CSE214 Algorithm Lab

Linear Search Implementation, Analysis, Worst case, Best case, Average Case and Bubble Sort Visualization.

Submitted to

Mr. Subroto Nag Pinku (SNP)

Submitted By

SIUM BIN NOOR

191-15-12516

Section: B/O14

Linear search

Implementation:

```
int Linear_search (int arr[], int n, int x)
{
    int i, index = -1;
    for (i=0; i<n; i++){
        if (arr[i]==x){
            index = i;
            break;
        }
    }
    return index;
}
```

Analysis

3	5	1	2	4

Let, This array have 5 elements. So $n=5$.

We want to search $x=2$.

At the early stage of iteration when $i=0$, which is less than n, the loop starts. After cheaking the value on array if it is found the loop is stop and return index because of break function. So, our searching value 2 is on the 3 no. index on array because the ~~ind~~ array index starts with 0. So, 0, 1, 2, 3 searching we found our value ~~20f~~ on index no. 3. Then the loop won't work because of break function. So, the loop execute only 4 times.

worst case

If the array has n elements and the value is not in the array or it is in the last position n-1, the the loops run for n times.

So, the complexity would be $O(n)$.

Best case

2	3	1	5	4

If $x = 2$, which is in the 1st index of array, the loop will run for 1 time.

So, the best case of complexity is $O(1)$.

Average case

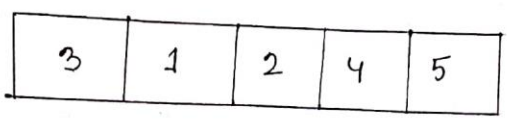
$$\text{Average case} = \frac{\text{All possible case time}}{\text{Numbers of cases}}$$

$$= \frac{1+2+3+\cdots+n}{n}$$

$$= \frac{\frac{n(n+1)}{2}}{n} = \frac{n+1}{2}$$

Ignoring the constant co-efficient, the complexity of Average case is $O(n)$.

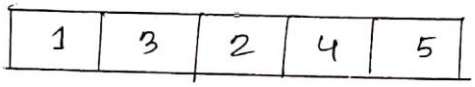
Bubble Sort Visualization



If we bubble sort this array, the visualization should be,

The loop check 1st array index and 2nd array index. If 1st > 2nd than it swap. In case,

3 > 1, so, swap.



Then again. 3 > 2, so. swap.

Then, loop again check if 3 is greater than 4 or not. In case, 3 is not greater than 4. so, loops check 4 is greater than next index array or not. 4 is not greater than 5. so, loops stops checking.

The ~~First~~ Final result of this array of bubble sort is,