Experiment No. : 08

Title: Implementation of searching algorithm

Batch: SY_IT(B3) **Roll No.:16010423076** **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 -**

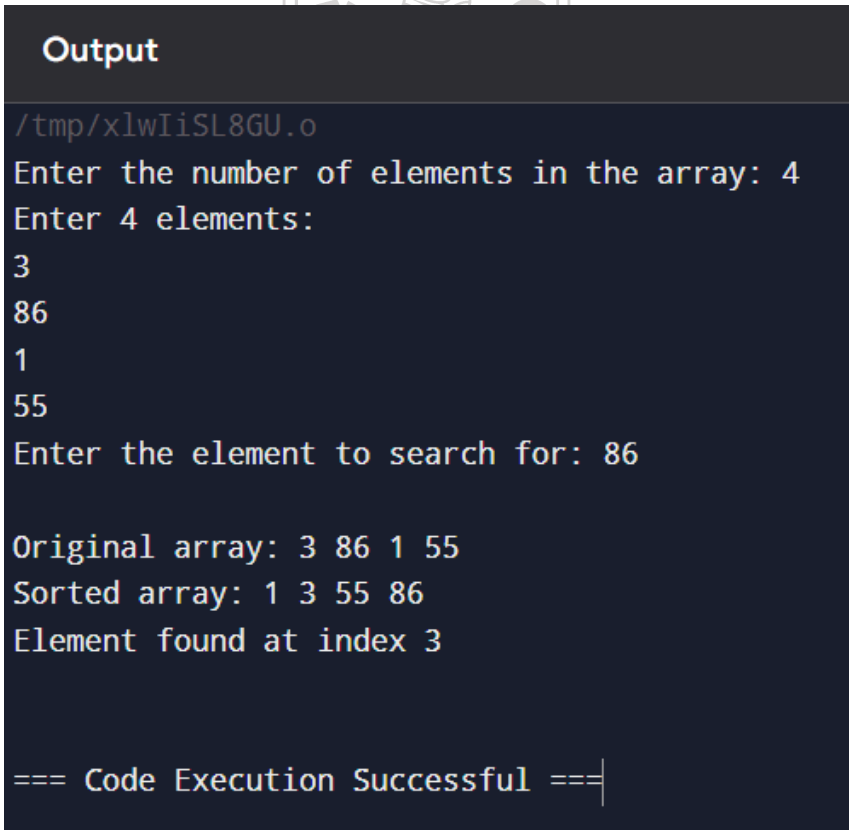
```
BinarySearch(list[], min, max, key)
if max <min then
   return  false
else
   mid = (max+min) / 2
   if list[mid] >key then
      return  BinarySearch(list[], min, mid-1, key)
   else if list[mid] <key then
      return  BinarySearch(list[], mid+1, max, key)
   else
      return  mid
   end if
end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

Activity:
1. Write C program to demonstrate the use of Sorting in binary search algorithm.

Code :
```
#include <stdio.h>

void bubblesort(int array[], int n);

int binarySearch(int array[], int low, int high, int x) {
   while (low <= high) {
      int mid = low + (high - low) / 2;
      if (array[mid] == x)
         return mid;
      if (array[mid] < x)
         low = mid + 1;
      else
         high = mid - 1;
   }
   return -1;
}

void bubblesort(int array[], int n) {
   for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - i - 1; j++) {
         if (array[j] > array[j + 1]) {
            // Swap
            int temp = array[j];
            array[j] = array[j + 1];
            array[j + 1] = temp;
         }
      }
   }
}

int main() {
   int n, x, i;

   printf("Enter the number of elements in the array: ");
   scanf("%d", &n);

   int array[n];

   printf("Enter %d elements:\n", n);
   for (i = 0; i < n; i++) {
      scanf("%d", &array[i]);
   }

   printf("Enter the element to search for: ");
```

```
    scanf("%d", &x);

    printf("\nOriginal array: ");
    for (i = 0; i < n; i++) {
        printf("%d ", array[i]);
    }

    bubblesort(array, n);

    printf("\nSorted array: ");
    for (i = 0; i < n; i++) {
        printf("%d ", array[i]);
    }

    int result = binarySearch(array, 0, n - 1, x);

    if (result == -1)
        printf("\nElement not found\n");
    else
        printf("\nElement found at index %d\n", result);

