EC4070: Data Structures and Algorithms

LAB 02

FINAL

K.J.M.U.G.S. Eranda Jayasinghe

2021/E/075

SEMESTER 4

EC4070

06.10.2023

Q1.

import java.util.Scanner;

public class Recursive{

public static void main(String[] args) {

Scanner x = new Scanner(System.in);

System.out.print("Enter a number to check triangular number: ");

int num1 = x.nextInt();

int trinum = checktriangulaer(num1);

System.out.println("Triangular number of " + num1 + " is " + trinum);

System.out.print("\nEnter a number to check factorial: ");

int num2 = x.nextInt();

int facnum = checkfactorial(num2);

System.out.println("Factorial of " + num2 + " is " + facnum);

x.nextLine();

System.out.print("\nEnter first string: ");

String str1 = x.nextLine();

System.out.print("Enter second string: ");

String str2 = x.nextLine();

boolean anagram = anagram(str1, str2);

if (anagram) {

System.out.println(str1 + " and " + str2 + " are anagrams.");

} else {

System.out.println(str1 + " and " + str2 + " are not anagrams.");

}

System.out.print("\nEnter the number for check Towers of Hanoi: ");

int num3 = x.nextInt();

hanoi(num3, 'A', 'C', 'B');

x.close();

}

public static int checktriangulaer(int n) {

if (n <= 0) {

return 0;

} else {

return n + checktriangulaer(n - 1);

}

}

public static int checkfactorial(int n) {

if (n == 0 || n == 1) {

return 1;

} else {

return n \* checkfactorial(n - 1);

}

}

public static boolean anagram(String str1, String str2) {

str1 = str1.toLowerCase();

str2 = str2.toLowerCase();

if (str1.length() != str2.length()) {

return false;

}

if (str1.length() == 0) {

return true;

}

char firstChar = str1.charAt(0);

int index = str2.indexOf(firstChar);

if (index == -1) {

return false;

} else {

String newStr1 = str1.substring(1);

String newStr2 = str2.substring(0, index) + str2.substring(index + 1);

return anagram(newStr1, newStr2);

}

}

public static void hanoi(int num3, char c1, char c2, char c3) {

if (num3 == 1) {

System.out.println("Move disk 1 from " + c1 + " to " + c2);

return;

}

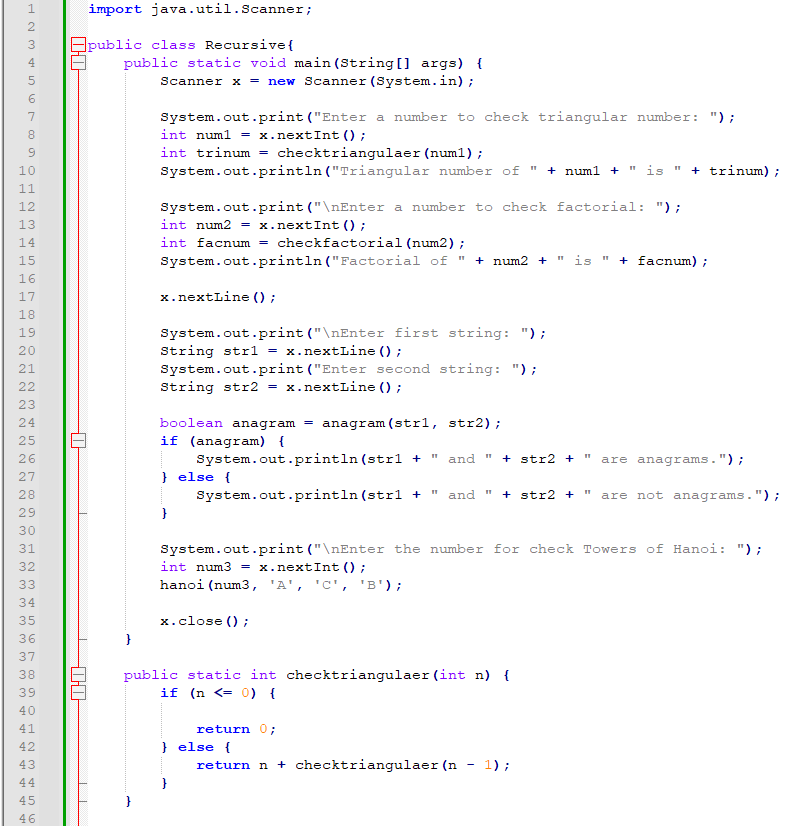
hanoi(num3 - 1, c1, c3, c2);

