DATA STRUCTURE AND ALGORITHM WITH JAVA ASSIGNMENT

Lecturer: Jean Bosco Willy NDUSHABANDI

PREPARED BY: Taufique IRADUKUNDA

RegNo: 217025080

Contents

[Questions 2](#_Toc31565772)

[1. Write an algorithm to find the largest of a set of numbers 2](#_Toc31565773)

[2. Write an algorithm in pseudocode that finds the average of (n) numbers. 2](#_Toc31565774)

[3. Implement and test the linear search method by creating an array of 10 elements 2](#_Toc31565775)

[4. Implement and test the binary search method. The program first creates an array of 10 elements 2](#_Toc31565776)

[5. Write a program to Add two Matrix 2](#_Toc31565777)

[6. Write a program to Multiply two Matrix 2](#_Toc31565778)

[7. Write a that Implement stack 2](#_Toc31565779)

[8. Write a program that Implement Queue 2](#_Toc31565780)

[Answer 3](#_Toc31565781)

[1. algorithm to find the largest of a set of numbers 3](#_Toc31565782)

[2. Algorithm in pseudocode that finds the average of (n) numbers 3](#_Toc31565783)

[3. LINEAR SEARCH PROGRAM 4](#_Toc31565784)

[4. BINARY SEARCH PROGRAM 5](#_Toc31565785)

[5. PROGRAM TO MULTIPLY TWO MATRIXES 7](#_Toc31565786)

[6. ADD TWO MATRIXES 8](#_Toc31565787)

[7. STACK PROGRAM 9](#_Toc31565788)

[8. QAUEUE PROGRAM 10](#_Toc31565789)

# Questions

1. Write an algorithm to find the largest of a set of numbers. You do not know the number of numbers.

## Write an algorithm in pseudocode that finds the average of (n) numbers.

For example, numbers are [4,5,14,20,3,6]

1. Implement and test the linear search method by creating an array of 10 elements of int type randomly and then display this array. Prompt the user to enter a key for testing the linear search.
2. Implement and test the binary search method. The program first creates an array of 10 elements of int type. It displays this array and then prompts the user to enter a key for testing binary search.

## Write a program to Add two Matrix

## Write a program to Multiply two Matrix

## Write a that Implement stack

## Write a program that Implement Queue

# Answer

1. algorithm to find the largest of a set of numbers

Step 1: Start

Step 2: Declare variables **a**, b and c.

Step 3: Read variable a, b, and c.

Step 4: If a > b and a > c

Display a is the largest number.

Else If b > a and b > c

Display b is the largest number

Else

Display c is the largest number

Step 5: Stop

1. Algorithm in pseudocode that finds the average of (n) numbers.

input: integer Array list [] of N positive integers

Output: Average of numbers

Process:

Double sum ←0

Double average ←0

Count ←0

While (count <size of list)

Sum ← sum + list[count]

Count ← count + 1

End while

Average ←sum /n

Return Average

## LINEAR SEARCH PROGRAM

**package** datastructure;

**import** java.util.Scanner;

**public** **class** linearSearch{

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

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

**int** a[] = {1,2,3,4,5,6,7,8,9,10};

**for**(**int** j=0;j<a.length;j++)//length is the property of array

System.***out***.println(+a[j]);

System.***out***.println("Enter Element from the array to find its location");

