Day 21

12 December 2022 22:00

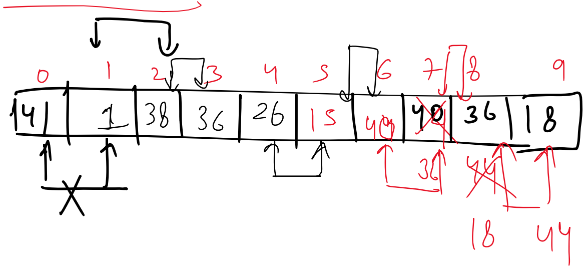
① Bubble sort
② Selection Sort
③ Insertion sort

① Bubble sort

① greatest last में भेजने कि कोशिश करते है।।

① Two adjacent array elements का compare करना

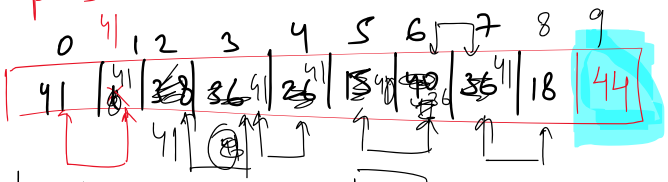
if smaller index पर greater element होगा with respect to its adjacent swap



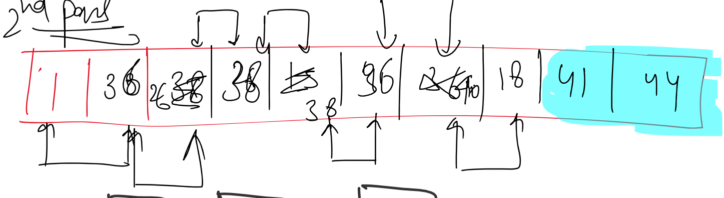
1 pass

41	1	38	36	26	15	40	36	18	44

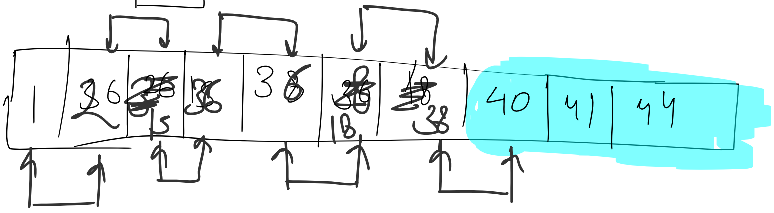
2 pass



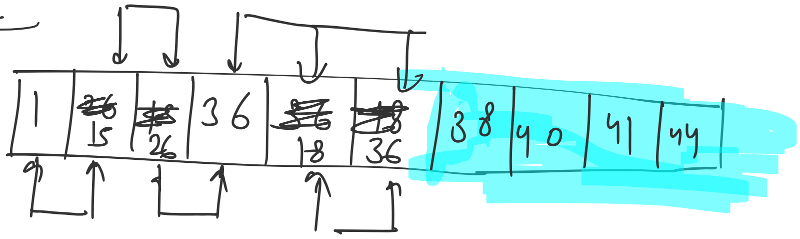
