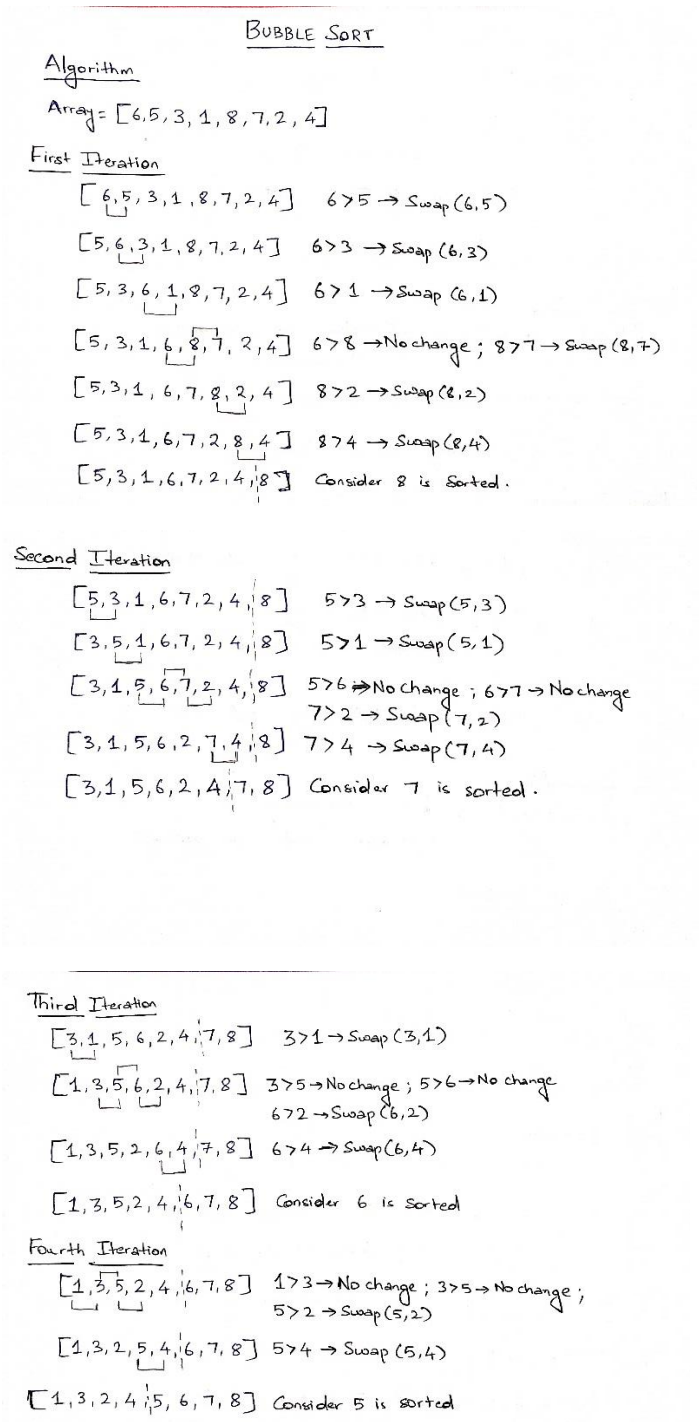
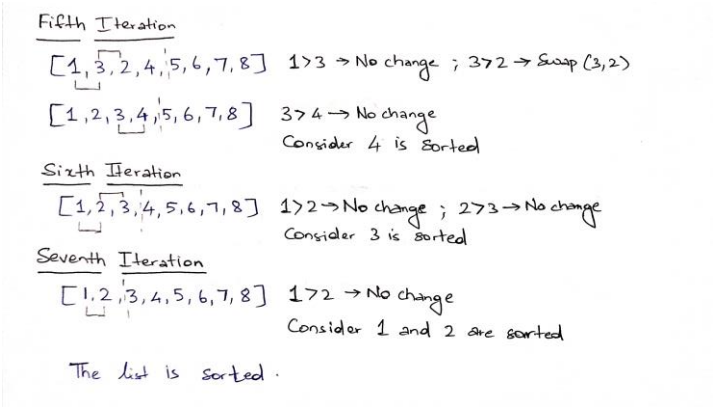
Bubble Sort

In Bubble sort, each element is compared with its adjacent element. Incase of ascending order if the first element is greater than the second one, then the positions of the elements are interchanged, otherwise it is not changed.