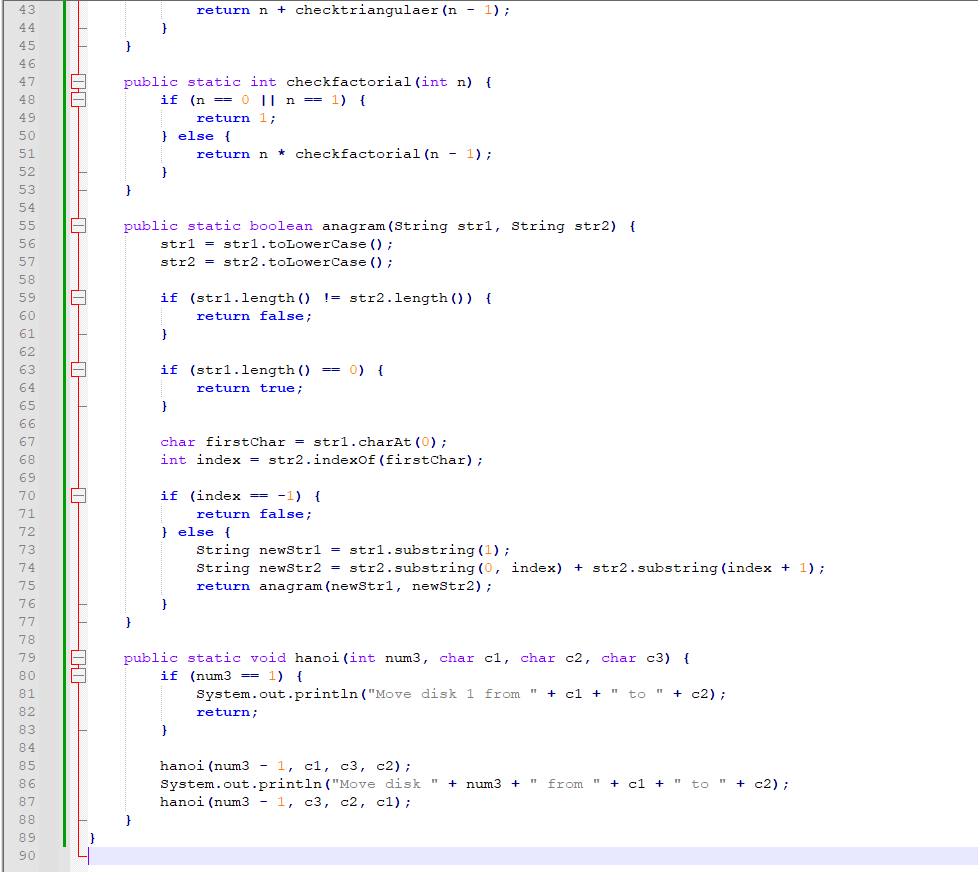
System.out.println("Move disk " + num3 + " from " + c1 + " to " + c2);