After 2nd pass



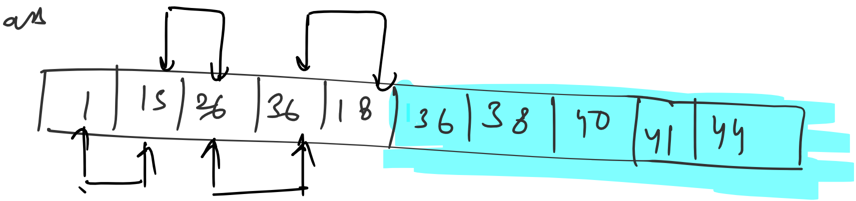
3 pass



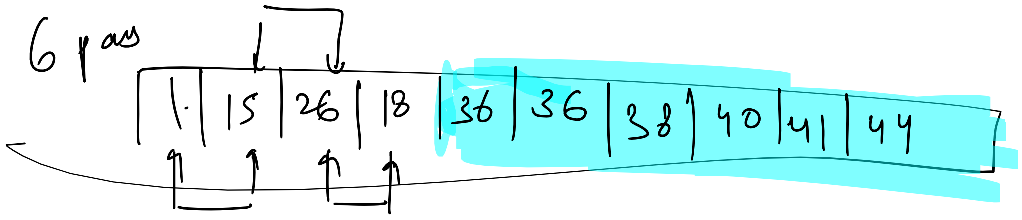
4 pass



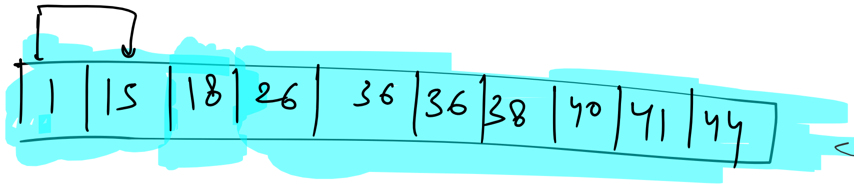
5 pass

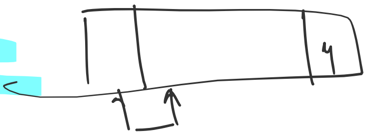


6 pass

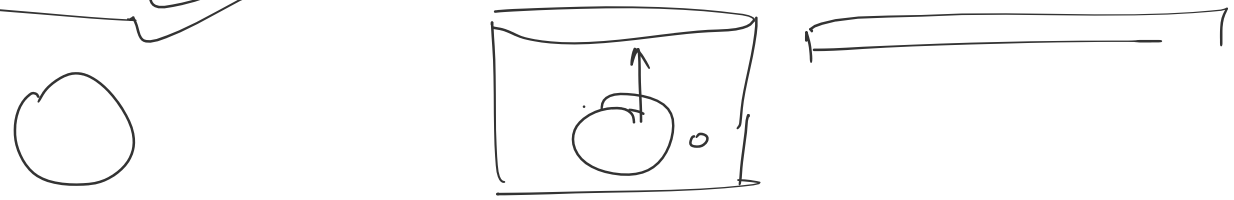


7 pass





n-1 (9) pass



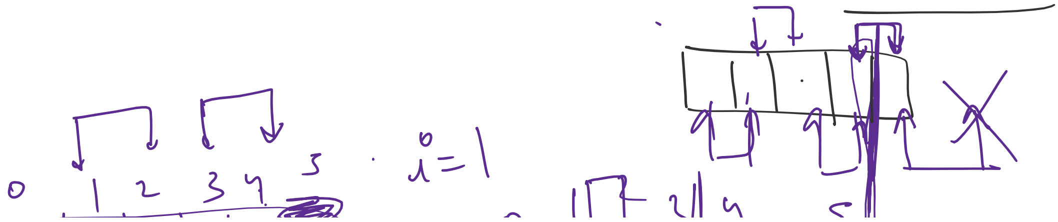
Code

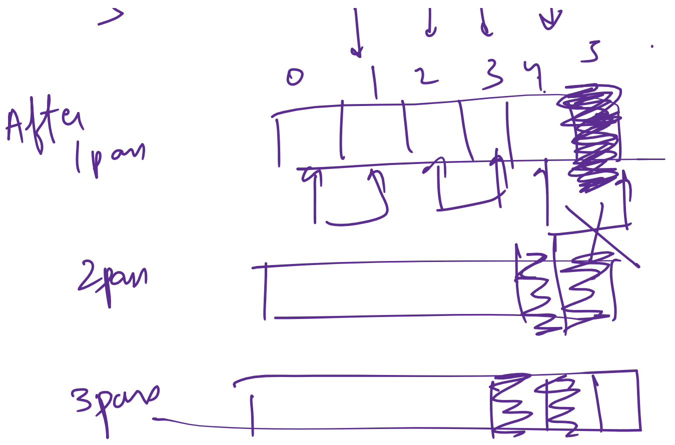
```
for(int i = 0; i < n-1; i++) {
    for(int j = 0; j < n-1-i; j++){
        if(A[j] > A[j+1]) {
            swap(A[j], A[j+1]);
        }
    }
}
```

