Today our topic of discussion is **SEARCHING**.

Let us define the term SEARCHING.

Searching is a method used to find the location of a particular item (called key) in a list.

Consider **given list is 12, 67, 4, 78, 23, 90** and **key is 78**.

So we need a **method** to firstly **check** whether the key **78** is present in the list or not. Then the **method** should provide us the location of the key **78** in the list.

We say search is

SUCESSFUL **if** key is found in the list

otherwise UNSUCCESSFUL.

In this example, search is **SUCCESSFUL**

and **78 is the 4th item** in the list.

We are going to discuss following two searching methods –

1. Linear Search (Sequential Search).
2. Binary Search.

We will **assume** that **the given list** will be **stored** in an **array**. That is, we have to **search an array** to find the location of key (i.e. a given item).

Basic Idea of Linear Search:

- ✓ Array is usually unordered (not sorted)
- ✓ Start at the beginning of the array.
- ✓ Inspect elements one by one to see if it matches the key.

Examples to Illustrate the Linear Search method

Example – 1:

List: 12, 67, 4, 78, 23, 90 and key: 78

1. The list is stored in an array "**a**" as shown below –

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
12	67	4	78	23	90

2. At first, element a[0] and key, that is,12 and 78 are compared. Element 12 does not match the key.
3. Then a[1] and key, that is, 67 and 78 are compared. Element 67 does not match the key.
4. Next, 4 and 78 are compared. Element 4 does not match the key.
5. Element 78 and 78 are compared. This time element a[3] matches the key. So we can say that key is present in the array at index 3.
6. Hence the method will return a value 3 to mean that the key appears in the array at index 3.

Example – 2:

List: 12, 67, 4, 78, 23, 90 and key: 5

1. The list is stored in an array "**a**" as shown below –

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
12	67	4	78	23	90

2. At first, element a[0] and key, that is,12 and 5 are compared. Element 12 does not match the key.
3. Then a[1] and key, that is, 67 and 5 are compared. Element 67 does not match the key.
4. 4 and 5 are compared. Element 4 does not match the key.
5. Likewise each remaining element is compared with 5. But, no element matches the key. That is key is NOT found. In other words search is Unsuccessful.
6. Hence the method will return a value **-1** to mean that the key does NOT appear in the array.

   [Note: -1 is used to denote that the key does not appear in the array because lowest value of index is zero where key may appear.]

Implementation Plan (How coding is planned):

In main function:

1. We will accept the value of **n** (the number of elements) from the user.
2. Then we need to accept all the **n** elements of the array from the user.
3. Now we take the key (i.e. item to be searched) from user.
4. Then the linear_search function will be called with 3 arguments. First argument is the array itself, next is the size of array and last is the key.
5. After getting the "**loc**" value returned by the linear_search function, we need to check its value. If it is -1, "key does

not exist in the array" will be printed otherwise "Key found at array index loc".

In linear_search function:

1. We have to perform two tasks –
   a) Comparison
   b) Return a value (either -1 or index value).
2. One or more elements need to be compared with key. Hence, comparison task is to be executed inside a loop. When there is a match, immediately index value is returned to main(). When there is no element to compare, -1 is returned to main().

Now YOU HAVE TO GO THROUGH the **Second_Note_on_Searching_CSE_A_20**.
