Name: ZOHAIB HASSAN SOOMRO

RollNo#: 19SW42

Subject: DSA

Task#1: Apply linear and binary search algorithm on array of string and characters also print execution time of each result.

Code:

```
public class Task1_ExecutionTime {
    public static <E> int linearSearch(E[] array, E target)
{
        if (array.length == 0)
            return -1;
        for (int i = 0; i < array.length; i++) {
            if (array[i].equals(target))
                return i;
        }
        return -1;
    }
```

```
    public static <E> int binarySearch(E[] array, E target)
{
        if (array.length == 0)
            return -1;
        int start=0;
        int end=array.length-1;
        while (start<=end) {
            int mid= (start+end)/2;
            if (array[mid].equals(target)) return mid;
            else
if(array[mid].toString().compareTo(target.toString())<0)
                start=mid+1;
            else end=mid-1;
        }
        return -1;
    }
```

```
    public static void main(String[] args) {
        // System.currentTimeMillis();
        String strArray[] = { "Ali", "Is", "Lover", "Of",
"Sara" };
```

```
        Character charArray[] = {'d', 'e', 'f', 'g'
,'h','i','j'};
        long miliSecsStart = System.nanoTime();
        System.out.println("linearSearch(strArray,
\"This\") index = " +linearSearch(strArray, "This"));
        long miliSecsEnd = System.nanoTime();
        System.out.println("linearSearch(strArray,
\"This\") Execution time = " + (miliSecsEnd -
miliSecsStart)+" ns\n");

        miliSecsStart = System.nanoTime();
        System.out.println("linearSearch(charArray, 'i')
index = " +linearSearch(charArray, 'i'));
        miliSecsEnd = System.nanoTime();
        System.out.println("linearSearch(charArray, 'i')
Execution time = " + (miliSecsEnd - miliSecsStart)+"
ns\n\n");
        miliSecsStart = System.nanoTime();
        System.out.println("binarySearch(strArray,
\"Lover\") index = " +binarySearch(strArray, "Lover"));
        miliSecsEnd = System.nanoTime();
        System.out.println("binarySearch(strArray,
\"Lover\") Execution time = " + (miliSecsEnd -
miliSecsStart)+" ns\n");
        miliSecsStart = System.nanoTime();
        System.out.println("binarySearch(charArray, 'i')
index = " + binarySearch(charArray, 'i'));
        miliSecsEnd = System.nanoTime();
        System.out.println("binarySearch(charArray, 'i')
Execution time = " + (miliSecsEnd - miliSecsStart)+" ns");

    }

}
```

```
<terminated> Task1_ExecutionTime [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (Jan 27, 20
linearSearch(strArray, "This") index = -1
linearSearch(strArray, "This") Execution time = 183400 ns

binarySearch(strArray, "Lover") index = 2
binarySearch(strArray, "Lover") Execution time = 22700 ns

linearSearch(charArray, 'i') index = 5
linearSearch(charArray, 'i') Execution time = 83600 ns

binarySearch(charArray, 'i') index = 5
binarySearch(charArray, 'i') Execution time = 33900 ns
```

Task#2: Apply linear and binary search algorithm on an array of any user defined object of your type.

Code:

```
class MyRectangle implements Comparable {
    int area;

    public MyRectangle(int length, int width) {
        area = length * width;
    }

    @Override
    public int compareTo(Object obj) {
        if (this.area < (int) obj)
            return -1;
        if (this.area > (int) obj)
            return 1;
        return 0;
    }
```

```
    @Override
    public boolean equals(Object obj) {
        return (compareTo(obj) == 0);
    }
}

public class Task2_UserDefined {
    public static int linearSearch(MyRectangle[] array, int
target) {
        if (array.length == 0)
            return -1;
        for (int i = 0; i < array.length; i++) {
            if (array[i].equals(target))
                return i;
        }
        return -1;

    }

    public static int binarySearch(MyRectangle[] array, int
target) {
        if (array.length == 0)
            return -1;
        int start = 0;
        int end = array.length - 1;
        while (start <= end) {
            int mid = (start + end) / 2;
            if (array[mid].equals(target))
                return mid;
            else if (array[mid].compareTo(target) < 0)
                start = mid + 1;
            else
                end = mid - 1;
        }
        return -1;
    }

    public static void main(String[] args) {
        MyRectangle rectangles[] = new MyRectangle[6];
```

```
        for (int i = 0; i < rectangles.length; i++) {
            rectangles[i] = new MyRectangle(i + 4, i + 5);
// area=20,30,42,58,72,90
        }

        System.out.println("Index of rectangle having area=
42 using... \nbinarySearch(rectangles,42): "
                + binarySearch(rectangles, 42));


        System.out.println("\n\nIndex of rectangle having
area= 56 using... \nlinearSearch(rectangles,56): "
                + linearSearch(rectangles, 56));

        System.out.println("\n\nIndex of rectangle having
area= 57 using... \nbinarySearch(rectangles,57): "
                + binarySearch(rectangles, 57));


    }

}
```

```
<terminated> Task2_UserDefined [Java Application] C:\Program Files\Java\jre1.8.0_261\b
Index of rectangle having area= 42 using...
binarySearch(rectangles,42): 2


Index of rectangle having area= 56 using...
linearSearch(rectangles,56): 3


Index of rectangle having area= 57 using...
binarySearch(rectangles,57): -1
```

Task#3: Apply linear and binary search algorithm on a 2-D integer array (+1 marks).

Code:

```
import java.util.Arrays;

public class Task3_2DSearch {
	public static int[] linearSearch(int[][] array, int target) {
		if (array.length == 0)
			throw new NullPointerException("Array has no element!");
		int index[] = { -1, -1 };
		for (int i = 0; i < array.length; i++) {
			for (int j = 0; j < array[i].length; j++) {
				if (array[i][j] == (target)) {
					index[0] = i;
					index[1] = j;
					return index;
				}
			}
		}
		return index;
	}

	public static int[] binarySearch(int[][] array, int target) {
		if (array.length == 0)
			throw new NullPointerException("Array has no element!");
		int into1DArray[] = new int[array.length * array[0].length];
		for (int i = 0; i < array.length; i++) {
			System.arraycopy(array[i], 0, into1DArray, i * array[i].length,
array[i].length);
		}
		int start = 0;
		int end = into1DArray.length - 1;
		while (start <= end) {
			int mid = (start + end) / 2;
			if (into1DArray[mid] == target) {
				int indices[] = { mid / array[0].length, mid %
array[0].length };
				return indices;
			} else if (into1DArray[mid] < target)
				start = mid + 1;
			else
				end = mid - 1;

		}
		return new int[] { -1, -1 };
	}

	public static void main(String[] args) {
		int[][] array = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
		int target = 5;
		System.out.println("Whole 2D-Array: ");
		for (int i = 0; i < array.length; i++) {
			System.out.println(Arrays.toString(array[i]));
		}
```

```
            int indices[] = linearSearch(array, target); // as there willbe two
indexes
            System.out
                        .println("Index of " + target + " using linearSearch
algorithm = [" + indices[0] + "][" + indices[1] + "]");

            target = 7;

            indices = binarySearch(array, target);
            System.out
                        .println("Index of " + target + " using binarySearch
algorithm = [" + indices[0] + "][" + indices[1] + "]");

      }
}
```