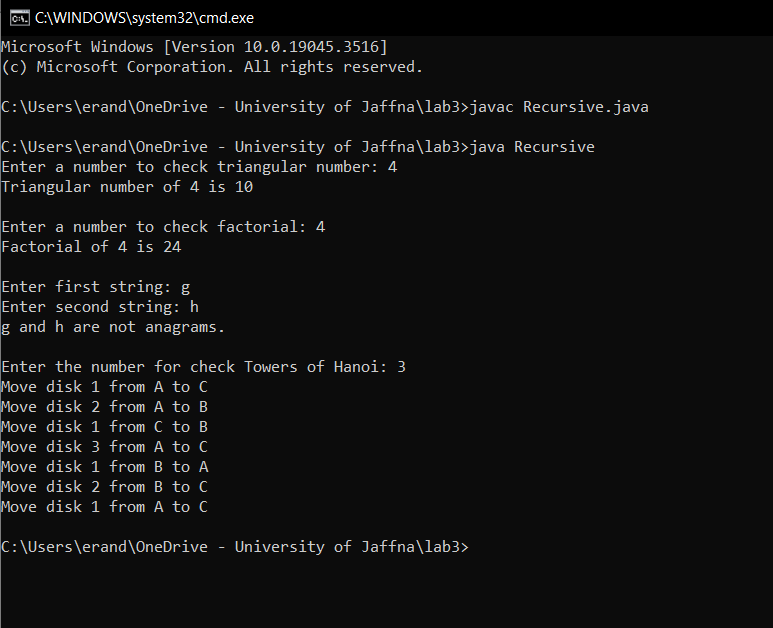
hanoi(num3 - 1, c3, c2, c1);

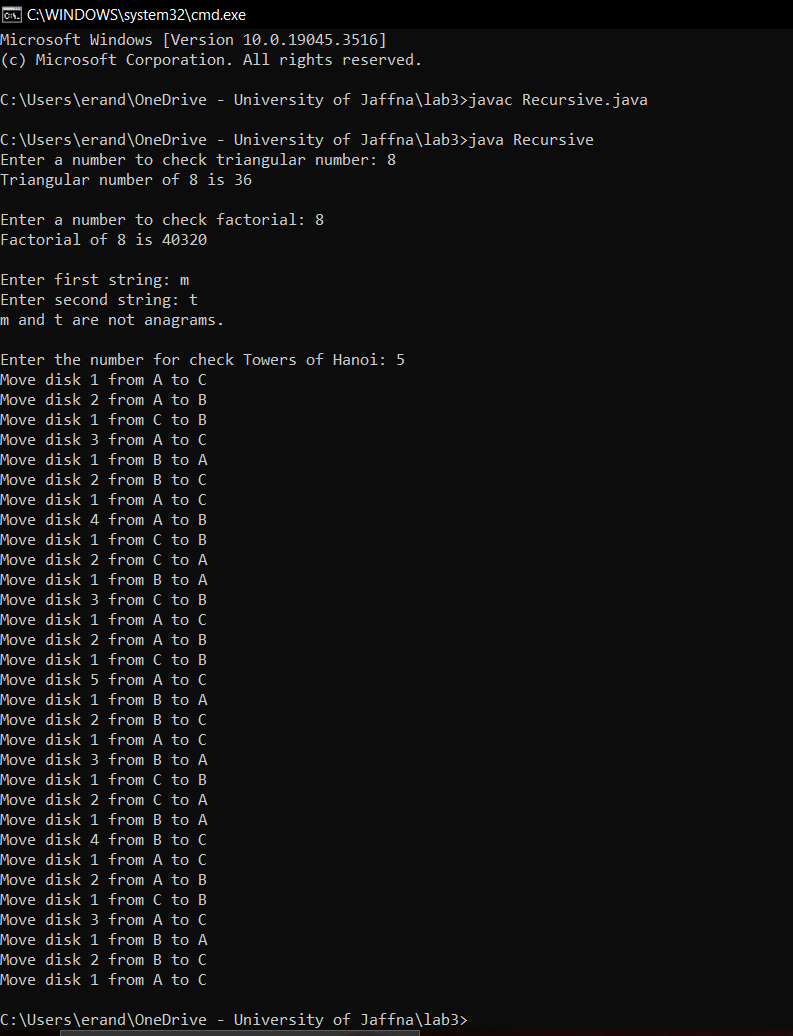
}

}









Q2.a

import java.util.Scanner;

public class binaryS {

public static int search(int[] arr, int t) {

int left = 0;

int right = arr.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == t) {

return mid;

} else if (arr[mid] < t) {

left = mid + 1;

