Experiment No. : 08

Title: Implementation of searching algorithm

Batch:A3 **Roll No.:16010423099** **Experiment No 8**

Aim: Demonstrate the use of Sorting in binary searching algorithm

---

Resources Used: Turbo C/ C++ editor and C compiler.

Theory:

Searching –

Search is a process of finding a value in a list of values. In other words, searching is the process of locating given value position in a list of values.

The **binary search algorithm** can be used with only sorted list of element. That means, binary search can be used only with list of element which are already arranged in a order. The binary search cannot be used for list of element which are in random order. This search process starts comparing of the search element with the middle element in the list. If both are matched, then the result is "element found". Otherwise, we check whether the search element is smaller or larger than the middle element in the list. If the search element is smaller, then we repeat the same process for left sub-list of the middle element. If the search element is larger, then we repeat the same process for right sub-list of the middle element. We repeat this process until we find the search element in the list or until we left with a sub-list of only one element. And if that element also doesn't match with the search element, then the result is "Element not found in the list".
By implementing and observing the interplay between sorting and binary search, this lab aims to highlight the importance of a sorted dataset for efficient binary search operations.

Algorithm :

1) **PreCondition -** Sort the given input array using any sorting algorithm as counting sort**,** quick sort or merge sort
2) **Binary Search algorithm -**

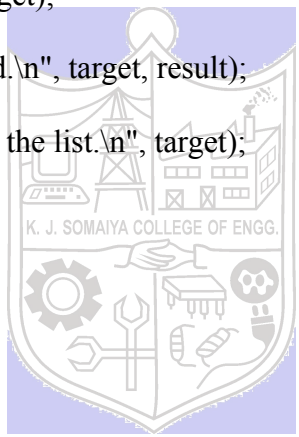
```
BinarySearch(list[], min, max, key)
if max <min then
   return  false
else
   mid = (max+min) / 2
   if list[mid] >key then
      return  BinarySearch(list[], min, mid-1, key)
   else if list[mid] <key then
      return  BinarySearch(list[], mid+1, max, key)
   else
      return  mid
   end if
end if
```

Activity:
1. Write C program to demonstrate the use of Sorting in binary search algorithm.
2. Demonstrate sorting and searching algorithms in Virtual Lab (screen shots of simulation result and quiz)
https://www.vlab.co.in/broad-area-computer-science-and-engineering

Program:

