Selection Sort

Selection sort is one of the easiest sort. Reason it swaps current element with the smallest element in array.

It does this over and over again till last element.

So for first element (n-1) comparisons are done.

For second element (n-2) comparisons are done.

And so on.

We see Arithmatic Progression going on here.

(n-1) + (n-2) + (n-3)+ ...= n\*(n-1)/2=O(n^2)

Now let us see the pseudo code:

for i=0 to length(a)

index=i

for j=i+1 to n

if a[j]<a[index]

index=j

if index!=i

swap a[index] and a[i]

Below is the code for Selection Sort:

**package** com.Sorting;

/\*\*

\* Selection Sort.

\*

\* Selection sort finds the min in array and swaps it with the

\* current element.

\*

\* Time complexity is O(n^2).

\*

\* Find lowest in (n-1) elements.

\*

\* After one pass again find lowest in remaiing (n-1) elements.

\*

\* So (n-1)+(n-2)+...=n(n-1)/2

\*

\* So O(n^2)

\* \*/

**class** SelectionSort {

**public** **static** **void** main(String args[]) {

**int** list[] = { 9, 8, 7, 6, 5, 4, 3, 2, 1 };

**int** index, i, j, smallNumber;

System.***out***.print("Input is ");

**for** (i = 0; i < list.length; i++) {

System.***out***.print(list[i] + " ");

}

System.***out***.println();

System.***out***.println();

/\*\*

\* Loop through the array from 0 to n-1.

\*

\* \*/

**for** (i = 0; i < list.length - 1; i++) {

// assume min is the current element.

index = i;

**for** (j = i + 1; j < list.length; j++) {

// if this element is less then we have new min

**if** (list[j] < list[index])

// change the index of new min

index = j;

}

// if index and j are not same then swap.

**if** (index != j) {

smallNumber = list[index];

list[index] = list[i];

list[i] = smallNumber;

}

System.***out***.print("Pass " + (i + 1) + ": ");

**for** (**int** k = 0; k < list.length; k++) {

System.***out***.print(list[k] + " ");

}

System.***out***.println();

}

}

}

Output:

Input is 9 8 7 6 5 4 3 2 1

Pass 1: 1 8 7 6 5 4 3 2 9

Pass 2: 1 2 7 6 5 4 3 8 9

Pass 3: 1 2 3 6 5 4 7 8 9

Pass 4: 1 2 3 4 5 6 7 8 9

Pass 5: 1 2 3 4 5 6 7 8 9

Pass 6: 1 2 3 4 5 6 7 8 9

Pass 7: 1 2 3 4 5 6 7 8 9

Pass 8: 1 2 3 4 5 6 7 8 9