lenplus;

    }

    lenplus+=i.length()+1;

}

System.out.println("The largest word is :" + large + "," + index + " length :" +max);

}

}

class Large{

    public static void main(String[] args){

            Largestword a=new Largestword();

            a.getinput();

            a.find();

}

}

OUTPUT:

Close-up of a handwritten letter

AI-generated content may be incorrect.

CASE 1

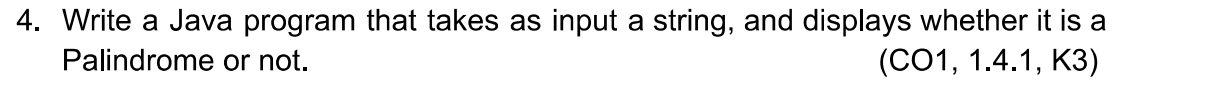
A screen shot of a computer program

AI-generated content may be incorrect.

CASE 2

A screen shot of a computer

AI-generated content may be incorrect.



CLASS DIAGRAM:

A close-up of a list of notes

AI-generated content may be incorrect.

CODE:

import java.util.Scanner;

class Palindromecheck

{

    String s;

    String reversed;

    public void getinput()

    {

    System.out.println("Enter a string:");

    Scanner sc = new Scanner(System.in);

    s=sc.nextLine();

    reversed=new StringBuilder(s).reverse().toString();

    }

    public boolean check()

    {

    return(reversed.equalsIgnoreCase(s));

    }

}

class Palindrome{

    public static void main(String[] args){

        Palindromecheck a=new Palindromecheck();

        a.getinput();

        boolean result=a.check();

        if(result)

        {

            System.out.println("It is a palindrome");

        }

        else

        {

            System.out.println("It is not a palindrome");

        }

    }

}

Close-up of a note

AI-generated content may be incorrect.

OUTPUT:

CASE 1:

A screenshot of a computer program

AI-generated content may be incorrect.

CASE 2:

A screenshot of a computer program

AI-generated content may be incorrect.

A white background with black text

AI-generated content may be incorrect.

CLASS DIAGRAM:

A close-up of a list of tasks

AI-generated content may be incorrect.

CODE:

import java.util.Scanner;

class Palindromecheck

{

    String s;

    String reversed;

    public void getinput()

    {

    System.out.println("Enter a string:");

    Scanner sc = new Scanner(System.in);

    s=sc.nextLine();

    reversed=new StringBuilder(s).reverse().toString();

    }

    public boolean check()

    {

    return(reversed.equalsIgnoreCase(s));

    }

}

class Palindrome{

    public static void main(String[] args){

        Palindromecheck a=new Palindromecheck();

        a.getinput();

        boolean result=a.check();

        if(result)

        {

            System.out.println("It is a palindrome");

        }

        else

        {

            System.out.println("It is not a palindrome");

        }

    }

}

TEST CASES:

A test paper with a red line

AI-generated content may be incorrect.

A close-up of a white board

AI-generated content may be incorrect.

OUTPUT:

CASE 1:

A screenshot of a computer

AI-generated content may be incorrect.

CASE 2:

A close up of words

AI-generated content may be incorrect.

**LEARNING OUTCOMES:**

* **Learned to implement string manipulation tasks in java**
* **Learned to work with arrays in java**
* **Learned to define logic for merge sort in java**