**int** key = myObj.nextInt();

**for**(**int** i=0;i<a.length;i++) {

**if**(a[i] == key){

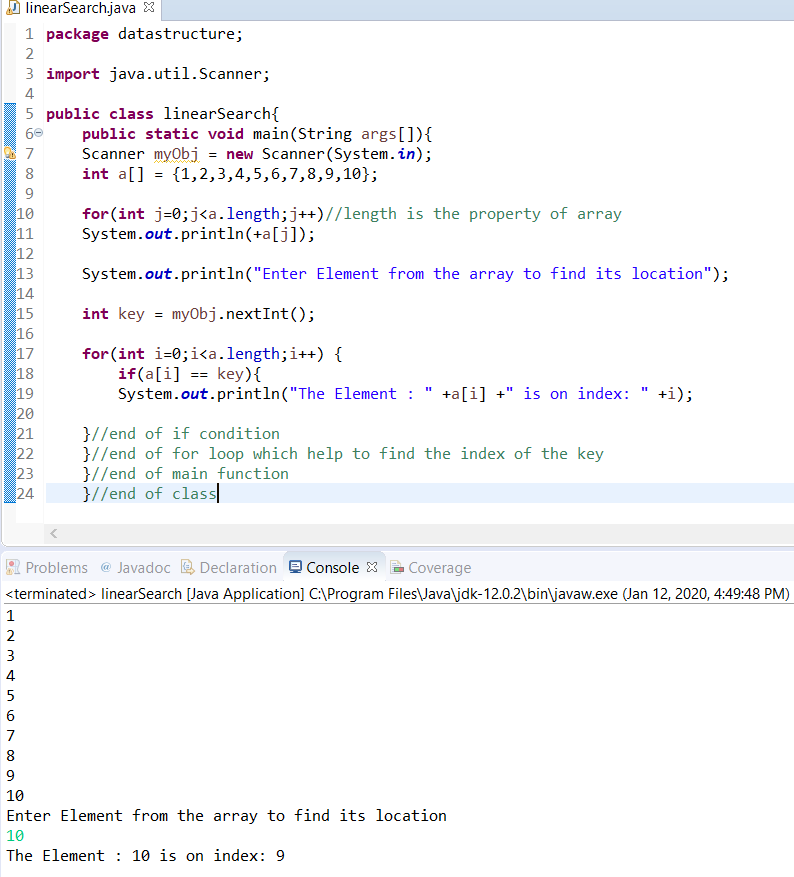
System.***out***.println("The Element: " +a[i] +" is on index: " +i);

}//end of if condition

}//end of for loop which help to find the index of the key

}//end of main function

}//end of class



***FIGURE 1: Linear search program and output***

## BINARY SEARCH PROGRAM

**package** datastructure;

**import** java.util.\*;

**public** **class** BinarySearch {

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

**int**[] arr = {16, 19, 20, 23, 45, 56, 78, 90, 96, 100};

**int** item, location = -1;

**for**(**int** j=0;j<arr.length;j++)//length is the property of array

System.***out***.println(arr[j]);

System.***out***.println("Enter the item from the list to find its location");

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

item = sc.nextInt();

location = *binarySearch*(arr,0,9,item);

**if**(location != -1)

System.***out***.println("the item is in the Location "+location);

**else**

System.***out***.println("Item not found");

}

**public** **static** **int** binarySearch(**int**[] a, **int** beg, **int** end, **int** item)

{

**int** mid;

**if**(end >= beg)

{

mid = (beg + end)/2;

**if**(a[mid] == item)

{

**return** mid+1;

}

**else** **if**(a[mid] < item)

{

**return** *binarySearch*(a,mid+1,end,item);

}

**else**

{

**return** *binarySearch*(a,beg,mid-1,item);

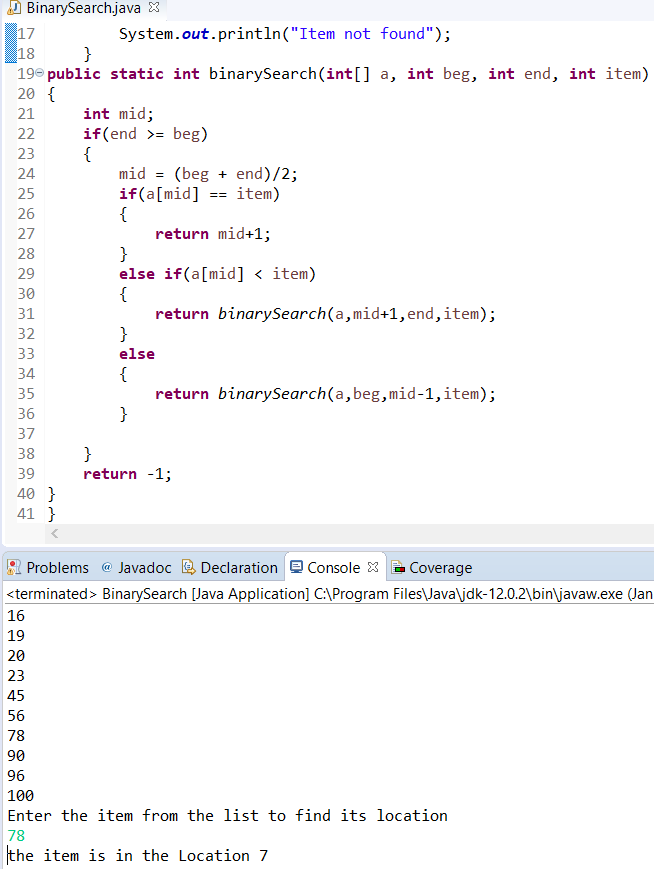
}

}

**return** -1;

}

}



***FIGURE2: Binary Search program and output***

## PROGRAM TO MULTIPLY TWO MATRIXES

**package** datastructure;

**public** **class** multipy2matrix{

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

//creating two matrices

**int** a[][]={{1,1,1},{2,2,2},{3,3,3}};

**int** b[][]={{1,1,1},{2,2,2},{3,3,3}};

//creating another matrix to store the multiplication of two matrices

**int** c[][]=**new** **int**[3][3]; //3 rows and 3 columns

System.***out***.println("The Answer is: ");