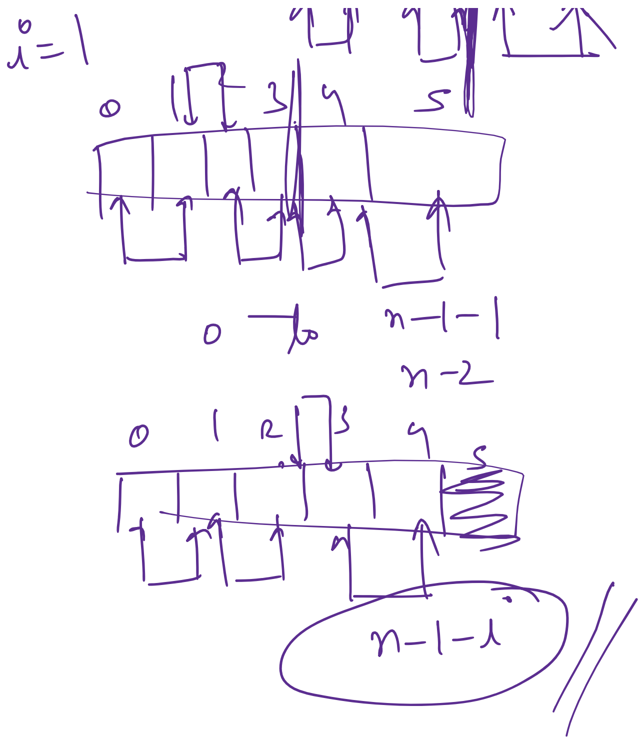
① Greater वह वाले unsorted part of array find करता रहता।

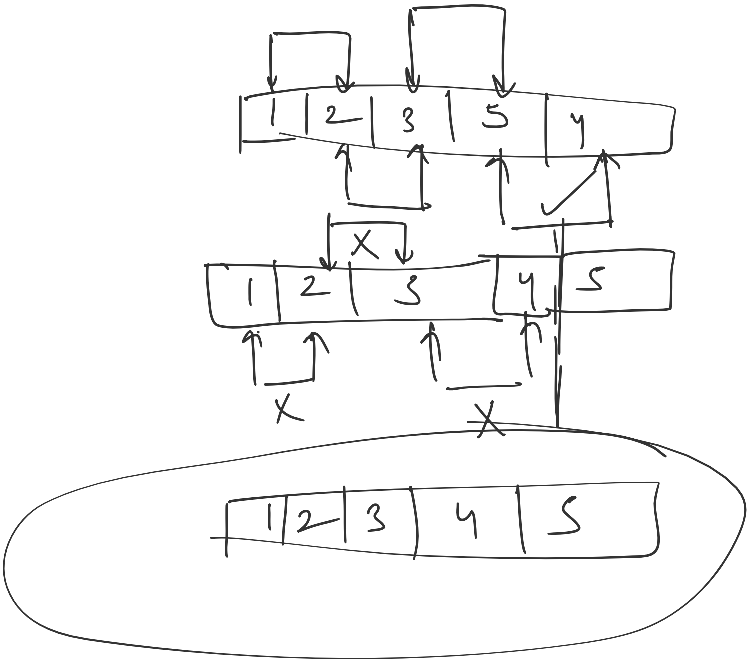
② Adjacent elements को compare करता रहता ||

③ smaller index पर greater value with respect to its adjacent swap.

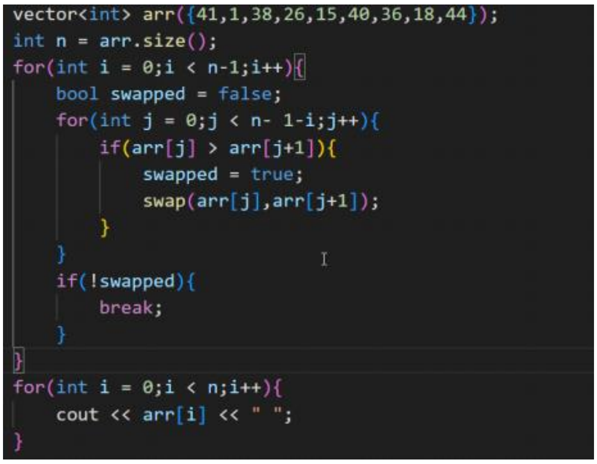








Conclusion: If in a pass no swapping has occurred.
I would stop the algo. because
my array is is sorted.

