THE 'x' SORT ALGORITHM

In this sorting technique, the adjacent elements are compared and sorted. This adjacency continues for every alternative elements of the sequence.

Consider the sequence,

3 6 5 1 9 7 4

On first pass,

{3 6} {5 1} {9 7} {4 }

are grouped and sorted internally. 4 is left alone.

Now,

{3 6} {5 1} {7 9} {4 }

is the sequence.

Let the above sorting be called the 'even' sort.

Starting from the second element or from rightmost end,

{ 3} {6 5} {1 7} {9 4}

On sorting the above sequence internally,

3 5 6 1 7 9 4

is obtained. Let this be called the 'odd' sort. Performing such 'even' and 'odd' sorts alternatively by grouping appropriately, the required sorted sequence is obtained.

The Algorithm:

for k looping 'n-1' times [the algorithm worked fine for n/2 iterations]

[start with 1st or 2nd element]
if count = even number, make j = 1
else make j = 2

for j looping n – 1 times
compare adjacent elements and swap if necessary
and increment j by 2 [end of inner loop]

finally increment count by 1 [end of outer loop]

On running 'time' command, the following details were seen:

Number of Elements	Time Taken
1000	0 sec 4 ms
10,000	0 sec 254 ms
100,000	25 sec 560 ms
1,000,000	More than 15 min

Machine: 2.4 GHz Core2Duo Processors

Though the time taken was very high, interesting results were obtained by decreasing the number of outer loops to n/4, n/8, n/16, etc. The time taken was greatly reduced by such iterations.

Keeping the outer loop to n/36 (an arbitrary value), time taken for 10 lakh numbers was 2min and 25.560 sec.