//multiplying and printing multiplication of 2 matrices

**for**(**int** i=0;i<3;i++){

**for**(**int** j=0;j<3;j++){

c[i][j]=0;

**for**(**int** k=0;k<3;k++)

{

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

}//end of k loop

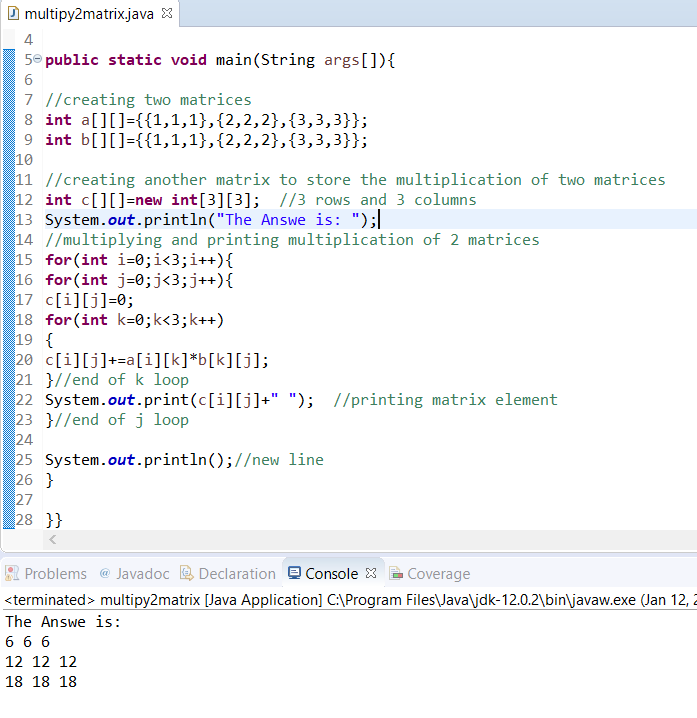
System.***out***.print(c[i][j]+" "); //printing matrix element

}//end of j loop

System.***out***.println();//new line

}

}}



***FIGURE 3: program to Multiply two matrix.***

## ADD TWO MATRIXES

**package** datastructure;

**public** **class** AddTwoMatrix {

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

//creating two matrices

**int** a[][]={{1,3,4},{2,4,3},{3,4,5}};

**int** b[][]={{1,3,4},{2,4,3},{1,2,4}};

//creating another matrix to store the sum of two matrices

**int** c[][]=**new** **int**[3][3]; //3 rows and 3 columns

System.***out***.println("Addition of Two matrix is: ");

//for loops used to loop in the matrix

**for**(**int** i=0;i<3;i++){

**for**(**int** j=0;j<3;j++){

c[i][j]=a[i][j]+b[i][j];

System.***out***.print(c[i][j]+" ");