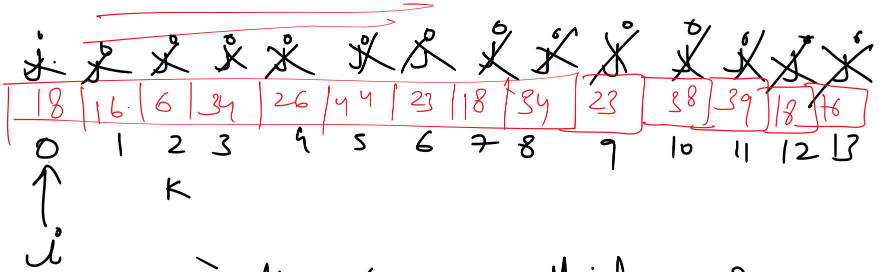


```
vector<int> arr({41,1,38,26,15,40,36,18,44});
int n = arr.size();
for(int i = 0;i < n-1;i++){
    bool swapped = false;
    for(int j = 0;j < n- 1-i;j++){
        if(arr[j] > arr[j+1]){
            swapped = true;
            swap(arr[j],arr[j+1]);
        }
    }
    if(!swapped){
        break;
    }
}
for(int i = 0;i < n;i++){
    cout << arr[i] << " ";
}
```

T.C — O(N^2)

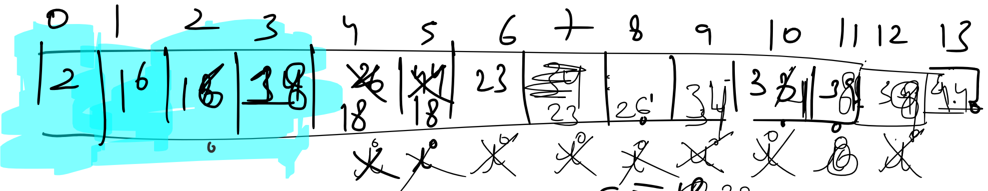
S.C — O(1)

