MODULE 16

SORTING OF NUMBERS USING SELECTION SORT

Up to now you have learnt comparing and sorting data. Considering the time and space complexities, there are some techniques to sort the data efficiently and effectively

Introduction:

Sorting is an important concept that is extensively used in the fields of computer science. Sorting is nothing but arranging the elements in some logical order.

For example, we want to obtain the telephone number of a person. If the telephone directory is not arranged in alphabetical order, one has to search from the very first to till the last page.

If the directory is sorted, we can easily search for the telephone number.

These are some sorting techniques like bubble sort, quick sort, insertion sort, selection sort, merge sort and heap sort etc. Among these sorting techniques you will learn selection sort, insertion sort and bubble sort.

Selection Sort :

- This is one of the simplest sorting techniques
- In the first step, first smallest element is searched in the list. Once the smallest element is found it is exchanged with the element, which is placed at first position. This completes the first pass.
- In the next step it searches for the second smallest element in the list and it is interchanged with the element placed at second position. This is done in second pass.
- This process is repeated until all the elements are sorted.
- In the first pass first smallest element is placed in first position. In the second pass second smallest element is placed in second position and in the ith pass ith smallest element is placed in the ith position. In ith pass it requires n-I comparisons.
- It requires n-1 passes to complete the sorting.

Algorithm

The algorithm works as follows:

1. Find the minimum/maximum 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

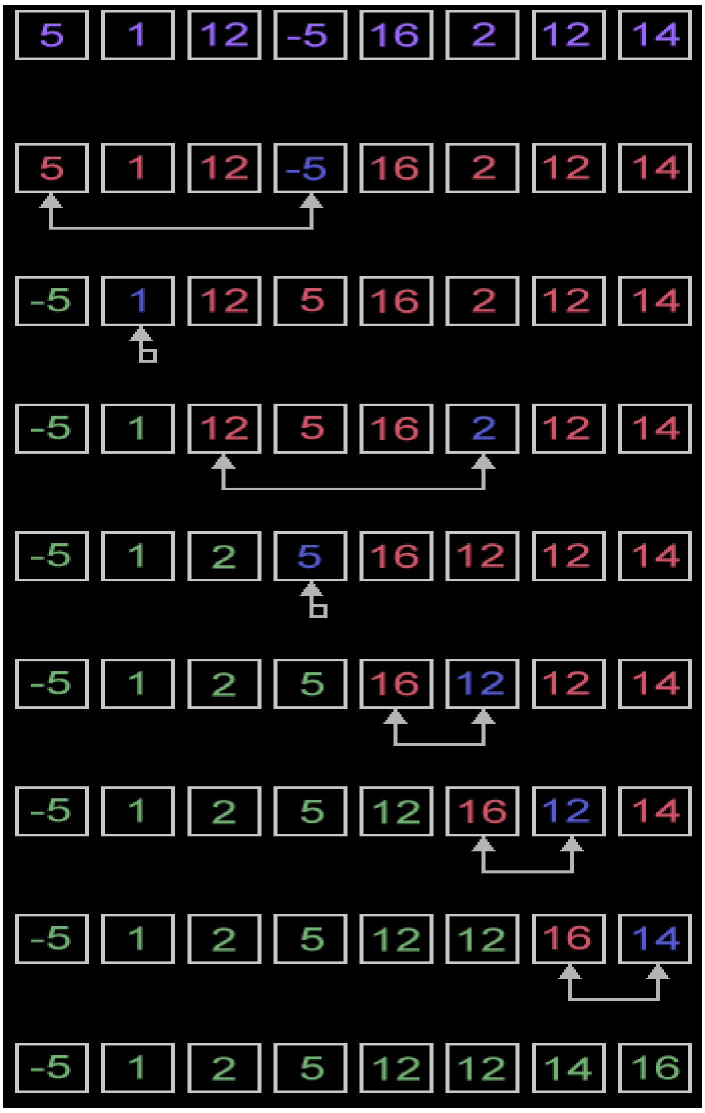
Pseudo code

SELECTION_SORT (A)

```
for $i$ ← 1 to $n$-1 do
    min $j$ ← $i$;
    min $x$ ← A[$i$]
    for $j$ ← $i$ + 1 to n do
        If A[$j$] < min x then
            min $j$ ← $j$
            min $x$ ← A[j]
    A[min $j$] ← A [$i$]
    A[$i$] ← min $x$
```

Let us see an example of sorting an array to make the idea of selection sort clearer.

Example. **Sort {5, 1, 12, -5, 16, 2, 12, 14} using selection sort.**



5 1 12 -5 16 2 12 14

5 1 12 -5 16 2 12 14

-5 1 12 5 16 2 12 14

-5 1 12 5 16 2 12 14

-5 1 2 5 16 12 12 14

-5 1 2 5 16 12 12 14

-5 1 2 5 12 16 12 14

-5 1 2 5 12 12 16 14

-5 1 2 5 12 12 14 16

Swapping example1:

```
a=12

b=13

print "Before swapping a:",a," and b:",b

temp=0

temp=a

a=b

b=temp

print "After swapping a:",a," and b:",b
```

Swapping example2:

```
a=12

b=13

print "Before swapping a:",a," and b:",b

a=a+b

b=a-b

a=a-b

print "After swapping a:",a," and b:",b
```

Swapping example3:

```
a=12

b=13

print "Before swapping a:",a," and b:",b

a=a*b

b=a/b

a=a/b

print "After swapping a:",a," and b:",b
```

Exercise problem:

Sort {2,6,3,9,2,6,3,4,56} using selection sort.

Hints to solve:

1. Insert the numbers into list (list size n and list elements should be entered by user)
2. Find out the minimum value from the list
3. Swap the minimum value with the first value in the list
4. Find the minimum value from remaining list
5. Swap with second value in the list
6. Continue this process until n-1 times