Then next element is compared with its adjacent element and the same process is repeated for all the elements in the array until we get a sorted array.

For better understanding refer the below images, which illustrates the algorithm of Bubble Sort.





Code

```
def Bubble_Sort(arr,detailed_steps=0):
    for i in range(len(arr)-1):
        for j in range(len(arr)-1-i):
            if arr[j]>arr[j+1]:
                temp=arr[j]
                arr[j]=arr[j+1]
                arr[j+1]=temp
            if detailed_steps: #To check the array after every iteration
                print("\033[1m" + f'When {i=} and {j=}\n{arr=}\n')
    print("\033[1m" + f'The final sorted Array is {arr=}') #"\033[1m" to print in bold
```

```
arr=[6,5,3,1,8,7,2,4]
```

```
Bubble_Sort(arr)
```

The final sorted Array is arr=[1, 2, 3, 4, 5, 6, 7, 8]

```
print("\033[1m" + 'Output after each iteration\n')
Bubble_Sort(arr,1) #Passing 1 to get the output after every step
```

Output after each iteration

When i=0 and j=0
arr=[5, 6, 3, 1, 8, 7, 2, 4]

When i=0 and j=1
arr=[5, 3, 6, 1, 8, 7, 2, 4]

When i=0 and j=2
arr=[5, 3, 1, 6, 8, 7, 2, 4]

When i=0 and j=3
arr=[5, 3, 1, 6, 8, 7, 2, 4]

When i=0 and j=4
arr=[5, 3, 1, 6, 7, 8, 2, 4]

When i=0 and j=5
arr=[5, 3, 1, 6, 7, 2, 8, 4]

When i=0 and j=6
arr=[5, 3, 1, 6, 7, 2, 4, 8]

When i=1 and j=0
arr=[3, 5, 1, 6, 7, 2, 4, 8]

When i=1 and j=1
arr=[3, 1, 5, 6, 7, 2, 4, 8]

```
When i=1 and j=2
arr=[3, 1, 5, 6, 7, 2, 4, 8]

When i=1 and j=3
arr=[3, 1, 5, 6, 7, 2, 4, 8]

When i=1 and j=4
arr=[3, 1, 5, 6, 2, 7, 4, 8]

When i=1 and j=5
arr=[3, 1, 5, 6, 2, 4, 7, 8]

When i=2 and j=0
arr=[1, 3, 5, 6, 2, 4, 7, 8]

When i=2 and j=1
arr=[1, 3, 5, 6, 2, 4, 7, 8]

When i=2 and j=2
arr=[1, 3, 5, 6, 2, 4, 7, 8]

When i=2 and j=3
arr=[1, 3, 5, 2, 6, 4, 7, 8]

When i=2 and j=4
arr=[1, 3, 5, 2, 4, 6, 7, 8]

When i=3 and j=0
arr=[1, 3, 5, 2, 4, 6, 7, 8]

When i=3 and j=1
arr=[1, 3, 5, 2, 4, 6, 7, 8]
```

```
When i=3 and j=2
arr=[1, 3, 2, 5, 4, 6, 7, 8]

When i=3 and j=3
arr=[1, 3, 2, 4, 5, 6, 7, 8]

When i=4 and j=0
arr=[1, 3, 2, 4, 5, 6, 7, 8]

When i=4 and j=1
arr=[1, 2, 3, 4, 5, 6, 7, 8]

When i=4 and j=2
arr=[1, 2, 3, 4, 5, 6, 7, 8]

When i=5 and j=0
arr=[1, 2, 3, 4, 5, 6, 7, 8]

When i=5 and j=1
arr=[1, 2, 3, 4, 5, 6, 7, 8]

When i=6 and j=0
arr=[1, 2, 3, 4, 5, 6, 7, 8]

The final sorted Array is arr=[1, 2, 3, 4, 5, 6, 7, 8]
```

Time complexity : Since nested for loops are involved, the average time complexity for Bubble Sort is O(N^2).

Advantage

- When data set is small, bubble sort is efficient
- Easy to implement

Disadvantage

- It is time-inefficient as it is having O(N2) time complexity.
- For large data set it is not very efficient as time grows exponentially.