**Practical 1 – Advanced Data Structures Lab**

**1. Write a java program to implement a sequential search algorithm for searching for a key**

**Program:**

**LinearSearch.java:**

package mypack;

import java.util.Scanner;

public class LinearSearch {

public static int linearSearch(int[] num, int size, int key) {

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

if (num[i] == key) {

return i;

}

}

return -1;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int size = scanner.nextInt();

int[] num = new int[size];

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

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

num[i] = scanner.nextInt();

}

System.out.println("Enter the target or key: ");

int key = scanner.nextInt();

int output = linearSearch(num, size, key);

if (output == -1) {

System.out.println("The " + key + " is not found");

} else {

System.out.println("The " + key + " is found at index " + output);

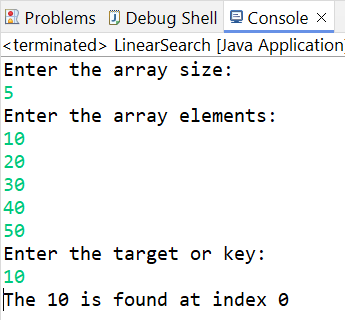
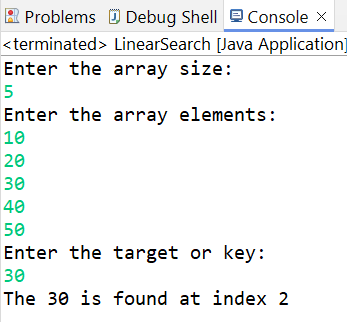
}

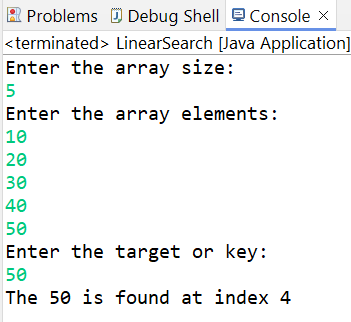
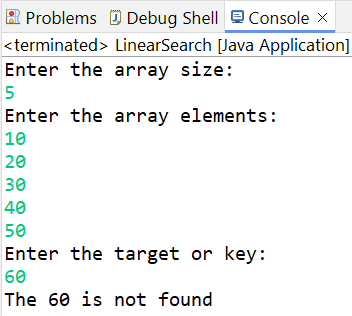
scanner.close();

}

}

**Output:**

** **

** **

**2. Write a java program to implement a recursive binary search algorithm for searching for a key.**

**Program:**

**RecursiveBinarySearch.java**

package mypack;

import java.util.Scanner;

public class RecursiveBinarySearch {

public static int recursiveBinarySearch(int[] num, int low, int high, int key) {

if (low > high) {

return -1; // Base case: key not found

}

int mid = (low + high) / 2;

// Check if the mid element is the key

if (num[mid] == key) {

return mid;

}

// If key is greater, search in the right half

else if (num[mid] < key) {

return recursiveBinarySearch(num, mid + 1, high, key);

}

// If key is smaller, search in the left half

else {

return recursiveBinarySearch(num, low, mid - 1, key);

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int size = scanner.nextInt();

int[] num = new int[size];

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

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

num[i] = scanner.nextInt();

}

System.out.println("Enter the target or key: ");

int key = scanner.nextInt();

// Call the recursive binary search with initial low and high bounds

int output = recursiveBinarySearch(num, 0, size - 1, key);

if (output == -1) {

System.out.println("The " + key + " is not found");

} else {

System.out.println("The " + key + " is found at index " + output);

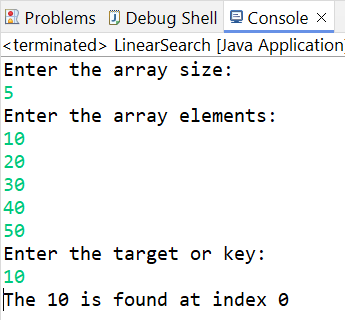
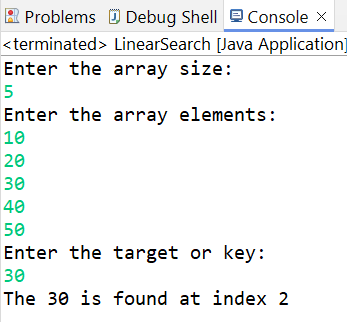
}

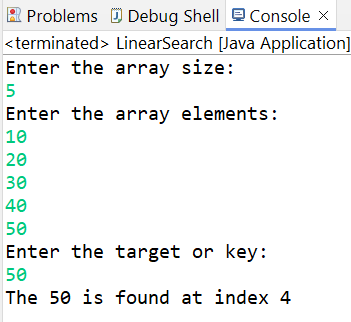
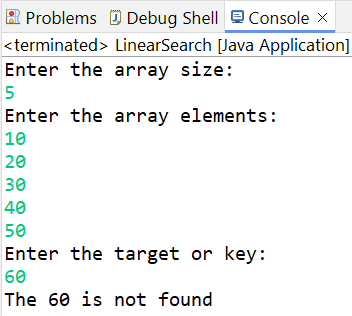
scanner.close();