} else {

right = mid - 1;

}

}

return -1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("how many elements in your array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements in sorted order:");

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

arr[i] = sc.nextInt();

}

System.out.print("Enter the target value: ");

int t = sc.nextInt();

int sum = search(arr, t);

if (sum != -1) {

System.out.println("Element " + t + " found at index " + sum);

} else {

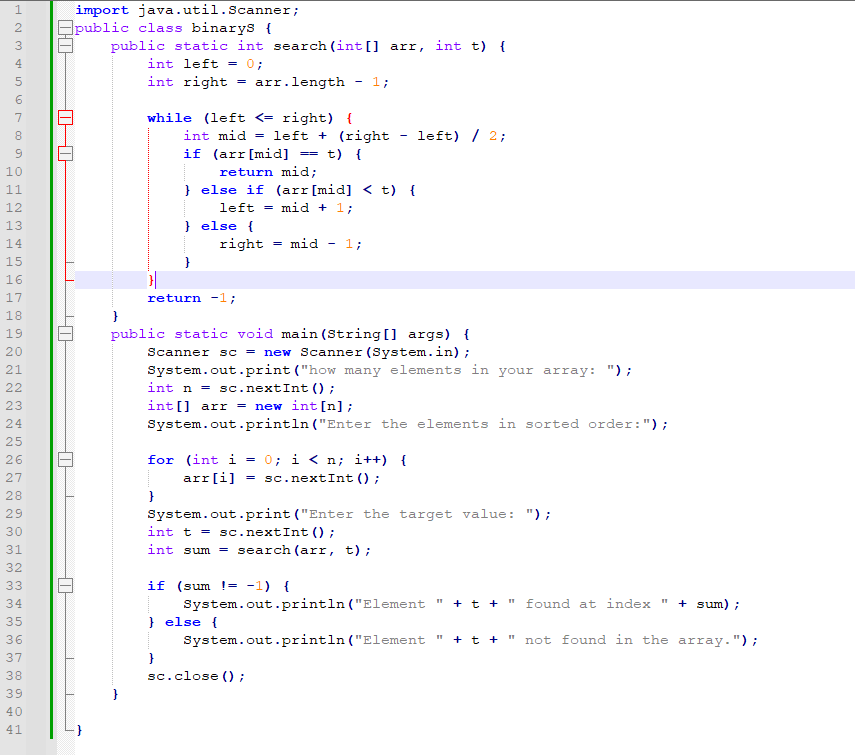
System.out.println("Element " + t + " not found in the array.");

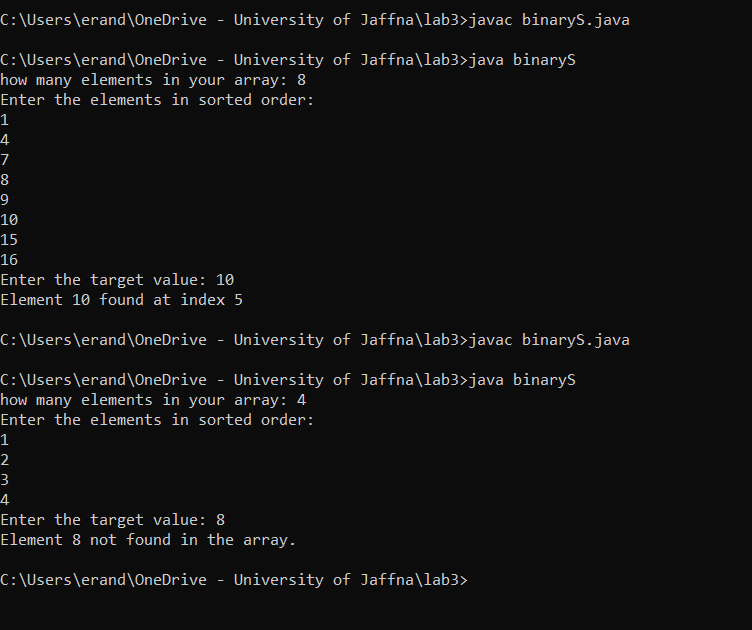
}

sc.close();

