Exercise 7 Sorting Methods

7. Write a C++ Program for sorting integer numbers using following methods
a) Bubble sort b) Selection sort

Objective: The objective of this exercise is to enable you to perform sorting of integer numbers using bubble sort and selection sort methods.

Procedure and description:

a. Bubble sort:

Bubble sort is a straightforward and simplistic method of sorting data that is used very commonly. It starts at the beginning of the data set. It compares the first two elements, and if the first is greater than the second, then it swaps them. It continues doing this for each pair of adjacent elements to the end of the data set. It then starts again with the first two elements, repeating until no swaps have occurred on the last pass.

Algorithm:

K [ ]: array elements
N: size of array
LAST: position of the last unsorted element
PASS: maintain the iteration count
I: is used as index of array elements variable
EXCHS: is used to count the number of exchanges
Algorithm for Bubble sort:

```
Step 1:  [Initialize] LAST←N (entire list is unsorted at this point)
Step 2:  [Loop on pass index]
           Repeat thru step 5 for PASS = 1, 2, N-1
Step 3:  [initialize exchanges counter for this pass]
            EXCHS←0
Step 4:  [perform pairwise comparisons on unsorted elements]
             Repeat for 1 = 1, 2, LAST-1
            If K [1] > K [I+1]
           Then K [I] <-> K [I+1]
            EXCHS ← EXCHS +1
```

```
Step 5:  [were any exchanges made on this pass]
                 If EXCHS=0
                       then Return
             Else LAST←LAST-1
Step 6:  [Finished]
             Return (maximum number of passes required)
```

Expected Output:
After execution of program. Enter Size of array and enter integer numbers
Example: list of array elements with size 6

45, 67, 12, 34, 25, 39

In the first step, the focus is on the first two elements, which are compared and swapped. The reaming steps performed as for algorithm and final output is elements in sorted order

Output: 12, 25, 34,39,45,67

b) Selection sort:
Selection sort is one of the sorting techniques that are typically used for sequencing small lists. It starts by comparing the entire list for the lowest item and moves it to the #1 position. It then compares the rest of the list for the next-lowest item and places it in the #2 position and so on until all items are in the required order.

Algorithm for selection sort:
1) Find the minimum value in the list
2) Swap it with the value in the first position
3) Repeat the steps above for the remainder of the list (starting at the second position and advancing each time).

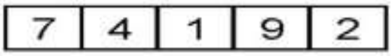
Algorithm:
A [ ]: array elements
n: size of array
MIN: position of the minimum element in array
i: is used as index of array elements variable
j: is used as index of array elements variable

```
Selection sort (A, n)
Step 1:   for i = 0 to n-1
             MIN← i
Step 2:   for j = i + 1 to n
              If (A[j] < A[i])
                MIN← j
               End if
Step 3:    Swap (A[j], A[i]; // swap min to front
Step 4:  Return
```

Expected Output:

After execution of program. Enter Size of array and integer numbers

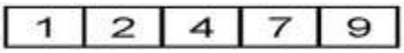
Example: list of array elements with size 5



7	4	1	9	2

The selection sort marks the first element (7). It then goes through the remaining data to find the smallest number (1). It swaps with the first element (7) and the smallest element (1) which is placed in its correct position. Performs next steps as for algorithm.

Output: list of array elements in sorted order



1	2	4	7	9