}

}

**Output:**

** **

** **

**3. Write a java program to implement a Bubble sort algorithm to sort 15 numbers.**

**Program:**

**BubbleSort.java**

package mypack;

import java.util.Scanner;

public class BubbleSort {

public static void bubbleSort(long[] list, long size) {

long hold, walker;

int flag;

for (hold = 0; hold < size - 1; hold++) {

flag = 0; // no swap

for (walker = 0; walker < size - hold - 1; walker++) {

if (list[(int) walker] > list[(int) (walker + 1)]) {

long t;

t = list[(int) walker];

list[(int) walker] = list[(int) (walker + 1)];

list[(int) (walker + 1)] = t;

flag = 1; // done swapping

}

}

if (flag == 0) {

break;

}

System.out.println("\n\nPass: " + (hold + 1) + " : ");

for (long i = 0; i < size; i++) {

System.out.print(list[(int) i] + " ");

}

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

long size = scanner.nextLong();

long[] num = new long[(int) size];

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

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

num[i] = scanner.nextLong();

}

System.out.println("\nUnsorted Array: ");

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

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

}

System.out.println("\n\nSorted array passes:");

bubbleSort(num, size); // num array will be passed by reference and size will be passed by value

System.out.println("\n\nSorted Array: ");

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

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

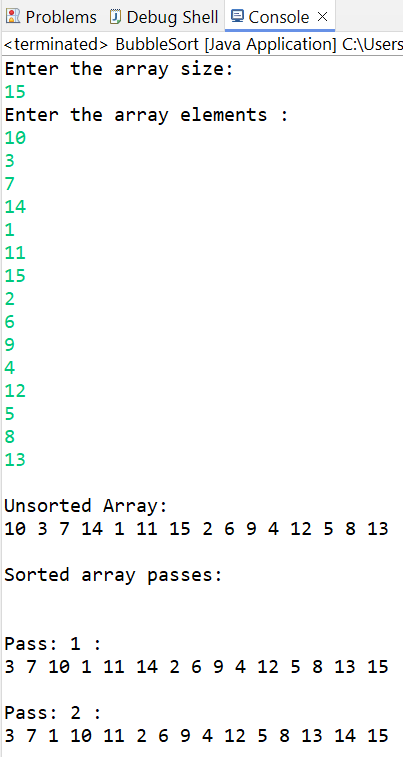
}

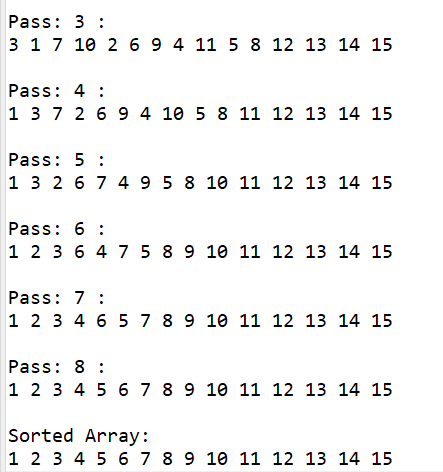
scanner.close();

}

}

**Output:**

****

****

**4. Write a java program to sort an array of 10 numbers using Insertion Sort.**

**Program:**

**InsertionSortExample.java:**

package mypack;

import java.util.Scanner;

public class InsertionSortExample {

public static void insertionSort(int[] list, int n) {

int key, hold, walker, i;

int flag=0;

for (key = 1; key < n; key++) {

walker = key - 1;

hold = list[key];

while (walker >= 0 && hold < list[walker]) {

list[walker + 1] = list[walker];

walker--;

flag = 1;

}

list[walker + 1] = hold;

if (flag == 0) {

break;

}

System.out.print("\nPass : " + key + " - ");

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

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

}

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int size = scanner.nextInt();

int[] list = new int[size];

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

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

list[i] = scanner.nextInt();

}

System.out.print("\n\nUnsorted Array: ");

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

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

}

System.out.println("\n\nSorted array:\n");

insertionSort(list, size);

System.out.print("\n\nSorted Array: ");

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

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

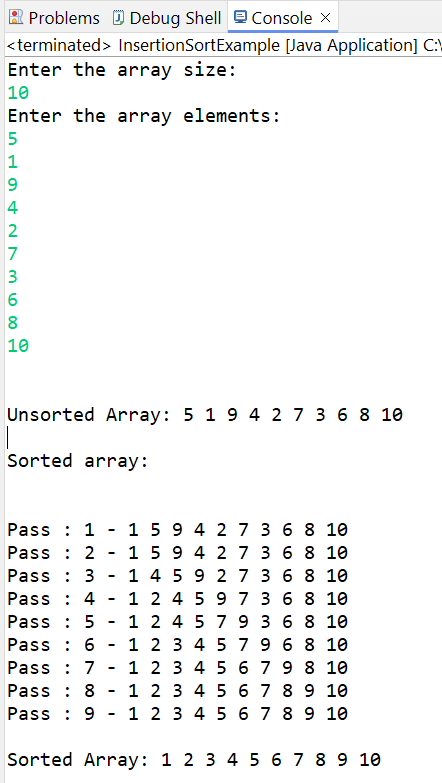
}

scanner.close();

