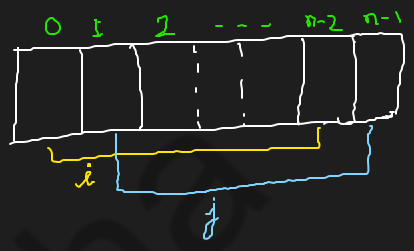
Sorting (8.3)

1. **Selection Sort:**

Idea: The inner loop selects the minimum element in the unsorted array and places the elements in increasing order.

Time complexity: $O(N^2)$

```
for (int i = 0; i < n - 1; i++) {
    for (int j = i + 1; j < n; j++) {
        if (arr[j] < arr[i]) {
            int temp = arr[j];
            arr[j] = arr[i];
            arr[i] = temp;
        }
    }
}
```

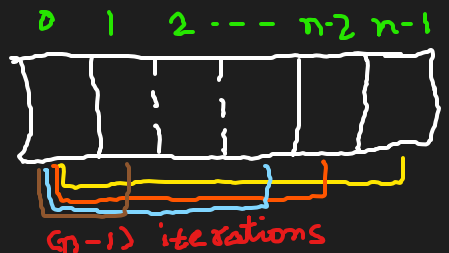


2. **Bubble Sort:**

Idea: if arr[i] > arr[i+1] swap them. To place the element in their respective position, we have to do the following operation N-1 times.

Time Complexity: $O(N^2)$

```
int counter = 0;
while (counter < n - 1) {
    for (int i = 0; i < n - counter - 1; i++) {
        if (arr[i] > arr[i + 1]) {
            int temp = arr[i];
            arr[i] = arr[i + 1];
            arr[i + 1] = temp;
        }
    }
    counter++;
}
```



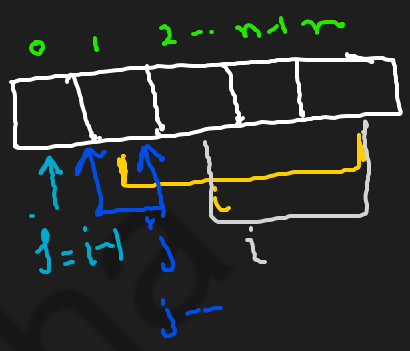
Note → It is called bubble sort as the maximum element rises up just like a bubble.

3. **Insertion Sort:**

Idea: Take an element from the unsorted array, place it in its corresponding position in the sorted part, and shift the elements accordingly.

Time Complexity: $O(N^2)$

```
for (int i = 1; i < n; i++) {
    int current = arr[i];
    int j = i - 1;
    while (arr[j] > current && j >= 0) {
        arr[j + 1] = arr[j];
        j--;
    }
    arr[j + 1] = current;
}
```



Imp Idea → Shifting array elements arr[j+1] = arr[j].

Homework: Implement selection sort, bubble sort, insertion sort on your own.