}

}





b.

import java.util.Scanner;

public class binaryS1 {

public static int search1(int[] arr, int t, int a, int b) {

if (a <= b) {

int mid = a + (b - a) / 2;

if (arr[mid] == t) {

return mid;

} else if (arr[mid] < t) {

return search1(arr, t, mid + 1, b);

} else {

return search1(arr, t, a, mid - 1);

}

}

return -1;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("How many elements in your array: ");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the sorted array:");

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

arr[i] = sc.nextInt();

}

System.out.print("Enter the target value to search for: ");

int t = sc.nextInt();

int sum = search1(arr, t, 0, arr.length - 1);

if (sum != -1) {

System.out.println(t + " is found at index " + sum);

} else {

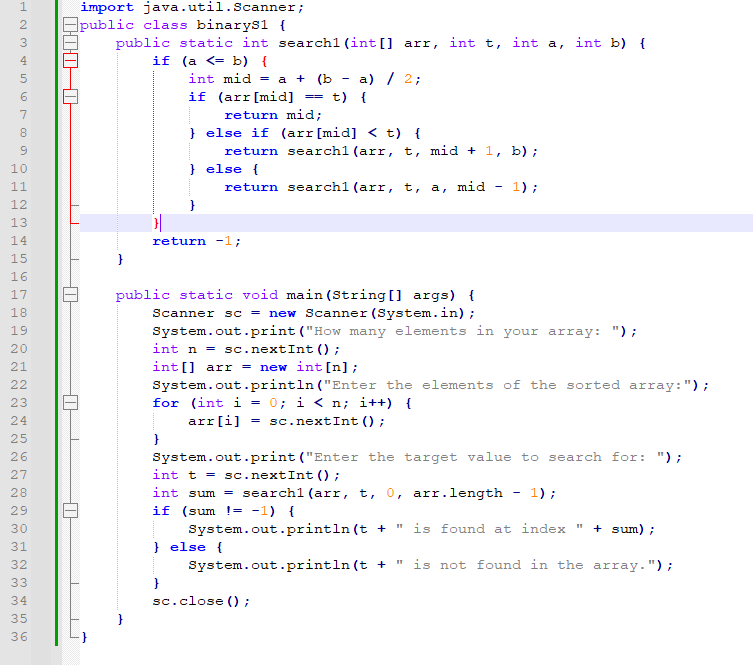
System.out.println(t + " is not found in the array.");

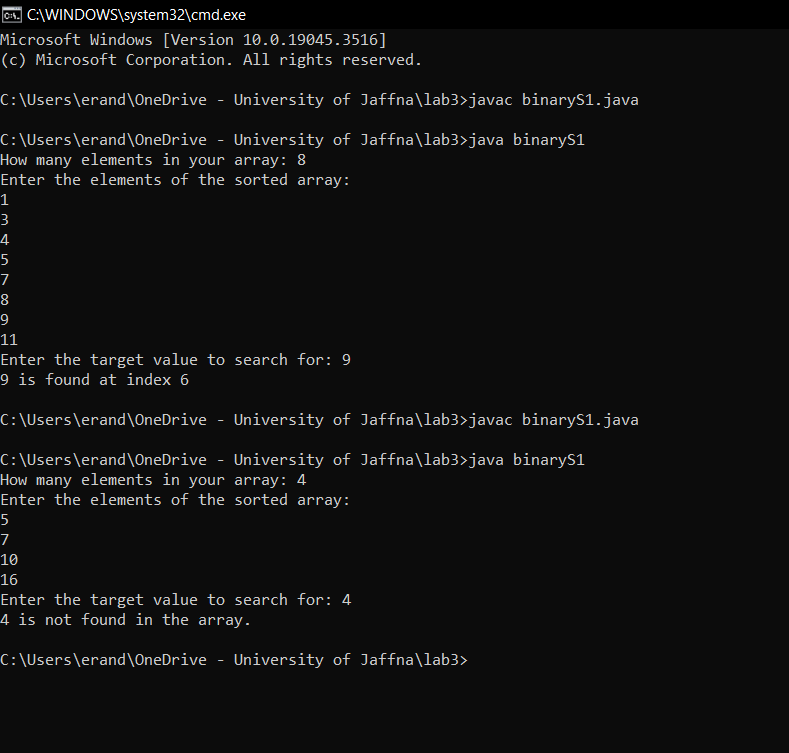
}

sc.close();

