SORTING

1. Insertion Sort
2. Selection Sort
3. Bubble Sort
4. Quick Sort
5. Merge Sort

1. Insertion Sort

Best Case – O(n)

Average Case – O(n^2)

Worst Case – O(n^2)

```cpp
#include<iostream>
using namespace std;
int main (){
    int n=7;
    int arr[n]={10,1,6,2,8,5,11};

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

2. Selection Sort

Best Case – O(n^2)

Average Case – O(n^2)

Worst Case – O(n^2)

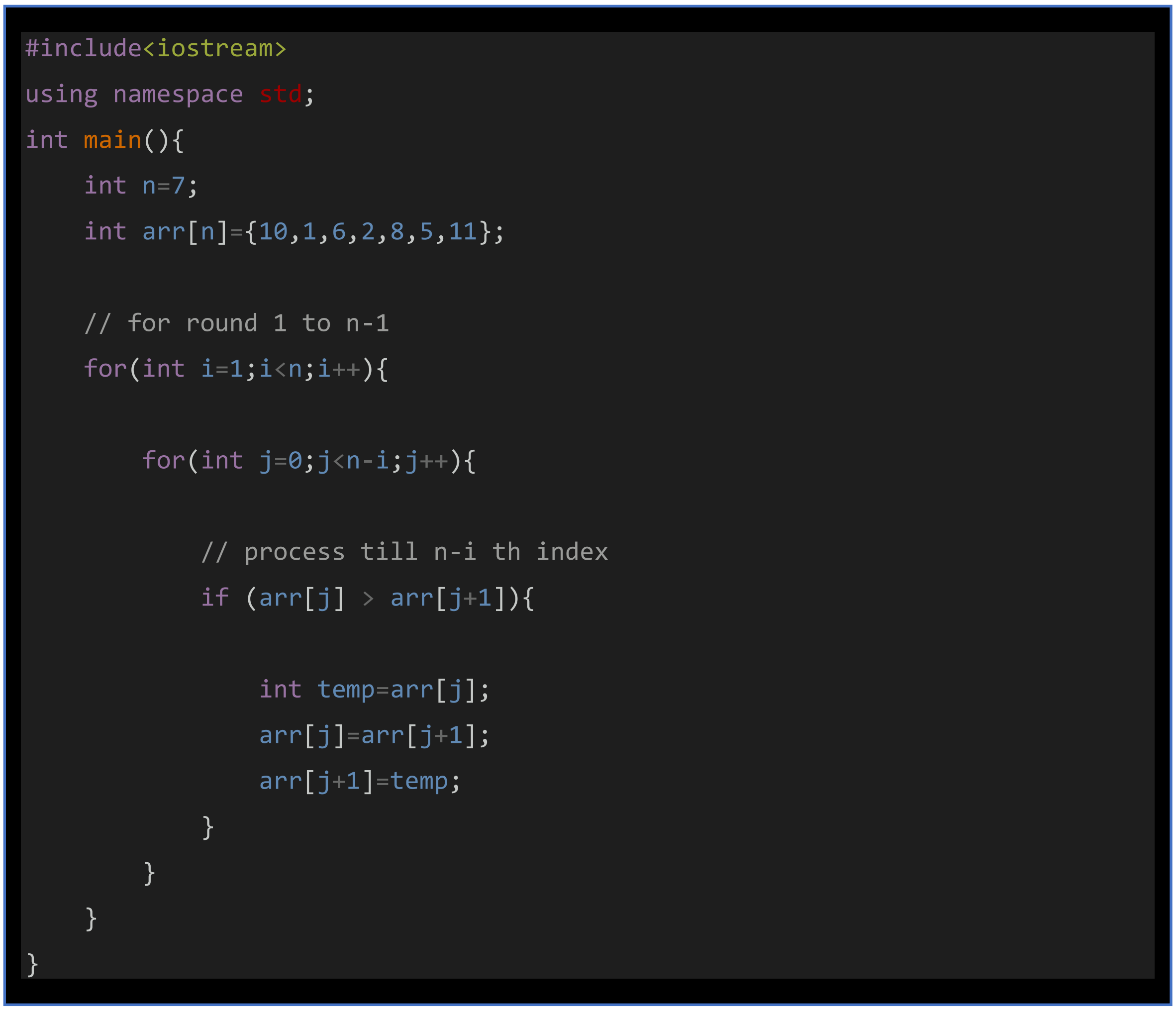
```
#include<iostream>
using namespace std;
int main(){
    int n=7;
    int arr[n]={10,1,6,2,8,5,11};

    for(int i=0;i<n-1;i++){
        int minIndex=i;

        // finding minimum element
        for(int j=i+1;j<n;j++){
            if(arr[j]<arr[minIndex]){
                minIndex=j;
            }
        }
        // swap
        int temp=arr[i];
        arr[i]=arr[minIndex];
        arr[minIndex]=temp;
    }
}
```

3. Bubble Sort

Best Case	– O(n)
Average Case	– $O(n^2)$
Worst Case	– $O(n^2)$



```
#include<iostream>
using namespace std;
int main(){
    int n=7;
    int arr[n]={10,1,6,2,8,5,11};

    // for round 1 to n-1
    for(int i=1;i<n;i++){

        for(int j=0;j<n-i;j++){

            // process till n-i th index
            if (arr[j] > arr[j+1]){

                int temp=arr[j];
                arr[j]=arr[j+1];
                arr[j+1]=temp;
            }
        }
    }
}
```

4. Quick Sort

Best Case	– O(n log n)
Average Case	– O(n log n)
Worst Case	– O(n^2)