Basic Sorting Algorithms

Dr. Yingwu Zhu

Sorting Problem

- Consider list

  $x_1, x_2, x_3, \ldots x_n$

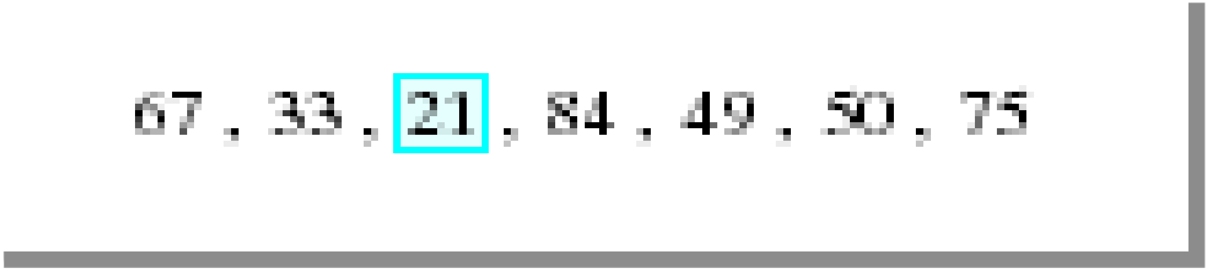
- Goal: arrange the elements of the list in order
  - Ascending or descending
- Some $O(n^2)$ schemes
  - easy to understand and implement
  - inefficient for large data sets

Basic Sorting Algorithms

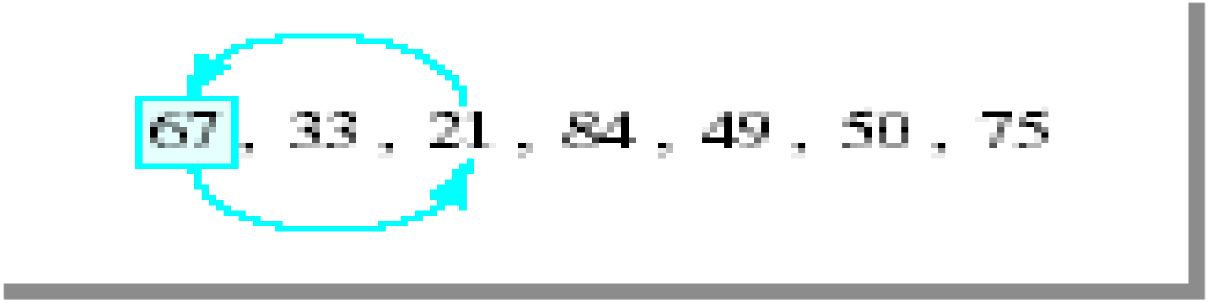
- Selection sort
- Insertion sort
- Bubble sort (Exchange sort)

Selection Sort

- Make passes through a list/sublist
- On each pass reposition correctly some element. E.g., find the smallest item in the sublist and put it into a right position