Selection Sort



j → i to n

smaller = ~~18~~ ~~16~~ 6 smallerindex = 2



S = ~~18~~ 23

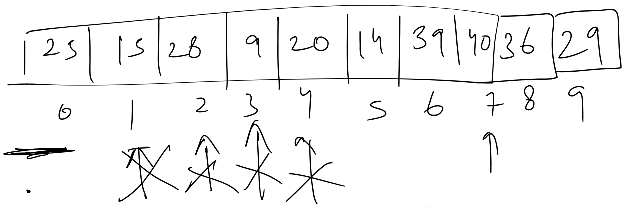
si = ~~13~~ 6

Sorted

i = 0 to n-1

j = i to n smallerindex = K

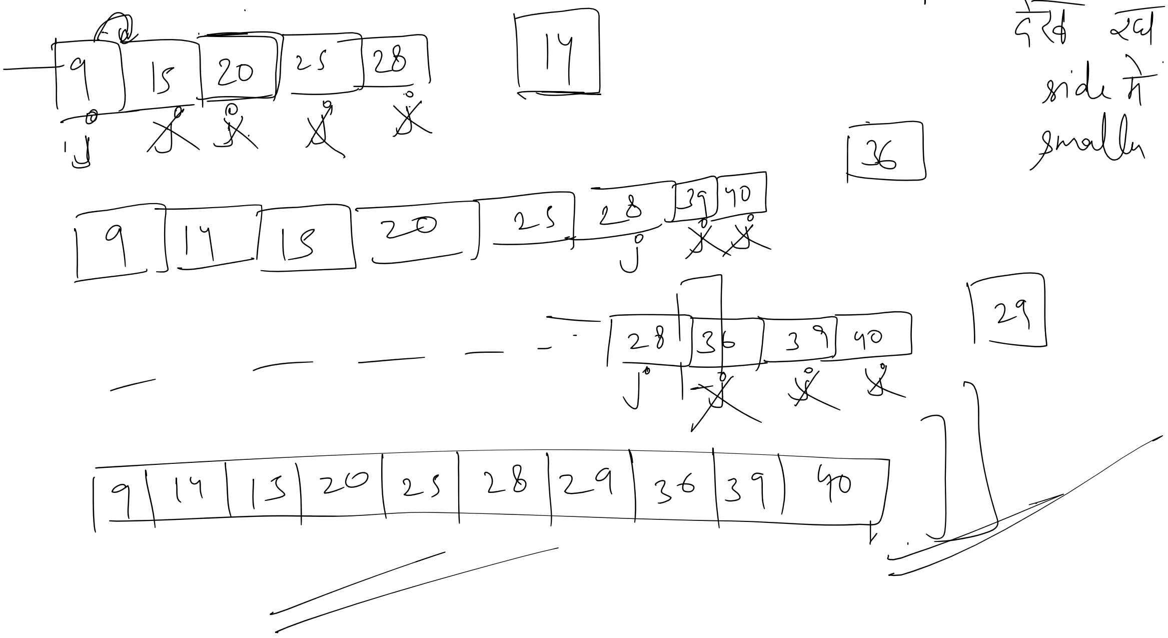
swap
i <=> K

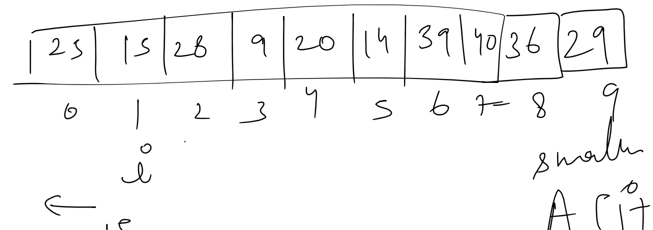


★ left से right array traversal करेगा और arrange हर एक element को उसकी correct position पर (left side में)

★ हर case में अगर मे element को left

★ ऐसा case मे अगर मे element देख रहा हु तो left side मे जब तक मुझे उससे smaller value नही मिलती