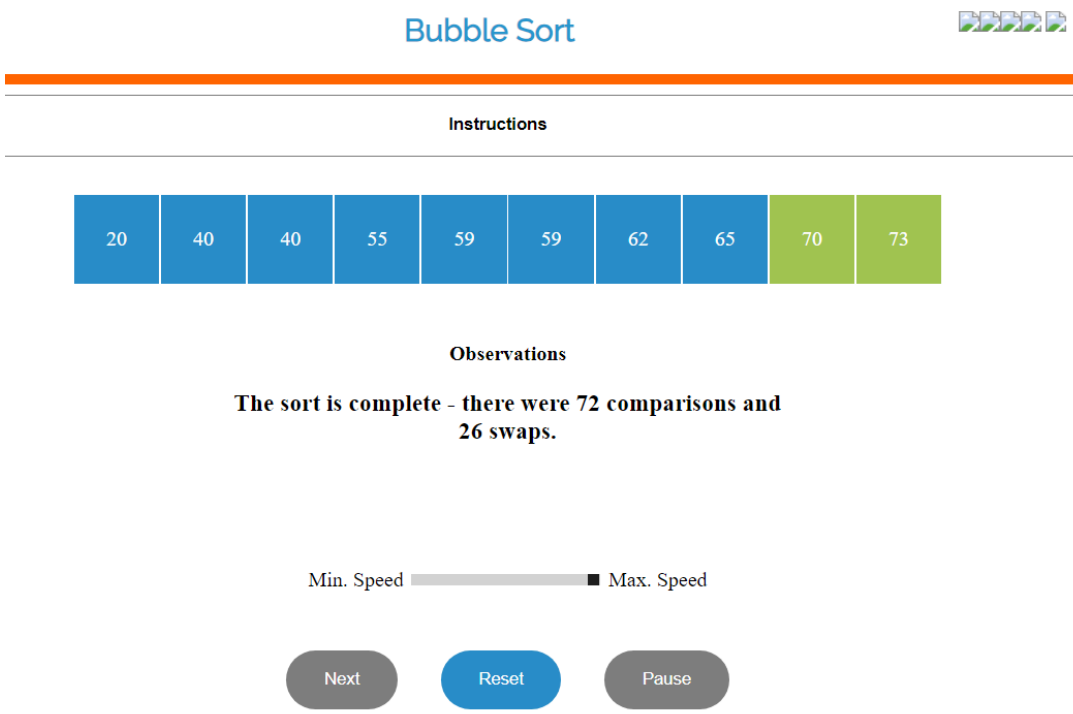
```
#include <stdio.h>
#include <stdlib.h>

void bubbleSort(int arr[], int n) {
    int swapped;
    for (int i = 0; i < n - 1; i++) {
        swapped = 0;
        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
                swapped = 1;
            }
        }
        if (!swapped) break;
    }
}

int binarySearch(int arr[], int n, int target) {
    int left = 0, right = n - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == target) return mid;
        if (arr[mid] < target) left = mid + 1;
        else right = mid - 1;
    }
    return -1;
}

int main() {
    int n, target;

    printf("Sorting and Searching Program\n");
    printf("Enter the number of elements: ");
    scanf("%d", &n);

    int arr[n];
```

```
    printf("Enter %d numbers:\n", n);
    for (int i = 0; i < n; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &arr[i]);
    }

    printf("Sorting the numbers...\n");
    bubbleSort(arr, n);

    printf("Here's the sorted list:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    printf("Enter a number to search: ");
    scanf("%d", &target);

    int result = binarySearch(arr, n, target);
    if (result != -1) {
        printf("Found %d at position %d.\n", target, result);
    } else {
        printf("The number %d is not in the list.\n", target);
    }

    return 0;
}
```



Output:

```
D:\MinGW\stuff\16010423099_EXP8_DS.exe
Sorting and Searching Program
Enter the number of elements: 10
Enter 10 numbers:
Element 1: 4
Element 2: 23
Element 3: 4
Element 4: 1
Element 5: 234
Element 6: 56
Element 7: 12
Element 8: 21
Element 9: 532
Element 10: 9
Sorting the numbers...
HereÆs the sorted list:
1 4 4 9 12 21 23 56 234 532
Enter a number to search: 21
Found 21 at position 5.

Process returned 0 (0x0)   execution time : 26.361 s
Press any key to continue.
```



Bubble Sort

Choose difficulty: ☑ Beginner ☑ Intermediate

1. How many iterations of the outer loop are needed by the algorithm when the input array of size N is already sorted?
- a: 0 Explanation
- b: 1 Explanation
- c: N Explanation
- d: 2N Explanation

2. How many comparisons (same as the number of iterations of the inner loop) are required in the next iteration after T iterations of the outer loop in the optimized algorithm? The array size is N.
- a: N Explanation
- b: N - 1 Explanation
- c: N - T Explanation
- d: N - T + 1 Explanation

3. How do we check if our array is sorted in order to preemptively stop the optimized algorithm?
- a: Run an extra iteration as part of the current iteration and check that all adjacent elements are in the right order. Explanation
- b: Check if any swaps occurred in the current iteration. Explanation
- c: There is no way to preemptively stop the algorithm, i.e. a minimum of N - 1 iterations is required where N is the array size. Explanation
- d: None of the above Explanation

Submit Quiz

Score: 3 out of 3

Merge Sort

Instructions

6 | 23 54 12 52 21 67

Setup | Reset | Next

Observations

Done sorting!

