**Chapter 11-1 – Sorting Algorithms and Their Efficiency**



Internal Sort – collection of data fits in memory

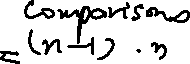


External Sort – collection of data does not fit in memory



* Must reside on secondary storage

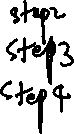
**Basic Sorting Algorithms**



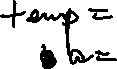
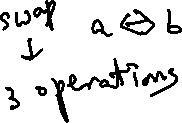
1. Selection Sort



Example) Trace the selection sort (ascending order)



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| Initial Array | 29 | 10 | 14 | 37 | 13 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |



1. Bubble Sort

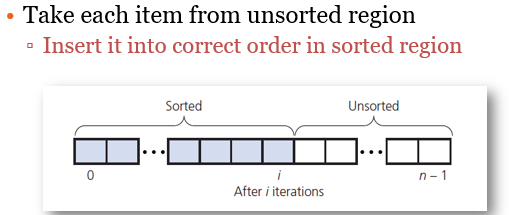
Example) Trace the bubble sort (ascending order)



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| Initial Array | 29 | 10 | 14 | 37 | 13 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |



1. Insertion sort



void insertionSort(int theArray[], int n)

{



for (int unsorted = 0; unsorted < n; unsorted++)



{

int nextItem = theArray[unsorted];



int loc = unsorted;



while ((loc) > 0 && (theArray[loc - 1] > nextItem))



{



theArray[loc] = theArray[loc - 1];



loc--;



}

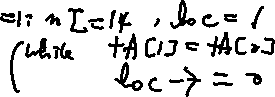


theArray[loc] = nextItem;

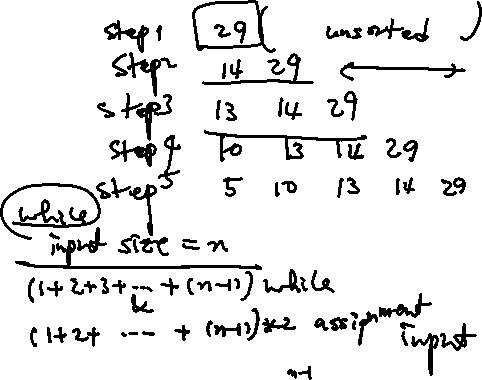


}

}

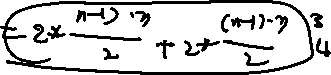
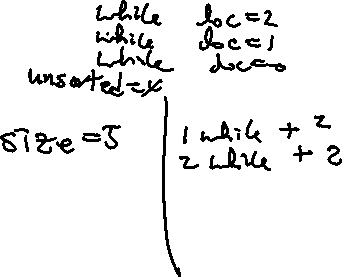


Example)



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| Initial Array | 29 | 14 | 13 | 10 | 5 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Quiz: Trace the following array into ascending order:



Array A ={20, 80, 40, 25, 60, 30}



1. Bubble Sort



1. Insertion Sort