}

}

**Output:**

****

**5. Write a java implement a Selection Sort algorithm for sorting 10 numbers.**

**Program:**

package mypack;

import java.util.Scanner;

public class SelectionSortExample {

public static void selectionSort(long[] list) {

int size = list.length;

for (int hold = 0; hold < size - 1; hold++) {

int pos = hold;

// Find the smallest element in the unsorted part of the array

for (int walker = hold + 1; walker < size; walker++) {

if (list[walker] < list[pos]) {

pos = walker;

}

}

// Swap the found minimum element with the first element if necessary

if (hold != pos) {

long temp = list[pos];

list[pos] = list[hold];

list[hold] = temp;

}

// Print the array after each pass

System.out.print("\nPass " + (hold + 1) + ": ");

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

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

}

}

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the array size: ");

int size = scanner.nextInt();

long[] list = new long[size];

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

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

list[i] = scanner.nextLong();

}

System.out.print("\nUnsorted Array: ");

for (long element : list) {

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

}

System.out.println("\n\nSorting array using selection sort:");

selectionSort(list);

System.out.print("\n\nSorted Array: ");

for (long element : list) {

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

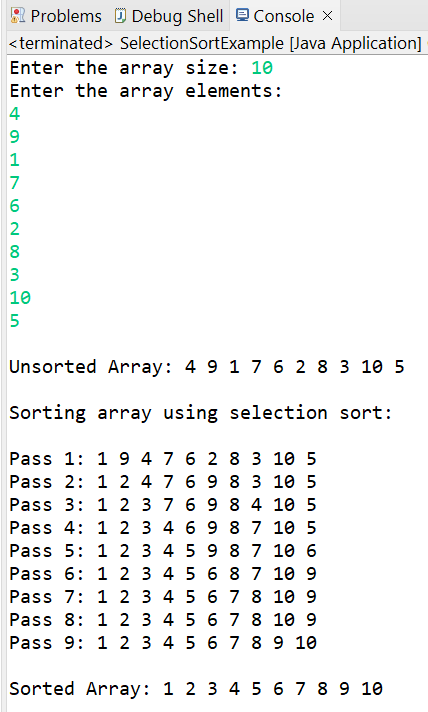
}

scanner.close();

}

}

**Output:**



**6. Write a Java program to sort an array of 15 numbers using Shell Sort. Take input from num.txt file.**

**Program:**

package mypack;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class ShellSortExample {

public static void shellSort(int[] arr, int n) {

int pass = 1;

for (int gap = n / 2; gap > 0; gap /= 2) {

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

int temp = arr[i]; // storing gap element in temp

int j;

for (j = i; j >= gap && arr[j - gap] > temp; j -= gap) {

arr[j] = arr[j - gap]; // moving element if it is greater

}

arr[j] = temp; // placing temp element at appropriate position

}

System.out.println("\nGap = " + gap + "\nPass " + pass + " : ");

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

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

}

pass++;

}

}

public static void main(String[] args) {

int size = 15;

int[] myArray = new int[size];

// Opening the file in read mode

try {

Scanner infile = new Scanner(new File("D:\\MCA\\Eclipse Java\\ADS\\src\\mypack\\num.txt"));

// Reading the values from the file and storing in array

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

myArray[i] = infile.nextInt();

}

// Closing the file

infile.close();

} catch (FileNotFoundException e) {

System.err.println("Failed to open file for reading.");

return;

}

System.out.println("\nArray before sorting: ");

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

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

}

shellSort(myArray, size);

System.out.println("\nArray after sorting: ");

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

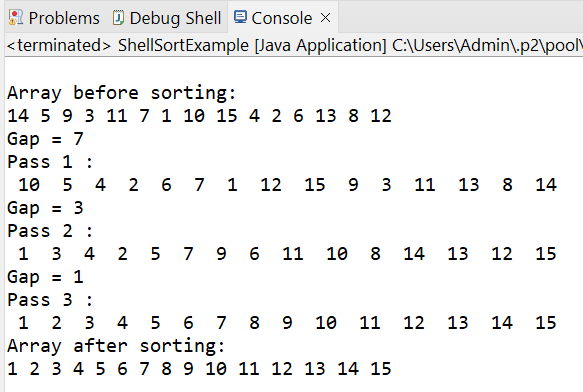
System.out.print(myArray[i] + " ");

}

}

}

**Output:**



**Conclusion: Implemented Sorting and Searching Algorithms Successfully.**