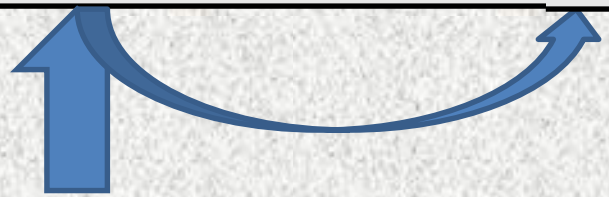
Selection Sort

Wei Guo

Selection sort Algorithm 1

- 1st. Find the index of the largest element and exchange the position with the element at the last index.

0	1	2	3	4	5	6	7	8
60	30	10	20	40	90	70	80	50



largest

0	1	2	3	4	5	6	7	8
60	30	10	20	40	50	70	80	90

2nd, Decrement the last index

When the array ends like this, the sorting stops

0	1	2	3	4	5	6	7	8
10	20	30	40	50	60	70	80	90

There are three procedures inside the selection sorting.

1st is comparison, the times of comparison in a n value array is (n(n-1))/2.

2nd is value assignment, the times of value assignment is between 0 to 3(n-1).

3rd is exchanging. The times need is between

0 to (n-1).

The best case of sorting is like:

1,2,3,4,5,6,7

The worst case of sorting is like:

7,6,5,4,3,2,1

But with the selection sort method, the time complexity for each case is the same, because whatever the array is looks like, the function will go through all the values in it. so the number of Comparisons and assignments is independent of the data.

The time complexity of selection sort

- When there is n value is the array, then during the first time, the function executes n-1 times, at the second time, it executes n-2 times…

  so the time need to sort is n-1+n-2+…+1=

  (n(n-1))/2, when n equal to infinite, it equals

  n^2

So , the best, average and worst case time complexities of the selection sort are all

O(n^2)

The algorithm of Selection sort will vary in different kinds of data structures.

If the data structure is stack or queue, the algorithm is different.

The algorithm can not run in-place, the execution need extra memory in computer. After comparison, the algorithm need extra memory to temporary store the value which need to be changed in the swap method.