ISSN 2278-7763

SELECT INSERTION SORT

GOUTAM MONDAL

Assistant teacher, Belsingha Sikshayatan

West Bengal, India

e-mail: gm_2006@rediffmail.com

Abstract:

In the paper, it is tried to pen-up a new way of sorting named select Insertion sort. The name consists of two words 'select' and 'Insertion'. The word 'select' is derived from selection sort and 'insertion' is derived from insertion sort. Truly it is said that the procedure of selection insertion sort is based on the selection sort as well as insertion sort. In the procedure, at first it is followed the way of selection sort, then the insertion sort in each step. Both procedures are followed partially.

Most of the cases, it is observed that the select insertion sort gives the better performance and more flexibility than the selection sort and insertion sort, though the time complexity is same.

SORTING:

'sorting' means rearranging the contents in increasing order generally .Let A be a list of n elements A1,A2,A3,……..,An in memory. Sorting A refers to the operation of rearranging the contents of A so that they are increasing in order either numerically or lexicographically that is, so that A1≤A2≤A3≤……≤An.

Select Insertion sort:

Suppose an array A with n elements A[1],A[2],……..,A[n] is in memory. The select insertion sort procedure for sorting A works as follows. First find the smallest element in the list and put it in the first position, moving forward one location each of the previous elements of selected elements. Then find the second smallest element in the list and put it in the second position, moving forward one location each of the previous elements of the selected elements and so on. That is:

Pass 1. Find the location k of the smallest element in the list of N elements, insert A[k] in A[1], moving forward each of the existing A[1],A[2],……..A[k-1] one location. Now A[1] is sorted.

Pass 2. Find the location k of the smallest element in the sub-list of N-1 elements, insert A[k] in A[2], moving forward each of the existing A[2], A[3],…….A[k-1] one location. Now A[1], A[2] is sorted, since A[1] ≤ A[2].

Pass 3. Find the location k of the smallest element in the sub list of N-2 elements, insert A[k] in A[3], moving forward each of the existing A[3], A[4], ……..,A[k-1] one location. Now A[1], A[2], A[3] is sorted, since A[2] ≤ A[3].

……
……
……

ISSN 2278-7763

Pass N-1. Find the location of the smaller element between A[N-1], A[N]. If A[N-1] ≤A[N], it need not be sorted. Otherwise , insert A[N] in A[N-1], moving existing A[N-1] in A[N].

N.B. This procedure takes maximum N-1 passes. If it is seen in any pass that the smallest element in the sub list is in its own position, then, follow the next pass.

Examples:

Suppose an array A contains 9 elements as follows:

40, 50, 20, 60, 80, 75, 95, 100, 10

Applying the select insertion sort procedure to A yields the data in the figure below. Observe that k gives the location of the smallest among A[M], A[M+1],……., A[N] during pass M. The second underlined element from left in each pass indicates which element to be inserted and first underlined element indicates in whose position. In any pass the only underlined element indicates both are same.

pass	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]
M=1, k=9	40	65	20	60	80	75	95	100	10
M=2, k=4	10	40	65	20	60	80	75	95	100
M=3, k=3	10	20	40	65	60	80	75	95	100
M=4, k=5	10	20	40	65	60	80	75	95	100
M=5, k=5	10	20	40	60	65	80	75	95	100
M=6, k=7	10	20	40	60	65	80	75	95	100
M=7, k=7	10	20	40	60	65	75	80	95	100
M=8, k=8	10	20	40	60	65	75	80	95	100
sorted	10	20	40	60	65	75	80	95	100

Complexity of select insertion sort:

The number f(n) of comparisons in the select insertion sort procedure can be easily computed. To find smallest element in k-th pass, it requires n-k comparisons. That is, there are n-1 comparisons during pass 1 to find the smallest element, n-2 comparisons during pass 2 to find the smallest that is the second smallest element and so on till N-1 passes. Accordingly

$F(n)=(n-1)+(n-2)+\ldots\ldots\ldots\ldots+2=(n-1)(n-2)/2 = o(n^2)$

REFERENCES:

[1]. S. Lipschutz and G.A.V. Pai , “Data Structures”, T. M. H. Education Pvt. Ltd.