}

}





Q3.

import java.util.Scanner;

public class mergesort {

public static void merge(int[] arr, int left, int right) {

if (left < right) {

int mid = left + (right - left) / 2;

merge(arr, left, mid);

merge(arr, mid + 1, right);

merge(arr, left, mid, right);

}

}

private static void merge(int[] arr, int left, int mid, int right) {

int n1 = mid - left + 1;

int n2 = right - mid;

int[] leftarr = new int[n1];

int[] rightarr = new int[n2];

for (int i = 0; i < n1; i++) {

leftarr[i] = arr[left + i];

}

for (int j = 0; j < n2; j++) {

rightarr[j] = arr[mid + 1 + j];

}

int i = 0, j = 0, k = left;

while (i < n1 && j < n2) {

if (leftarr[i] <= rightarr[j]) {

arr[k] = leftarr[i];

i++;

}

else {

arr[k] = rightarr[j];

j++;

}

k++;

}

while (i < n1) {

arr[k] = leftarr[i];

i++;

k++;

}

while (j < n2) {

arr[k] = rightarr[j];

j++;

k++;

}

}

public static void main(String[] args) {

Scanner x = new Scanner(System.in);

System.out.print("Enter the number of elements in the array: ");

int n = x.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array:");

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

arr[i] = x.nextInt();

}

merge(arr, 0, n - 1);

printArray(arr);

x.close();

}

public static void printArray(int[] arr) {

for (int num : arr) {

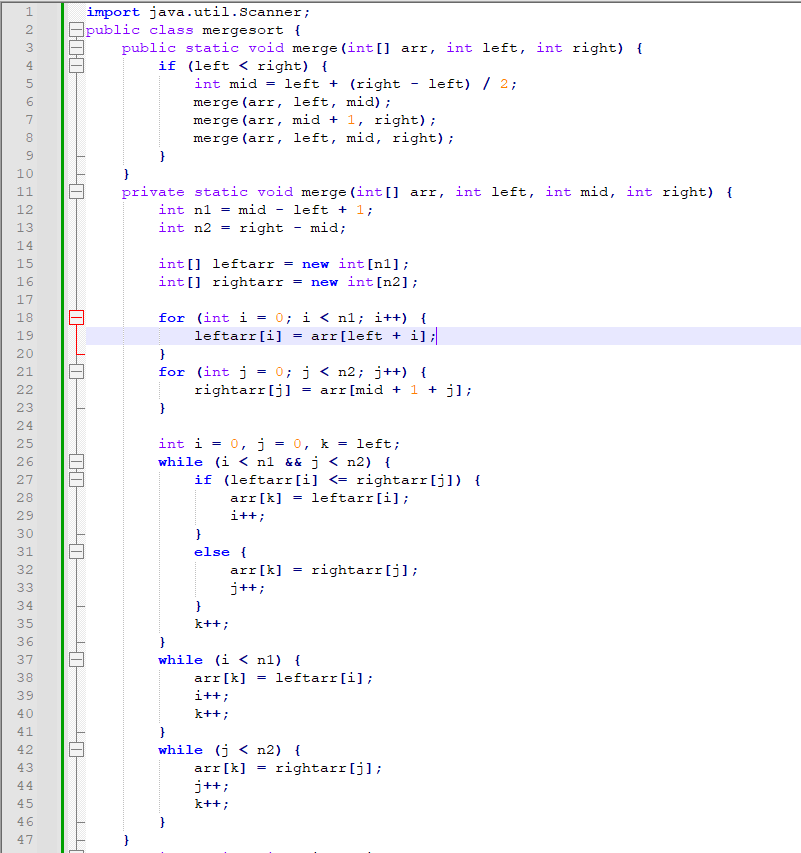
System.out.print(num + " ");