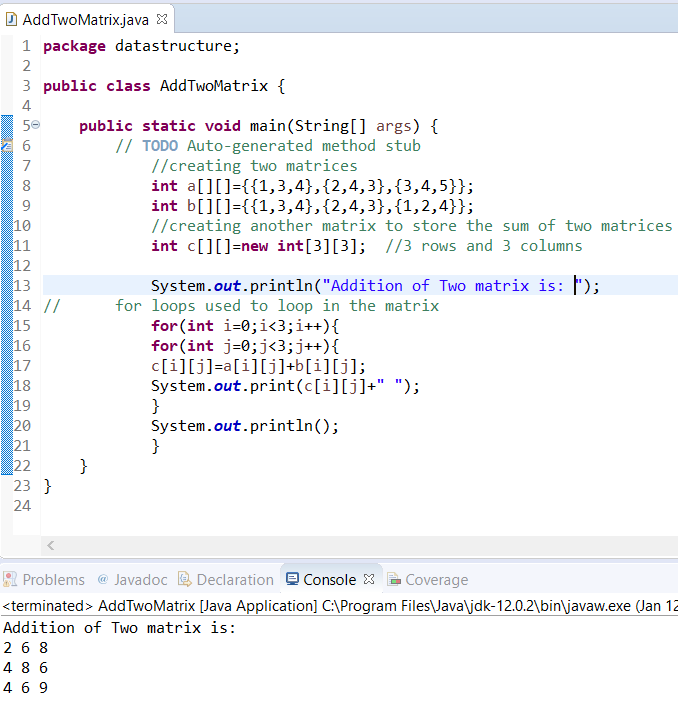
}

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

}

}

}



***FIGURE 4: program to Add Two Matrix program and output***

## STACK PROGRAM

**package** datastructure;

**import** java.util.\*;

**public** **class** StackPro {

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

{

// Creating an empty Stack

Stack<String> STACK = **new** Stack<String>();

// Use push() method to add elements

STACK.push("THIS");

STACK.push("IS");

STACK.push("STACK");

STACK.push("PROGRAM");

// Displaying the Stack

System.***out***.println("Initial Stack: " + STACK);

System.***out***.println("Stack After POP: ");

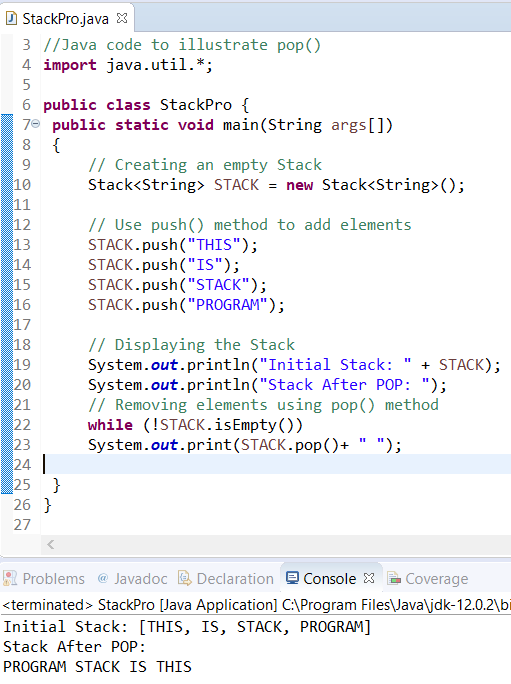
// Removing elements using pop() method

**while** (!STACK.isEmpty())

System.***out***.print(STACK.pop()+ " ");

}

}



***FIGURE 5: Stack program and output***

## QAUEUE PROGRAM

package datastructure;

//Java program to demonstrate working of Queue

//interface in Java

import java.util.LinkedList;

import java.util.Queue;

public class QueuePro

{

public static void main(String[] args)

{

Queue<Integer> q = new LinkedList<>();

// Adds elements {0, 1, 2, 3, 4} to queue

for (int i=0; i<5; i++)

q.add(i);

// Display contents of the queue.

System.out.println("Elements of queue: "+q);

System.out.println("dequeue element from queue: ");

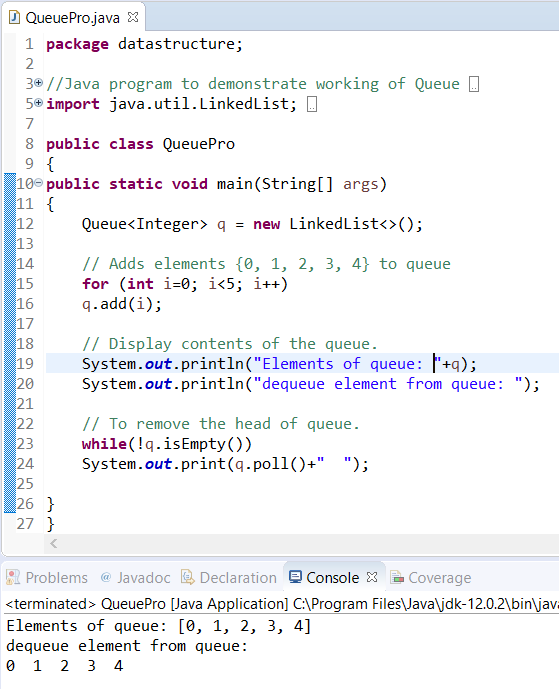
// To remove the head of queue.

while(!q.isEmpty())

System.out.print(q.poll()+" ");

}

}



***FIGURE 6: queue program and output***