DHA SUFFA UNIVERSITY
Department of Computer Science
Assignment-02 – Fall - 24
(Data Structures and Algorithms SE_2001)
SECTION: SE-3B

Course Instructor: **Ms. Sumera Rounaq** Max Marks: 10

Student's Name: _____ Reg. No: _____

Instructions:

- **Submit your assignment in soft copy.**
- **Due deadline of this assignment is 31st of December 2024.**
- **Late Submission will be subjected to deduction of two marks per day.**
- **Create MS word file and paste your code and snapshots of your output and then convert it in pdf file.**
- **Submit your pdf file and your java files.**
- **Rename your files as "Yourname-YourReg No"**

1. Explain the concept of binary search in your own words.
    - Define binary search.

      Binary search is a method used to find an item in a list quickly. Instead of checking every item one by one (like in linear search), it starts by checking the middle item in the list. If the middle item is the one you're looking for, you're done. If not, it decides whether to look in the left half or the right half of the list, depending on whether the middle item is smaller or larger than the target. It keeps repeating this process until the item is found or there's nothing left to check.
    - Describe how it differs from linear search.

      How They Work:

        - Linear search goes through the list one item at a time from the start until it finds the target or reaches the end.
        - Binary search starts in the middle of a sorted list and decides whether to search in the left half or the right half, cutting the search space in half each time.

      Speed:

        - Linear search can be slow because it checks every item, especially for long lists.
        - Binary search is much faster for large lists because it eliminates half of the list with each step.

      List Requirement:

- Linear search works on any list, whether it's sorted or not.
- Binary search only works on lists that are sorted beforehand.

Efficiency:

- Linear search is less efficient, especially when the list is very large.
- Binary search is highly efficient for searching in large sorted lists.

Use Cases:

- Linear search is used when the list is small or unsorted.
- Binary search is used when the list is sorted, and you want quick results.

- List the prerequisites for using binary search.

  The list must **already be sorted**.

  You should be able to access any item directly (e.g., in an array).

  You need a way to compare items to know their order.

2. Real-World Applications:
   - Identify and explain three real-world applications where binary search is used.
   - Justify why binary search is suitable for these applications.

     Finding a word in a dictionary:
     Dictionaries are sorted, so binary search is used to find words quickly.
     Why it works: The list is sorted alphabetically, perfect for binary search.

     Looking up data in databases:
     Databases often use sorted data for faster searches.
     Why it works: Binary search reduces the time needed to find data.

     Searching for a name in a leaderboard:
     If the leaderboard is sorted by scores, binary search can find a player's rank quickly.
     Why it works: It's efficient when the data is sorted.
3. Write a Java program to implement binary search for a given sorted list of integers.

   The program should:
   - Accept a sorted list and the target value as input.
   - Return the index of the target value if found, or -1 if not found.

```
import java.util.Scanner;


public class BinarySearch {
```

```
public static int binarySearch(int[] sortedArray, int target) {
    int left = 0;
    int right = sortedArray.length - 1;

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

        if (sortedArray[mid] == target) {
            return mid;
        }
        if (sortedArray[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;            }
    }
    return -1;    }

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter the size of the array:");
    int size = scanner.nextInt();
    int[] sortedArray = new int[size];

    System.out.println("Enter the elements of the sorted array:");
    for (int i = 0; i < size; i++) {
        sortedArray[i] = scanner.nextInt();
    }

    System.out.println("Enter the target value to search:");
    int target = scanner.nextInt();

    int result = binarySearch(sortedArray, target);
    if (result == -1) {
        System.out.println("Target value not found.");
    } else {
```

```
            System.out.println("Target value found at index: " + result);
        }


        scanner.close();
    }
}
```

**********GOOD LUCK**********