23	54	12	52	21	67
0	1	2	3	4	5

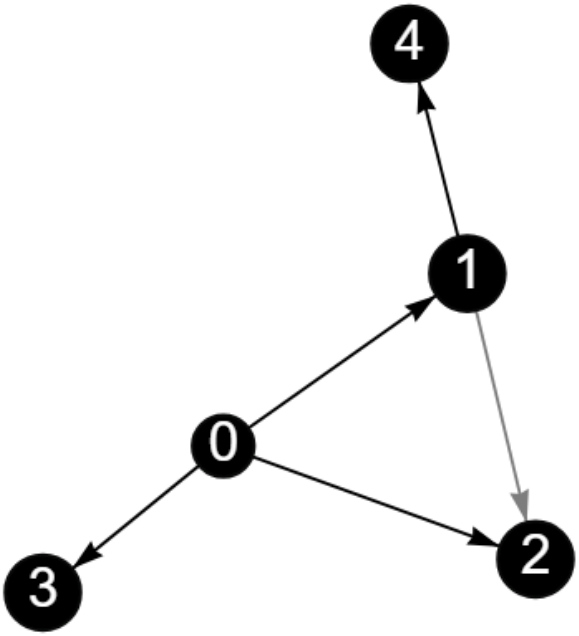
12		
0	1	2

52	21	67
0	1	2

23	54
0	1

0

Breadth First Search

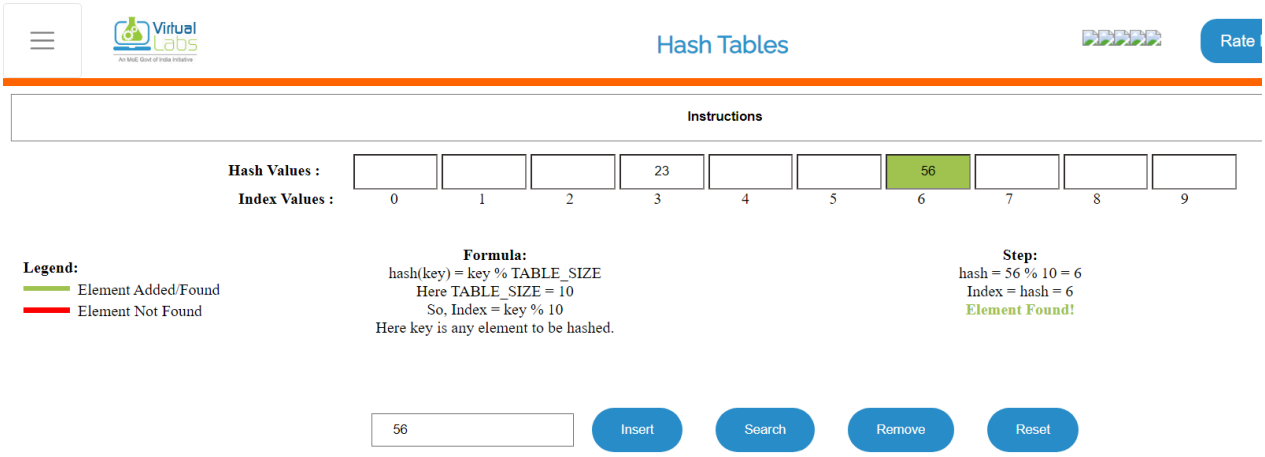


Observations

BFS is done on the node **0** !!!
Sequence of nodes visited on performing BFS on node 0 : 0 , 1 , 2 , 3 , 4

Min.Speed Max.Speed

Reset | New graph | Pause | Next



Outcome:

CO4: Demonstrate sorting and searching methods.

Conclusion:

Successfully applied the concept and algorithm of binary search and executed the program.

References:

Books/ Journals/ Websites:

- Y. Langsam, M. Augenstin and A. Tenenbaum, “Data Structures using C”, Pearson Education Asia, 1st Edition, 2002.
- Vlabs on binary search and counting sort.