IARE

INSTITUTE OF AERONAUTICAL ENGINEERING

(An Autonomous Institute affiliated to JNTUH, Hyderabad)

Dundigal, Hyderabad - 500 043

LABORATORY WORK BOOK

Name of the Student : N. Ravi Chandrika

Class : CSD - B Semester : IIIrd Semester

Roll Number: 2 3 7 5 1 A 0 6 7 B 3

Course Code : ACSD11 Course Name : OSL

Name of the Course Faculty : Dr. G. Sucharitha Reddy Faculty ID : IARE10805

Exercise Number : 2 Week Number : 2 Date : 10 - 09 - 2024

S. No.	Exercise Number	EXERCISE NAME	MARKS AWARDED					
			Aim/ Preparation	Algorithm / Procedure / Performance in the Lab	Source Code / Calculations and Graphs	Program Execution / Results and Error Analysis	Viva - Voce	Total
			4	4	4	4	4	20
1	2.1	Linear Search						
2	2.2	Binary Search						
3	2.3	Uniform Binary Search						
4	2.4	Interpolation Search						
5	2.5	Fibonacci Search						
6								
7								
8								
9								
10								
11								
12								

N. Ravi Chandrika

Signature of the Student : Signature of the Faculty

START WRITING FROM HERE

2.1 Aim: Linear search is defined as searching algorithm where the list of data is traversed from one end to find the desired value. Give an array of n elements, write a recursive function to search a given element x in an[].

Code:

```
import java.util.Scanner;
class Main{
    public static int linearSearch(int[] array, int x){
        boolean istrue = true;
        int result = 0;
        for(int i=0; i<array.length; i++){
            if(x==array[i]){
                istrue = false;
                result+=i;
            }
        }
        if(istrue){
            return -1;
        }
        else{
            return result;
        }
    }
public static void main(String[]args){
    Scanner sc=new Scanner(System.in);
    int n=sc.nextInt();
    String m=sc.nextLine();
```

```
String[] inputA = sc.nextLine().split(" ");
int[] array = new int[n];
for(int i=0; i<n; i++){
    array[i] = Integer.parseInt(inputA[i]);
}
int x = sc.nextInt();
System.out.println(linearSearch(array, x));
}
}
```

Input: arr[] = {10, 20, 80, 30, 60, 50, 110, 100, 130, 170}

x = 110;

Output: 6

Element x is present at index 6.

Aim: Binary Search is defined as a searching algorithm used in a sorted array by repeatedly dividing the search interval in half. The idea of binary search is to use the information that the array is sorted and reduce the complexity to O(log n).

Code:

```
import java.util.Scanner;
import java.util.Arrays;
class Main {
    public static int BinarySearch(int[] arr, int target){
```

```
        Arrays.sort(arr);
        int low = 0;
        int high = arr.length - 1;
        while (low <= high) {
            int mid = (low + high) / 2;
            if (arr[mid] == target) {
                return mid;
            }
            if (arr[mid] < target) {
                low = mid + 1;
            }
            if (arr[mid] > target) {
                high = mid - 1;
            }
        }
        return -1;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String m = sc.nextLine();
        String[] input = sc.nextLine().split(" ");
        int[] arr = new int[n];
        for (int i=0; i<n; i++) {
            arr[i] = Integer.parseInt(input[i]);
```

```
        }
        int target = sc.nextInt();
        System.out.println(binarySearch(arr,target));
    }
}
```

Input: arr = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

x = 23

Output: target = 23

Element 23 is present at index 5.

2.3 Aim: Uniform Binary Search is an Optimization of Binary search algorithm when many searches are made on same array of many arrays of same size.

Code:

```
import java.util.Scanner;
Class Main{
    public static int binarySearch(int[] arr, int x){
        int low=0;
        int high= array.length-1;
        while (low<=high){
            int mid = (low+high)/2;
            if (arr[mid]==x){
                return mid;
            }
            if (arr[mid]<x){
```

```
            low = mid + 1;
        }
        if (arr[mid] > x) {
            high = mid - 1;
        }
    }
    return -1;
}
public static void main (String args[]) {
    Scanner sc = new Scanner (System.in);
    int n = sc.nextInt();
    String m = sc.nextLine();
    String[] input = sc.nextLine().split(" ");
    int[] arr = new int[n];
    for (int i=0; i<n; i++) {
        arr[i] = Integer.ParseInt (input[i]);
    }
    int x = sc.nextInt();
    System.out.println(binarySearch(arr, x));
}
}
```

Input: array = {1, 3, 5, 6, 7, 8, 9}, V = 3

Output: Position of 3 in array = 2.

ROLL NUMBER :

2.4 Aim: Interpolation search works better than Binary search for a sorted & uniformly distributed array. Binary search goes to middle element to check, irrespective of search key. While interpolation goes to different locations wrt to search key.

Code:

```
import java.util.Scanner;
public class Main{
    public static int interpolationSearch(int[] arr, int n, int x){
        int low = 0, high = n-1;
        while(low<=high && x>=arr[low] && x<=arr[high]){
            if(low==high){
                if(arr[low]==x) return low;
                return -1;
            }
            int pos= low + ((x-arr[low])*(high-low))/
                                (arr[high]-arr[low]);
            if(arr[pos]==x) return pos;
            if(arr[pos]<x) low = pos+1;
            else high = pos-1;
        }
        return -1;
    }
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
```

```
        int n = scanner.nextInt();
        scanner.nextLine();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        int target = scanner.nextInt();
        int index = interpolationSearch(arr, n, target);
        if (index != -1) {
            System.out.println(index);
        } else {
            System.out.println("-1");
        }
        scanner.close();
    }
}
```

Input: arr = [1,2,3,4,5,6,7,8,9]

Output: Target = 5.

5 Aim: Given a sorted array arr[] of size n and an element x to be searched in it. Return index of x if it is present.

Code:

```
import java.util.Scanner;
public class main {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the no.of fibonacci");
        int n = scanner.nextInt();
        System.out.print("Sequence upto " + n + " terms:");
        int a = 0, b=1;
        for(int i=0; i<=n; i++) {
            System.out.print(a + " ");
            int next = a+b;
            a =b;
            b= next;
        }
        scanner.close();
    }
}.
```

Input : arr[] = {2,3,4,10,40}, x = 40

Output: 3

Element x is present at index 3.