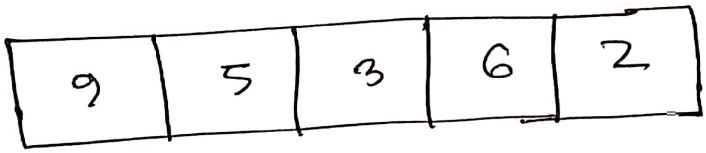
Linear Search

```
int 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



Here the array have 5 elements. So here $n = 5$

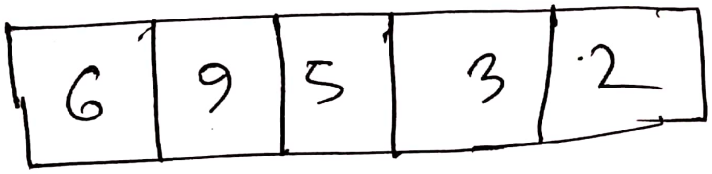
Now we want to search $x = 6$.

And now when $i = 0$ and 0 is less then n because $n = 5$ now loop starts. If it is found the loop will be stoped and return the index number because we I use break function. my search value is 6 and I found it on the index number 3.

Worst Case

If the array has n elements the value is not in the array or available at the last of the array ~~posith~~ position n-1, the loop run for n times, so here the complexity would be O(n)

Best Case



If n = 6, and which number is available at the 1st index in the array the loop will run for 1 times then break

So the best case of complexity is O(1)

Average case

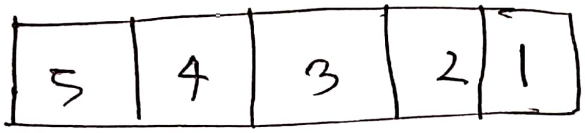
$$\text{Average Case} = \frac{\text{All possible case time}}{\text{Number of cases}}$$

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

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

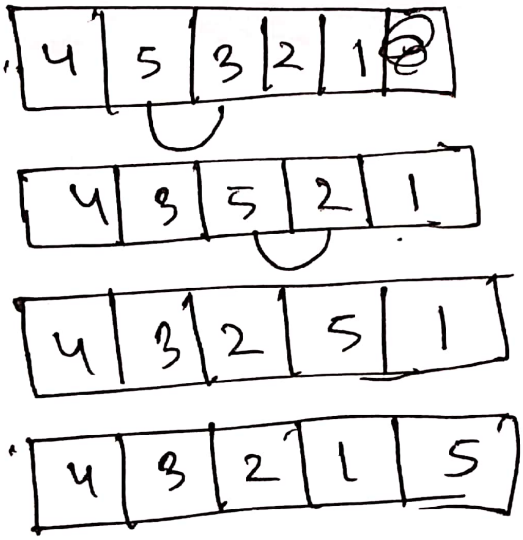
Here we ignoring the constant value so the complexity of Average is $O(n)$.

Bubble Sort Visualization



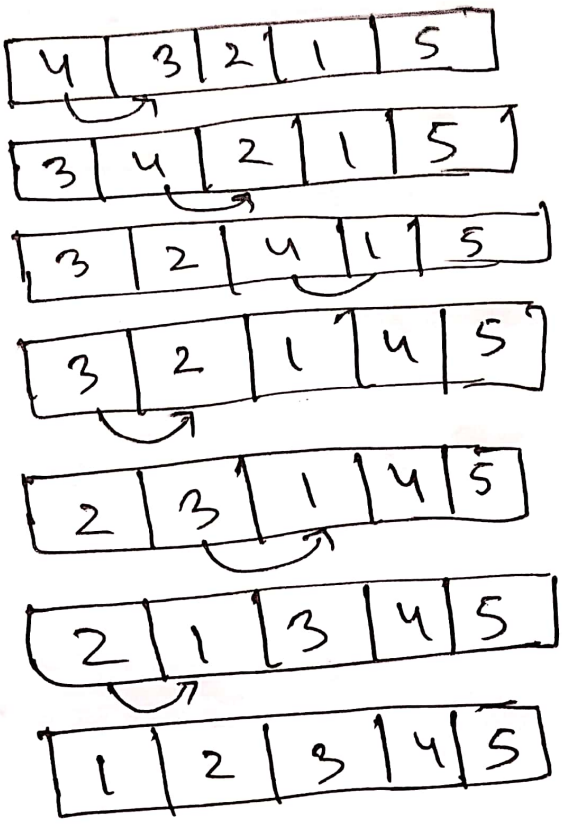
~~we should~~

First we should check 1st index and second index here $5 > 4$ so we use swap then ~~execute~~ it looks like



Here the biggest number is now at the last index. Here ~~condition~~ condition should be ~~arr[i] > arr~~ arr[i] > arr[i+1] then swap

Again



Here the look stop because it check $3 < 4$ so the loop will be stop.

So the final result of bubble sort is