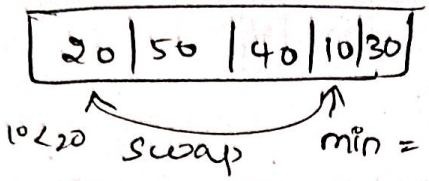
Selection Sort:

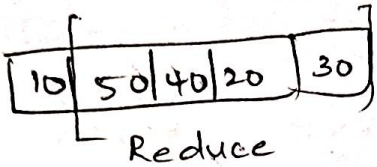
Algorithm:

→ Select the minimum
→ swap it with 1st element
→ Reduce the range / move the range to right

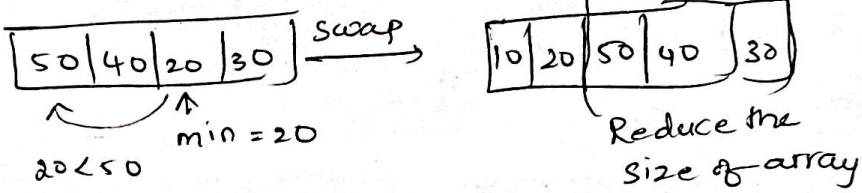
Repeat

1st time:

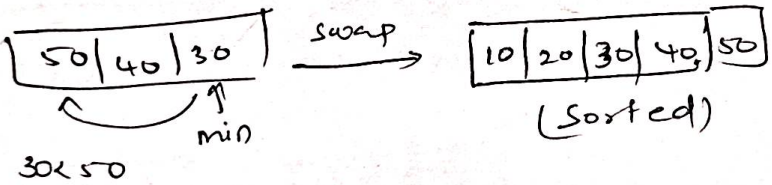




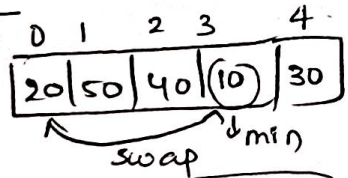
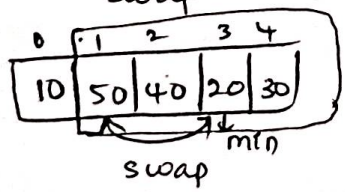
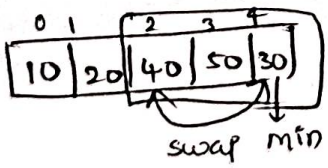
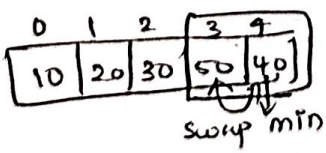
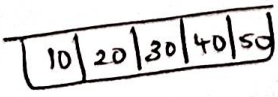
2nd time:



3rd:



Procedure:

		Range	swap
1st time		0 → 4	arr[min_index] & arr[0]
2nd time		1 → 4	arr[min_index] & arr[1]
3rd time		2 → 4	arr[min_index] & arr[2]
4th time		3 → 4	arr[min_index] & arr[3]
5th time		i → 4	arr[min_index] & arr[i]

Pseudo code:

```
for (int i=0; i<= n-2; i++){
    // find minimum
    // swap
}
```

minimum index finding:

0	1	2	3	4
20	50	40	10	30

min = arr[0]

j=0

i, j=0, arr[0] < 20, 20<20(x)

ii) j=1, 50<20(x)

iii, j=2 40<20 (x)

j=3

iii) 10<20 = (✓)

min_index = 3
min = 10

j=4

iv, 30<10 (x)

swap(arr[i], arr[min_index])

① | 10 | 50 | 40 | 20 | 30 | → Need to find from '1' index, j=1 (1 → 4) → (n-1)

② | 10 | 20 | 40 | 50 | 30 | j=2 (2 → 4) → (n-1)

③ | 10 | 20 | 30 | 50 | 40 | j=3 (3 → 4) → (n-1)

Range (j=i to n-1)

④ | 10 | 20 | 30 | 40 | 50 | j=4 so j=i

```
for (int i = 0; i <= n-2; i++) {
    int min = v1[i];
    int min_index = 0;
    for (int j = i; j <= n-1; j++) {
        if (v1[j] < min) {
            min = v1[j];
            min_index = j;
        }
    }
    swap(v1[i], v1[min_index]);
}
```