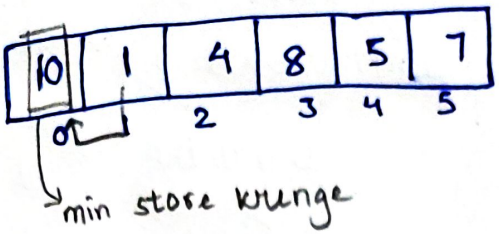
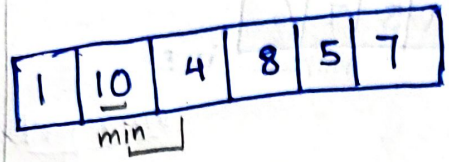
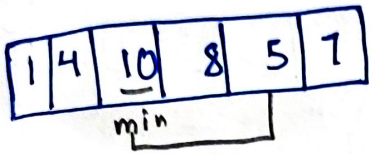
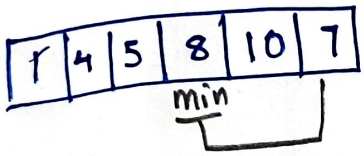
24 Feb

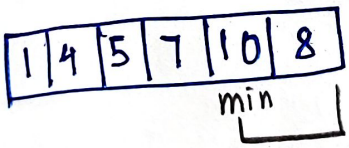
① Selection Sort :-

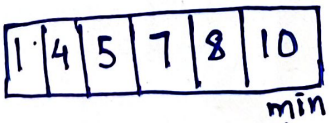




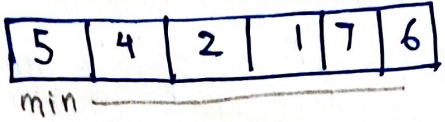


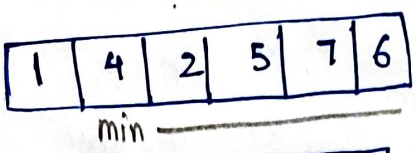


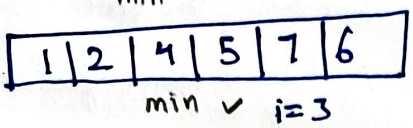


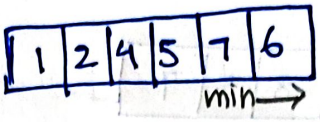


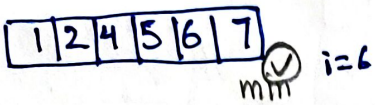
Ex 2



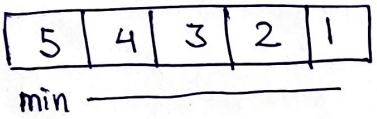


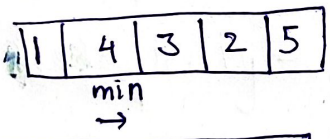


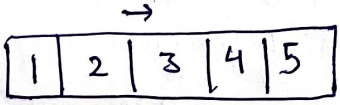




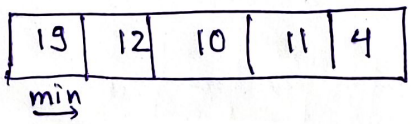
Ex3

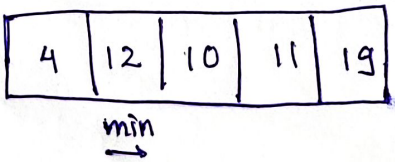


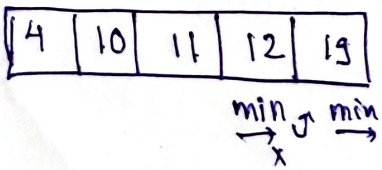




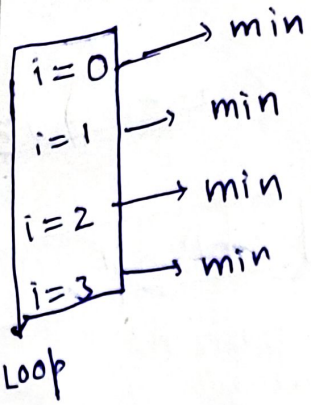
Ex4



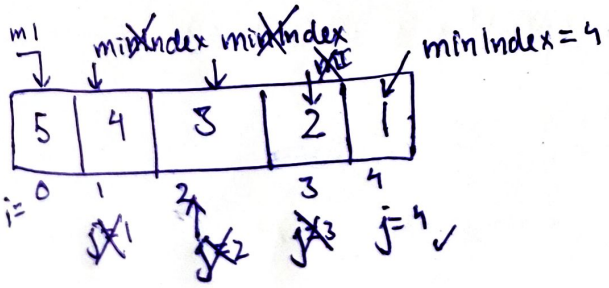




SORTED




```
int main () {
    vector <int> arr { 5, 4, 3, 2, 1};
    int n = arr.size();
    //selection sort
    //outerloop for number of rounds
    for (int i=0; i<n-1; i++) {

        int minIndex = i;
        //finds index of min. element from i to n
        for (int j = i+1; j<n; j++){
            if (arr[j] < arr [minIndex]){

            // new minimum, then store

                minIndex = j;
            }

        }
        //swap
        swap (arr[i], arr[minIndex]);

    }
```



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

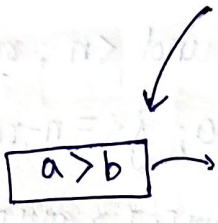
Time complexity → $O(N^2)$

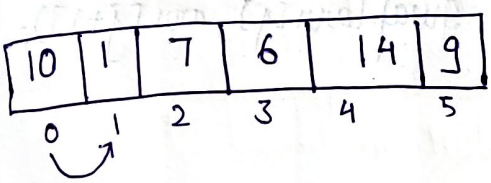
Space complexity → $O(1)$

Bubble Sort:-

→ ith Round

↳ ith largest element ko uski right position pr place krta hain





Round1 (5 comparison)