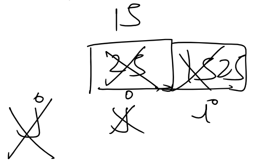




smaller मिल या रहा है||

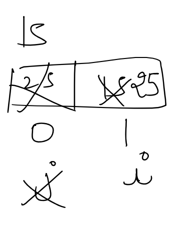
$A[j+1] = \text{value}$

greater mil रहा है||



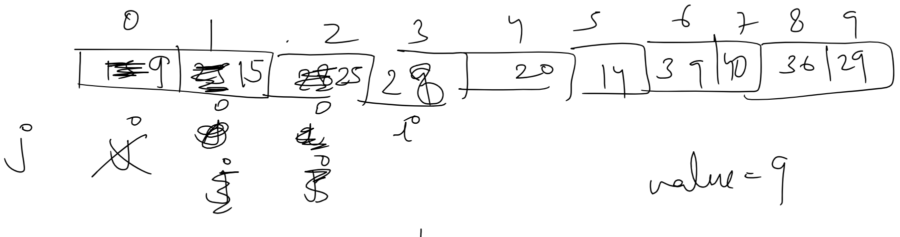
$A[j+1] = A[j]$

1 pass

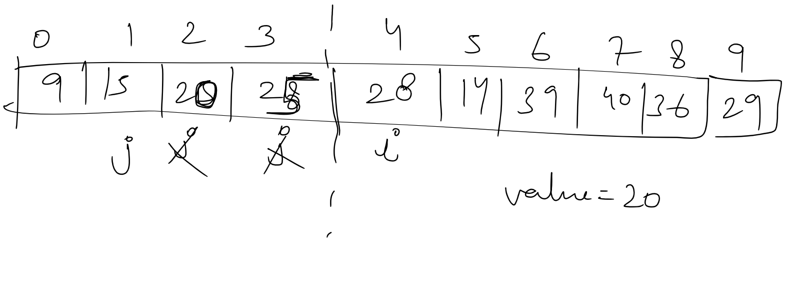


$A[j-1+1] = \text{value}$

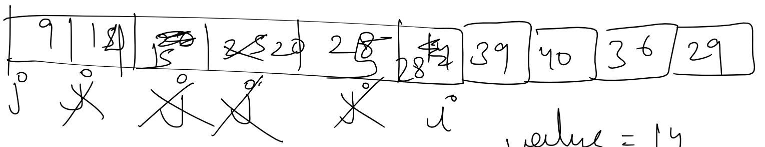
$A[j+1] = A[j]$



value=9



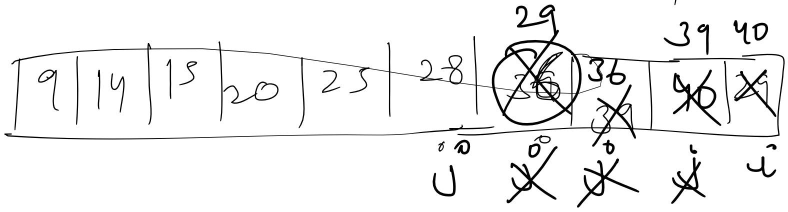
value=20



greater

$A[j+1] = A[j]$

value = 14



$A[j+1] = value$

value = 29

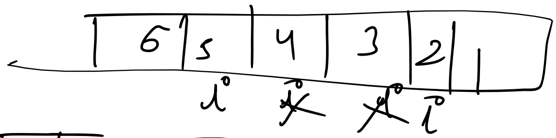
9	14	15	20	25	28	29	36	39	40

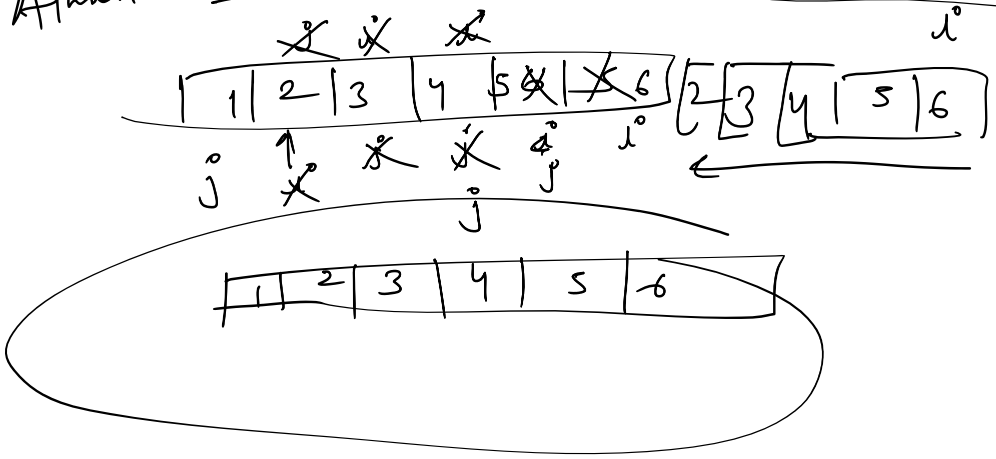
Best case

Worst case

O(N)