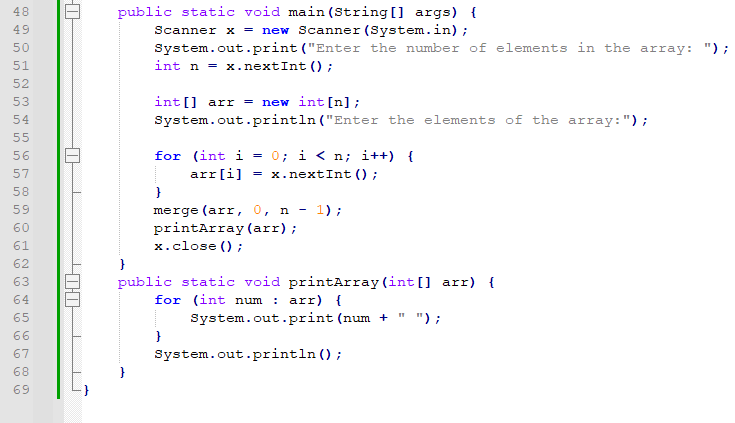
}

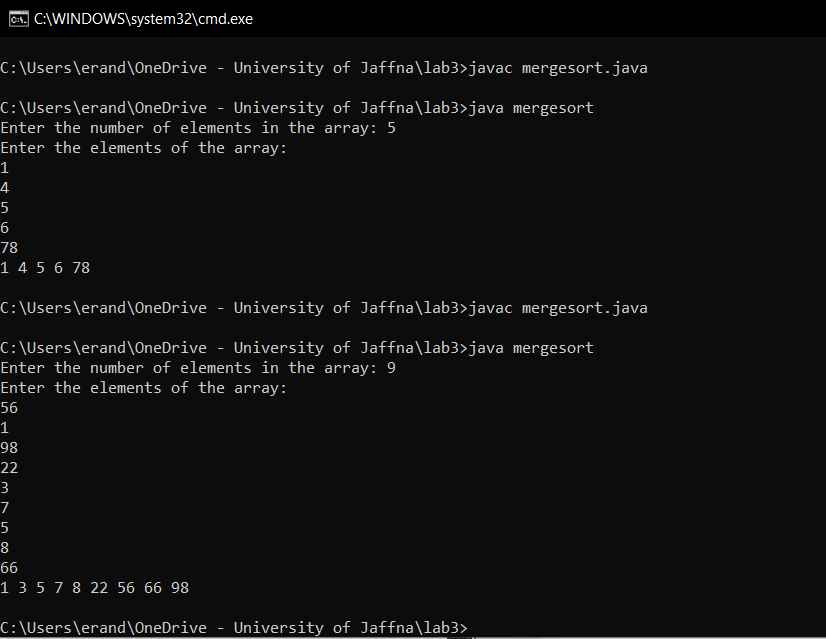
System.out.println();

}

}







import java.util.Scanner;

public class QuickSort {

public static void main(String[] args) {

Scanner x = new Scanner(System.in);

System.out.print("how many elements in your array: ");

int n = x.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements:");

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

arr[i] = x.nextInt();

}

quickSort(arr, 0, n - 1);

System.out.println("Sorted array:");

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

System.out.print(arr[i] + " , ");

}

x.close();

}

public static void quickSort(int[] arr, int low, int high) {

if (low < high) {

int id = ch(arr, low, high);

quickSort(arr, low, id - 1);

quickSort(arr, id + 1, high);

}

}

public static int ch(int[] arr, int low, int high) {

int num1 = arr[high];

int i = low - 1;

for (int j = low; j < high; j++) {

if (arr[j] < num1) {

i++;

swap(arr, i, j);

}

}

swap(arr, i + 1, high);

return i + 1;

}

public static void swap(int[] arr, int i, int j) {

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