1, 10, 7, 6, 14, 9

1, 7, 10, 6, 14, 9

1, 7, 6, 10, 14, 9

1, 7, 6, 10, 14, 9

1, 7, 6, 10, 9, 14

$4+3+2+1+0$

Round2 (4 comparison)

1, 7, 6, 10, 9, 14

1, 7, 6, 10, 9, 14

1, 6, 7, 10, 9, 14

1, 6, 7, 9, 10, 14

Round3 (2 comp)

1, 6, 7, 9, 10, 14

Round4

1, 6, 7, 9, 10, 14

Round5

1, 6, 7, 9, 10, 14

Code:

```
int main(){
    vector<int> arr{10,1,7,6,14,9};
    int n=arr.size();
    for(int round=1; round<n; round++){
        for(int j=0; j<=n-round-1; j++){
            if(arr[j]>arr[j+1]){
                swap(arr[j], arr[j+1]);
            }
        }
    }
```

TC: $O(N^2)$

Ques- Can we optimize Bubble sort ?

Soln Flag and Break use ↓

```
for(int round=1; round<=n; round++){
    int swapcount=0;
    for(int j=0; j<n-round; j++){
        if(arr[j]>arr[j+1]){
            swap(arr[j]>arr[j+1]){
            swapcount++;
        }
    }
}
```

```
        if (swapCount == 0)
            break;
        }
    }
```

Already sorted k case mei O(n) hogi.

That is Best Case: O(n)

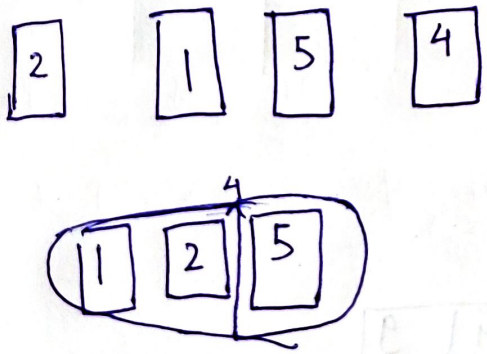
T.C.: $O(n^2)$

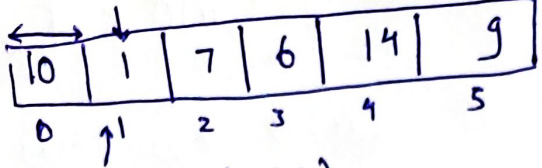
Worst case: $O(n^2)$

space Complexity: O(1)

H.W. Bubble sort ar Selection sort mei stable konsa hai?

Insertion Sort:





(A) val = 1 (fetch)
(B) 1 < 10 (comparison)
(C) shift → 10 (shifting)
(D) Put val in empty (copy)

Round 1 i = 1 → right position pr place

(A) val = 1

(B) 1 < 10 → 1 ko 10 se pehle rakhna h

(C) shifting

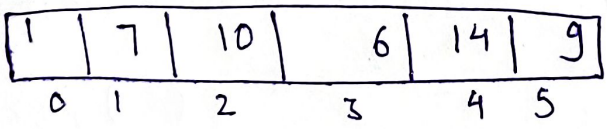
(D) copy

Round 2 i = 2 → right position pr

(A) val = 7

(B) compare 1 → (7) ← 10

(C) shifting (D) copy

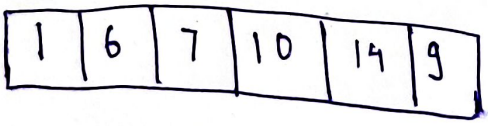


Round 3 i = 3 → right position pr

(A) val = 6

(B) compare 6 → 10 → 7 → ~~1~~ 6 > 1

(C) shift (7), (10) (D) copy



Round 4 i=4 → right posⁿ pr

(A) val = 14

(B) comp (C) X (D) X

Round 5

1	6	7	10	14	9

(A) val = 9

(B) compare 9 → 10 → 14

(C) shift (D) copy

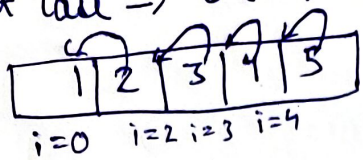
1	6	7	9	10	14

Code:

```
int main() {
    vector<int> arr {10, 1, 7, 6, 14, 9};
    int n = arr.size();

    for (int round = 1; round < n; round++){
        // Step A: Fetch
        int val = arr[i]
        // Step B: compare
        for (int j = i-1; j >= 0; j--)
            // Step C: shift
            if (arr[j] > val){
                arr[j+1] = arr[j];
            }
            else {
                //stop
                break;
            }
        }
        //step D: copy
        arr[j+1] = val;
    }
```

TC: $O(n^2)$

Best case => $O(n)$



Homework:

	Normal case		Best case		Worst case	
	TC	S.C	TC	SC	TC	SC
Selection Sort						
Bubble Sort						
Insertion Sort						

STL:

sort ( )

↳ c++ predefined func^n

Code:

```
int main(){
    vector<int> v{10,1,7,6,14,9};

    sort(v.begin(), v.end());

    //printing
    for(int i=0; i<n; i++){

        cout << v[i] << " ";

    }
    cout << endl;

    return 0;
}
```

H.W:-

① syntax

② sort predef func^{n} ki implementation konsi algo. kuti hai