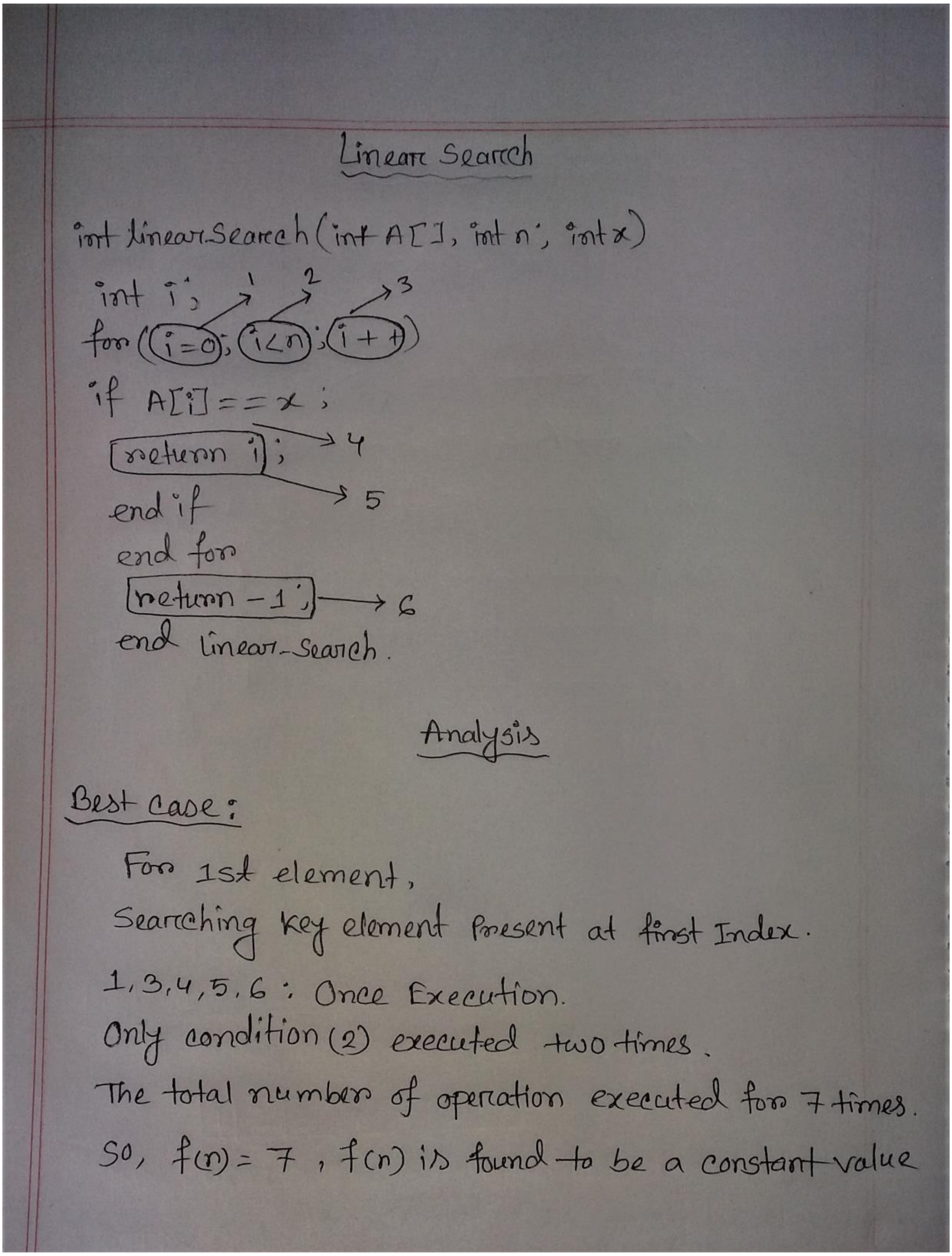
Name : Abruna Afrin

ID:191-15-12303

Sec:B



Linear Search

```
int linearSearch(int A[], int n; int x)
 int i;          1       2       3
 for (i=0; i<n; i++)
 if A[i]==x;  → 4
  return i;   → 5
  end if
  end for
   return -1; → 6
  end linear-Search.
```

Analysis

Best case:

For 1st element,

Searching key element Present at first Index.

1, 3, 4, 5, 6 : Once Execution.

Only condition (2) executed two times.

The total number of operation executed for 7 times.

So, $f(n) = 7$, $f(n)$ is found to be a constant value

If the running time of time complexity of Algorithm is a constant, then the time Complexity will be O(1).

Worst case:

Searching key element Present at last Index.

1,5,6 : executed one time

2,3,4 : executed for > n times.

The complexity function of the Algorithm is:

$$f(n) = 3n + 3$$

∴ Here, coefficient is ignorable from the complexity function.

∴ The time complexity is: O(n)

Average Case:

Searching key element Present at middle Index.

$$\text{Avg case} = \frac{\text{all possible case time}}{\text{no. of cases}}$$

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

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

The time complexity of the average case is : O(n)

Bubble Sort

```
Void bubble_sort( int A[], int n)
int i, j, temp;
for( i=0; i<n-1; i++) ⟶ n
  for (j=0; j<n-i-1; j++) → n×n
   if( A[j] > A[j+1]) ⟶ n ×n
    temp = A[j]; ⟶ n×n
    A[j] = A[j+1] ⟶ n×n
    A[j+1] = temp ⟶ n×n
  end if
  end for
  end for
  end bubble_sort
                                   f(n) = 5n^2 + n
```

Analysis

Best Case:

Assume, Here is a Sorted Array.

1	2	3	4	5	6	7

This array will Cheak of the last element by both loops. But, Any element of the array can't swap

~~bae~~ So, Always n numbers of the element of an array will cheak for n times both loop.

∴ The time complexity of best case time is : $O(n^2)$

Worst Case:

from the code,

we get,

the complexity of ~~of o~~ function of an algorithm is: $f(n) = 5n^2+n$.

∴ The time complexity of worst case time is: O(n

Average Case:

Both loop ~~will~~ is executed for n times. All Possible values are average case time.

Avg case = All Possible values.

$$\text{Avg time} = 1+2+3+\cdots+(n-2)+(n-1)+n$$

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

∴ The time complexity of ~~Avg~~ Average case time is: $O(n^2)$.

Bubble Sort Example

Increasing Order: 5
Enter the numbers to be sorted:
10 5 2 8 7

Here,

```
1st: 5  10  2  8  7        10>5 Swapping
     ↑   ↑

     5  2  10  8  7        10>2 swapping
        ↑   ↑

     5  2  8  10  7        10>8 swapping
           ↑   ↑
                           10>7 swapping
     5  2  8  7  10
              ↑   ↑
2nd:                       5>2 Swapping
     2  5  8  7  10
     ↑  ↑
                           5<7 noswapping
     2  5  7  8  10
        ↑  ↑               7<8 no swapping
     2  5  7  8  10
           ↑  ↑
3rd:                       2<5 no swapping
     2  5  7  8  10
     ↑  ↑                  5<7 no swapping
     2  5  7  8  10
        ↑  ↑
```

4th:

2 5 7 8 10 2<5 no swapping

↑ ↑

The Result after bubble sorting is:

2 5 7 8 10