Department of
Computer Science and Engineering

Title: Implement Linear Search Algorithm

Data Structure Lab

CSE 106

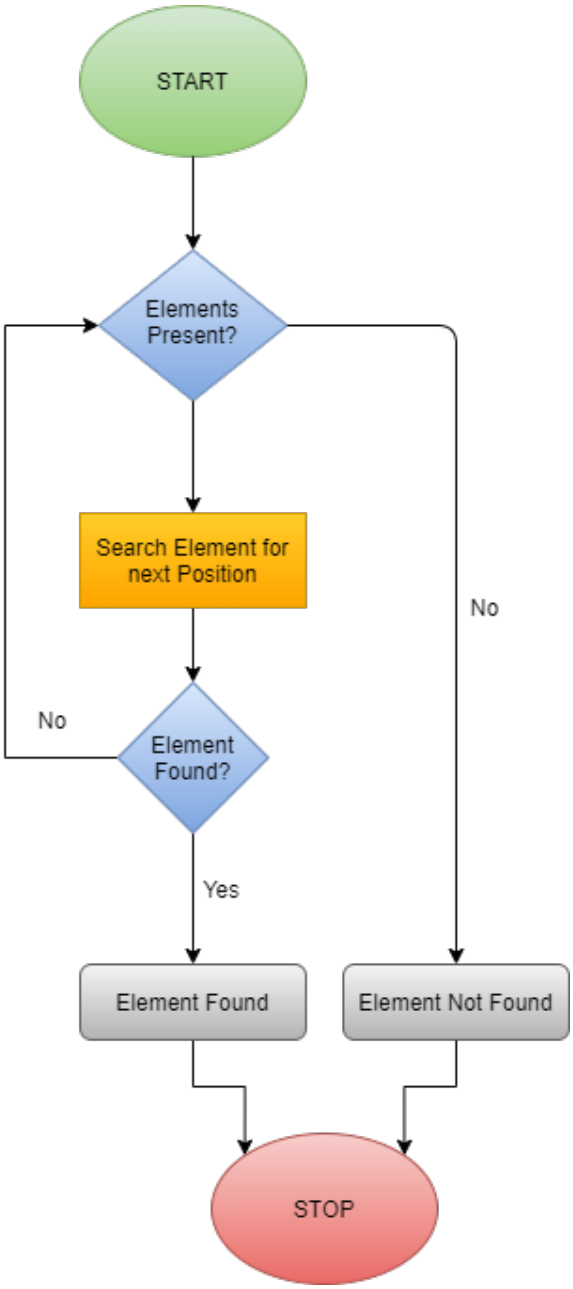
Green University of Bangladesh

1 Objective(s)

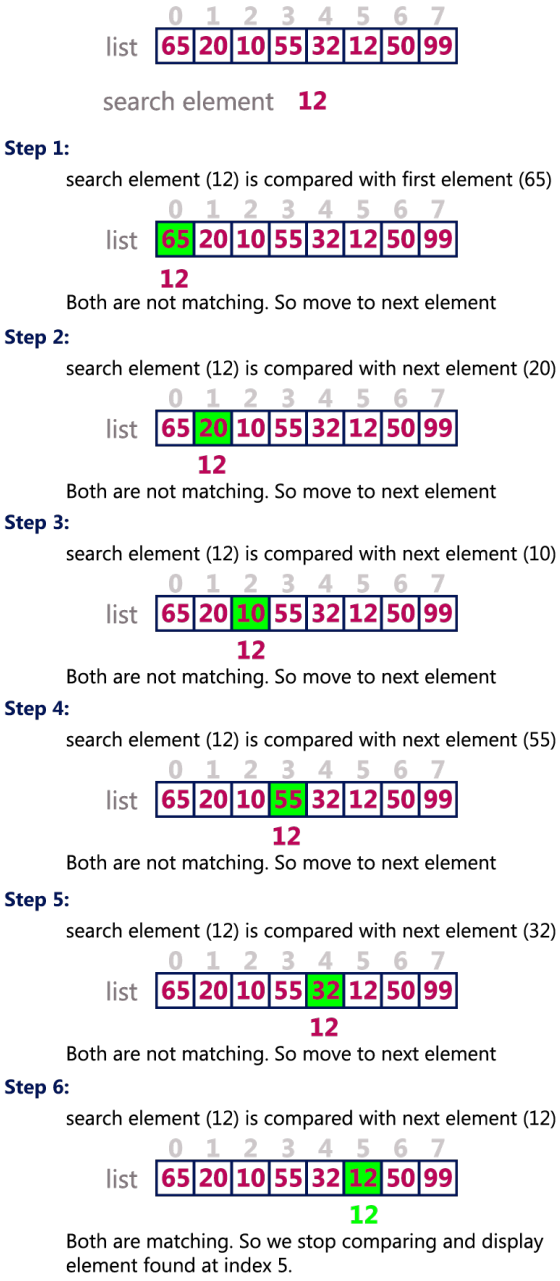
- To gather knowledge of different types of searching algorithms.
- To implement linear and binary search algorithms.

2 Problem analysis

Linear search is a very basic and simple search algorithm. In Linear search, we search an element or value in a given array by traversing the array from the starting, till the desired element or value is found. The time required to search an element using a linear search algorithm depends on the size of the array. In the best-case scenario, the element is present at the beginning of the list and in the worst-case, it is present at the end. The time complexity of a linear search is O (n).