Almost sorted

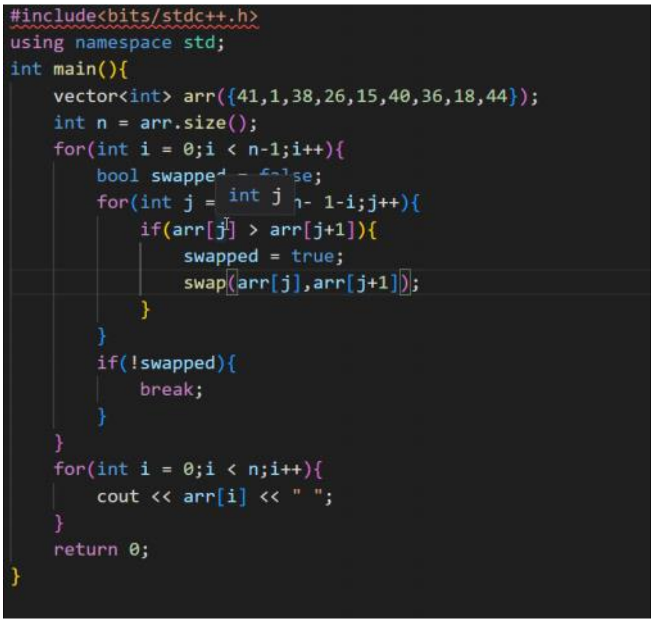




$O(N^2)$

T.C — $O(N^2)$

S.C — $O(1)$

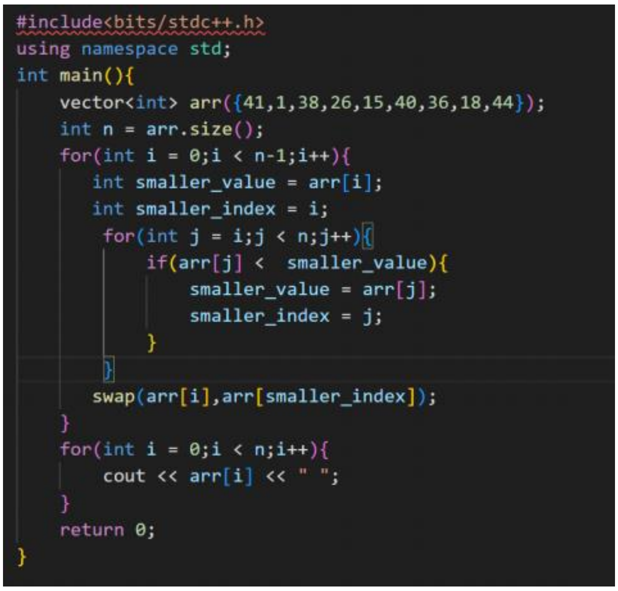


```
#include<bits/stdc++.h>
using namespace std;
int main(){
    vector<int> arr({41,1,38,26,15,40,36,18,44});
    int n = arr.size();
    for(int i = 0;i < n-1;i++){
        bool swappe     se;
        for(int j =       n- 1-i;j++){
            if(arr[j] > arr[j+1]){
                swapped = true;
                swap(arr[j],arr[j+1]);
            }
        }
        if(!swapped){
            break;
        }
    }
    for(int i = 0;i < n;i++){
        cout << arr[i] << " ";
    }
    return 0;
}
```

Bubble sort

T.C – $O(N^2)$

S.C – $O(1)$

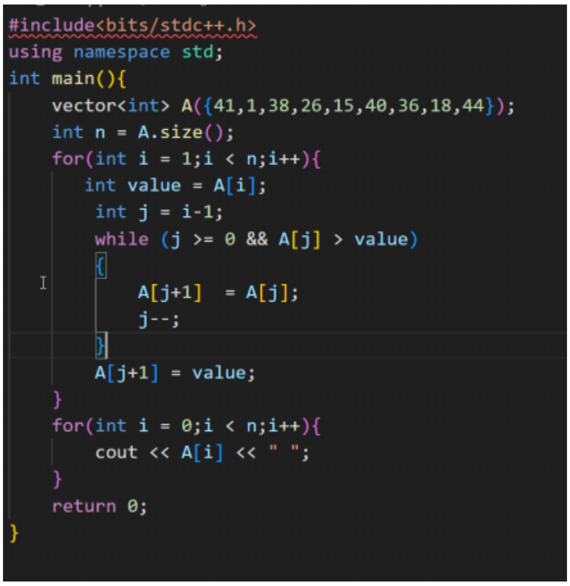


```
#include<bits/stdc++.h>
using namespace std;
int main(){
    vector<int> arr({41,1,38,26,15,40,36,18,44});
    int n = arr.size();
    for(int i = 0;i < n-1;i++){
       int smaller_value = arr[i];
       int smaller_index = i;
        for(int j = i;j < n;j++){
            if(arr[j] <  smaller_value){
                smaller_value = arr[j];
                smaller_index = j;
            }
        }
       swap(arr[i],arr[smaller_index]);
    }
    for(int i = 0;i < n;i++){
        cout << arr[i] << " ";
    }
    return 0;
}
```

Selection Sort

T.C — $O(N^2)$

S.C — $O(1)$



```
#include<bits/stdc++.h>
using namespace std;
int main(){
    vector<int> A({41,1,38,26,15,40,36,18,44});
    int n = A.size();
    for(int i = 1;i < n;i++){
       int value = A[i];
        int j = i-1;
        while (j >= 0 && A[j] > value)
        {
            A[j+1]  = A[j];
            j--;
        }
        A[j+1] = value;
    }
    for(int i = 0;i < n;i++){
        cout << A[i] << " ";
    }
    return 0;
}
```

Best Case

Insertion Sort

T.C – O(N)

S.C – O(1)

worst case

T.C – O(N^2)

S.C – O(1)