|  |  |
| --- | --- |
| Data Structures & Algorithms Diploma in IT, CSF  Year 2 (2024/25) Semester 4 | Week 8 |
| 2 Hours |
| **Practical 8 – Searching Techniques** | |

**Objectives**

At the end of this practical, the students should be able to:

* Under concept of searching
* Implement iterative and recursive sequential search for sorted arrays
* Implement iterative and recursive binary search for sorted arrays

|  |
| --- |
| **IMPORTANT**   * Upload all your work to DSA network folder **\\ictspace.ict.np.edu.sg\dsa**   by the designated timeline given by your tutor |

1. Write an iterative ***sequential search*** function to search for a target in a sorted array of integer numbers. The function header is given below.

*// search for an item in a sorted array using sequential search (Iterative version)*

*// returns the index of the item in the array if found, otherwise returns -1*

**int search (int dataArray[], int arraySize, int target)**

1. Write a sample program to do the following:
2. Declare an integer array, **dataArray**, of size 1000.
3. Initialize the **dataArray** with even numbers below:

2, 4, 6, . . ., 1996, 1998, 2000

1. Prompt the user to enter a target number to search
2. Read the number entered and stored it to a variable, **target**
3. Invoke the **search()** method above to search for the target and display the result

(i.e. display “Found” or “Not found”)

1. Write an iterative ***binary search*** function to search for a target in a sorted array of integer numbers. The function header is given below.

*// search for an item in a sorted array using binary search (Iterative version)*

*// returns the index of the item in the array if found, otherwise returns -1*

**int binarySearch (int dataArray[], int arraySize, int target)**

1. Modify the sample program in (2) above to invoke the **binarySearch()** method and display the result (i.e. display “Found” or “Not found”)
2. Modify the search() methods to count the number of **comparisons**

* Use a static global variable, e.g. static int comparisons = 0;
* Increment the value each time a comparison is made e.g. comparisons++;

1. Modify the sample program to invoke both the sequential search and binary search as follows:

comparisons = 0;

int idx1 = search(numArray, n, 0, target);

cout << "Sequential Search : " << comparisons << endl;

comparisons = 0;

int idx2 = binarySearch(numArray, 0, n-1, target);

cout << "Binary Search : " << comparisons << endl;

1. Run the sample program with different input numbers and analyze the results

|  |  |  |
| --- | --- | --- |
| **Target** | **Sequential Search**  **(Comparisons)** | **Binary Search**  **(Comparisons)** |
| 1 | 1000 | 10 |
| 2 | 1 | 1 |
| 99 | 50 | 10 |
| 100 | 50 | 10 |
| 999 | 500 | 10 |
| 1000 | 500 | 10 |
| 1999 | 1000 | 10 |
| 2000 | 1000 | 10 |

1. Write a ***recursive*** sequential search function to search for a target in a sorted array of integer numbers. The function header is given below.

*// search for an item in a sorted array using sequential search (Recursive version)*

*// returns the index of the item in the array if found, otherwise returns -1*

**int search (int dataArray[], int arraySize, int start, int target)**

1. Write a ***recursive*** binary search function to search for a target in a sorted array of integer numbers. The function header is given below.

*// search for an item in a sorted array using binary search (Recursive version)*

*// returns the index of the item in the array if found, otherwise returns -1*

**int binarySearch (int dataArray[], int first, int last, int target)**