(a) Flow chart



(b) Step by step example

Figure 1: Linear Search

3 Algorithm

Algorithm 1: Linear Search

Input: Array A, Value x

```
/* Linear Search (Array A, Value x)                                    */
```

1 Step 1: Set i to 1
2 Step 2: **if** $i > n$ **then**
3 | then go to step 7
4 **end**
5 Step 3: **if** $A[i] = x$ **then**
6 | then go to step 6
7 **end**
8 Step 4: Set i to i + 1
9 Step 5: Go to Step 2
10 Step 6: Print Element x Found at index i and go to step 8
11 Step 7: Print element not found
12 Step 8: Exit

4 Implementation in C

```c
/* Linear Search code */
#include<stdio.h>
int main()
{
  int list[20],size,i,sElement;

  printf("Enter size of the list: ");
  scanf("%d",&size);

  printf("Enter any %d integer values: ",size);
  for(i = 0; i < size; i++)
      scanf("%d",&list[i]);

  printf("Enter the element to be Search: ");
  scanf("%d",&sElement);

  // Linear Search Logic
  for(i = 0; i < size; i++)
  {
      if(sElement == list[i])
      {
          printf("Element is found at %d index", i);
          break;
      }
  }
  if(i == size)
      printf("Given element is not found in the list!!!");

  return 0;
}
```

5 Input/Output

Output of the program is given below.

```
Enter size of the list: 5
Enter any 5 integer values: 3 1 5 7 4
Enter the element to be Search: 7
Element is found at 3 index
```

6 Discussion & Conclusion

Based on the focused objective(s) to understand about linear search, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

7 Lab Task (Please implement yourself and show the output to the instructor)

1. Implement Binary Search Algorithm.

7.1 Problem analysis

Search a sorted array by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise, narrow it to the upper half. Repeatedly check until the value is found or the interval is empty. Binary search runs in logarithmic time in the worst case, making O (log n) comparisons, where n is the number of elements in the array. Binary search is faster than linear search except for small arrays. **However, the array must be sorted first to be able to apply binary search.**

7.2 Algorithm

Algorithm 2: Binary Search

Input: a, lower_bound, upper_bound, val

```
/* Binary_Search (a, lower_bound, upper_bound, val)                    */
1  step 1: set beg = lower_bound end = upper_bound, pos = - 1
2  step 2: repeat steps 3 and 4 while beg<=end
3  step 3: set mid = (beg + end)/2
4  step 4: if a [mid] = val then
5  |   set pos = mid
6  |   print pos
7  |   go to step 6
8  else
9  |   if a[mid] > val then
10 |   |   set end = mid - 1
11 |   else
12 |   |   set beg = mid + 1
13 |   end
14 end
15 step 5: if pos = -1 then
16 |   print "value is not present in the array"
17 end
18 step 6: exit
```

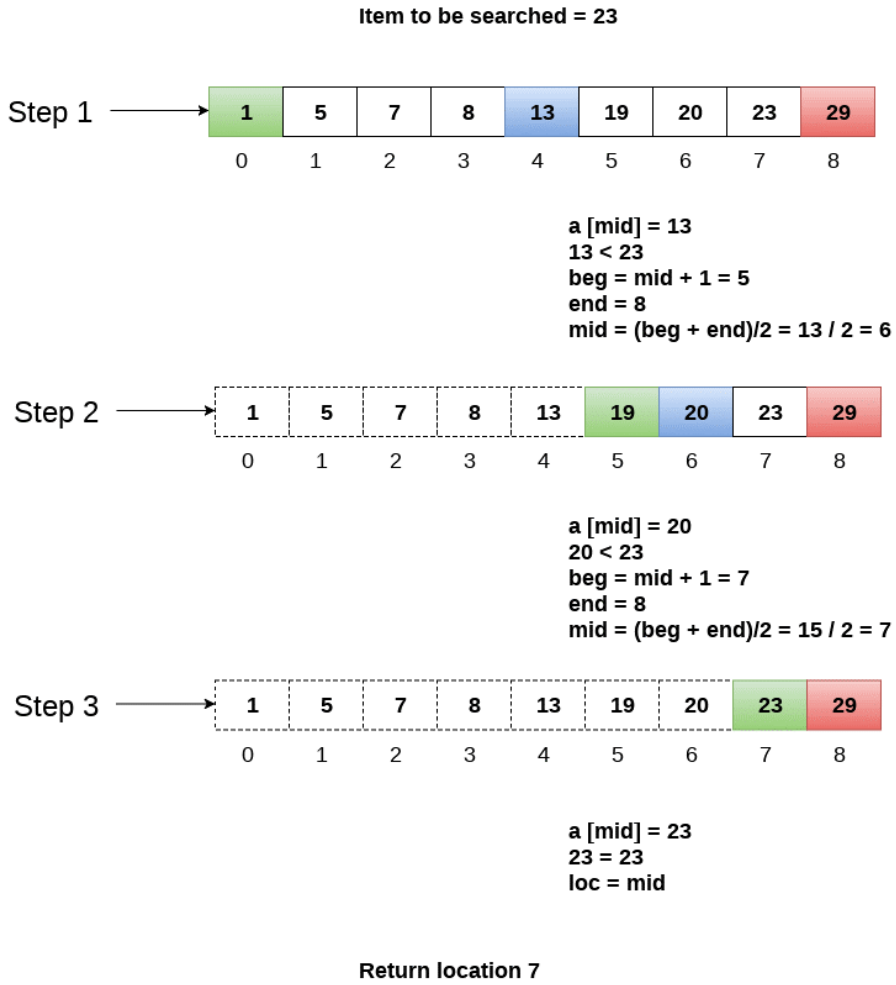


Figure 2: Step-by-step example of Binary Search

8 Lab Exercise (Submit as a report)

- Implement Linear Search for an array with character data.
- Implement Binary Search for an array with character data.

9 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.