First application of an array is Searching under which we will study different searching approaches

Searching → Linear Search & Binary Search

arr = [20, 40, 70, 10, 12, 11, 29, 75, 46]
0 1 2 3 4 5 6 7 8

n = 9

x = 47

Linear Search

x = 11

n = len(arr)

```
for i in range(n):        # 0 to n-1
    if arr[i] == x:
        return i
return -1
```

↳ Element is not present in an array

Time complexity

↳ Worst case Scenario :- O(n)

when element to be searched is present in the last index

↙ Element is present almost at the last index.

↗ Element is present at initial indexes.

Best case Scenario :- O(1)

Average case Scenario :-

$$\frac{1+2+3+4+\_\_\_+n}{n}$$

sum of n natural numbers

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

Space complexity → O(1)

Since we are not using any extra space that's why Space complexity is O(1).

Note that Temp variable are not considered as extra space. But if we are introducing a completely new list, stack, queue(Abstract data types) in our problem statement then it will definitely be counted under space complexity.

O(1) means constant space or no change in the space required during the runtime of the program.