    return 0;
}
```

Output :



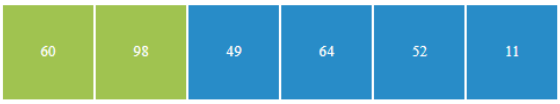
2. Demonstrate sorting and searching algorithms in Virtual Lab (screen shots of simulation result and quiz)
https://www.vlab.co.in/broad-area-computer-science-and-engineering

Bubble Sort :

Practice:

Exercise :

Instructions

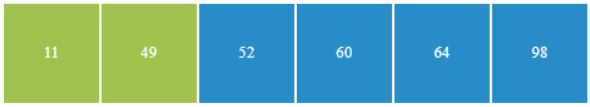


Observations

Number of iterations: 0

Submit Next Swap Undo Reset

Instructions

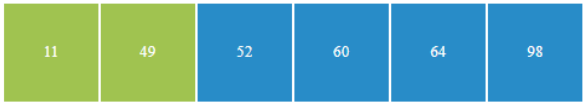


Observations

Number of iterations: 4

Submit Next Swap Undo Reset

Instructions

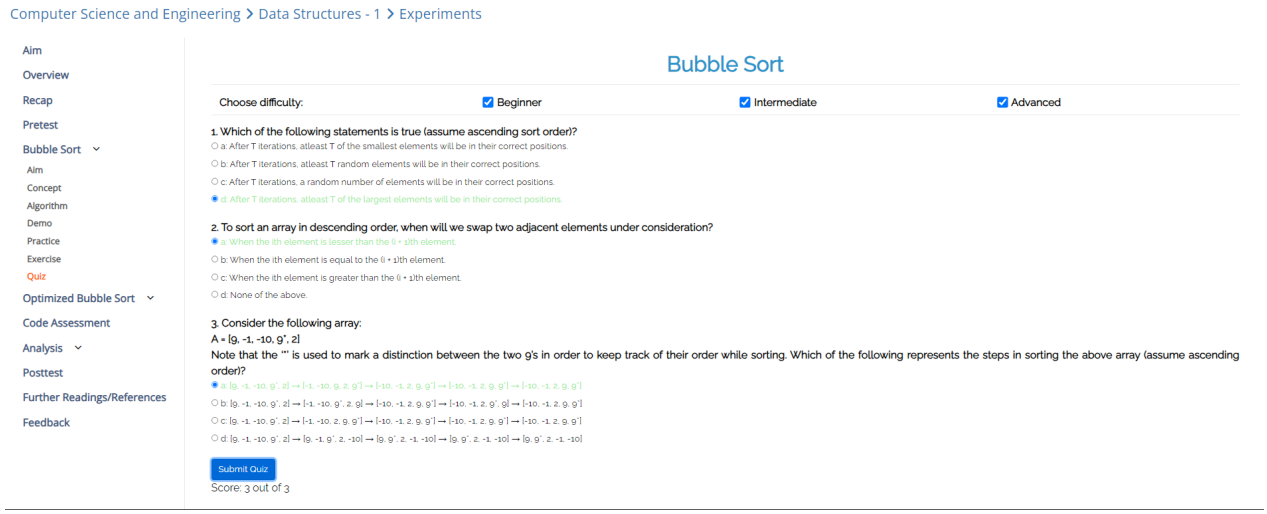


Observations

CORRECT ANSWER

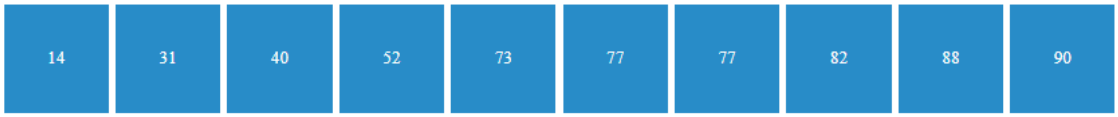
Submit Next Swap Undo Reset

Quiz :



Binary Search :

Practice :



Observations

Correct Answer, Great Job!

Number To be Searched: 77 Binary Search Order: 4, 1, 0

Check Answer

Exercise :

Instructions

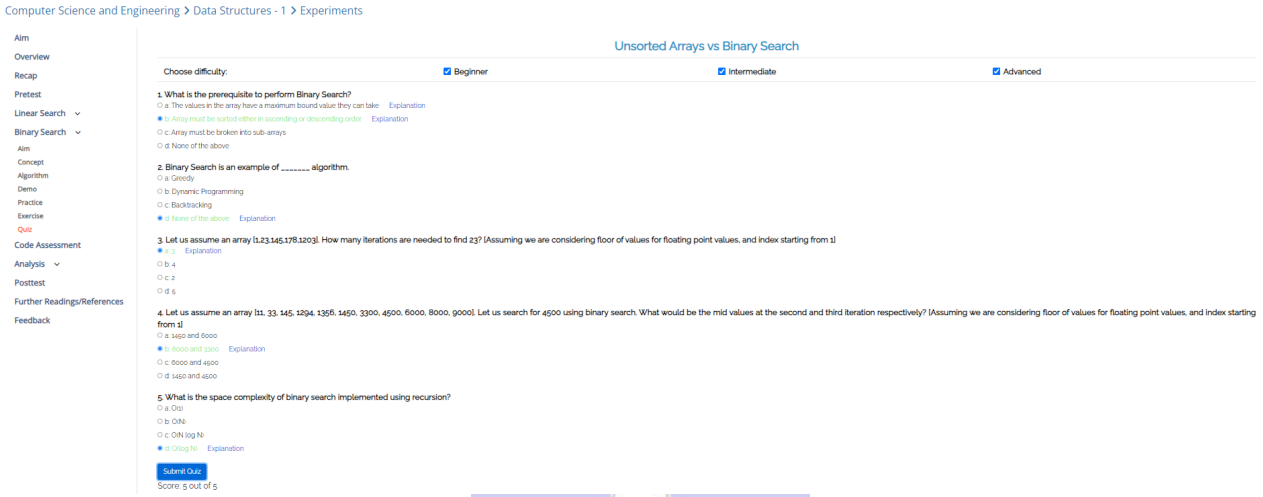
Observations

31 was found in the array at position 3. Binary Search Complete. Great Work!

Min. Speed Max. Speed

31 Reset

Quiz ::



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.