67 , 33 , 21 , 84 , 49 , 50 , 75



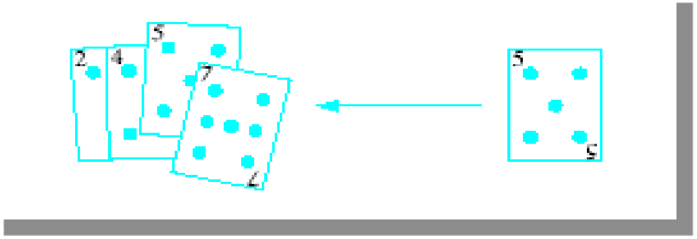
67 , 33 , 21 , 84 , 49 , 50 , 75

Implementation of Selection Sort

- Array-based
- Linked-list based

Insertion Sort

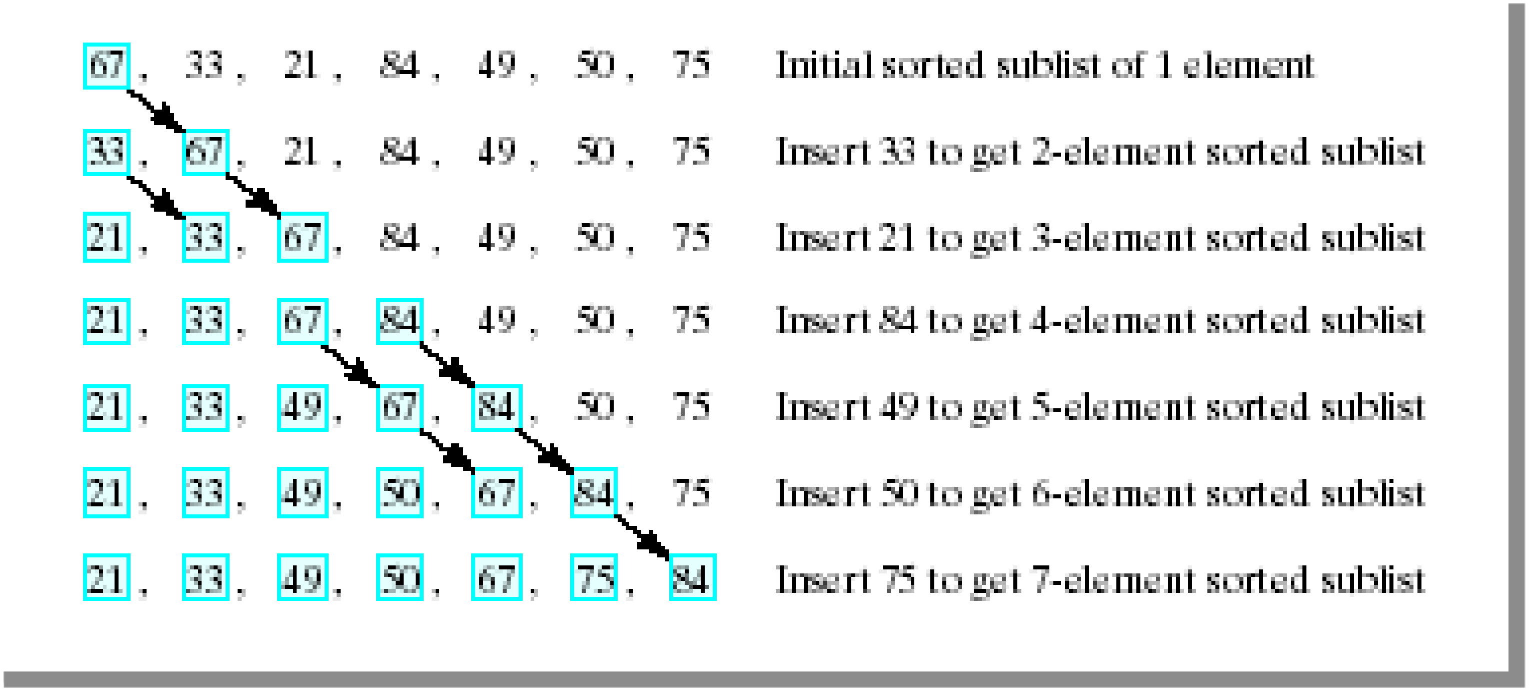
- Repeatedly insert a new element into an already sorted list



- Incremental algorithm
  - Incrementally build up the sorted list

Example of Insertion Sort

- Given list to be sorted
  67, 33, 21, 84, 49, 50, 75
  - Note sequence of steps carried out



67 ,	33 ,	21 ,	84 ,	49 ,	50 ,	75	Initial sorted sublist of 1 element
33 ,	67 ,	21 ,	84 ,	49 ,	50 ,	75	Insert 33 to get 2-element sorted sublist
21 ,	33 ,	67 ,	84 ,	49 ,	50 ,	75	Insert 21 to get 3-element sorted sublist
21 ,	33 ,	67 ,	84 ,	49 ,	50 ,	75	Insert 84 to get 4-element sorted sublist
21 ,	33 ,	49 ,	67 ,	84 ,	50 ,	75	Insert 49 to get 5-element sorted sublist
21 ,	33 ,	49 ,	50 ,	67 ,	84 ,	75	Insert 50 to get 6-element sorted sublist
21 ,	33 ,	49 ,	50 ,	67 ,	75 ,	84	Insert 75 to get 7-element sorted sublist

Insertion Sort

- Idea:
  - Two logical sublists: one is sorted and the other is unsorted
  - Each iteration chooses the first item from the unsorted list and inserts it into the sorted one.
  - Dynamically expand/shrink the two sublists

Implementation for Insertion Sort

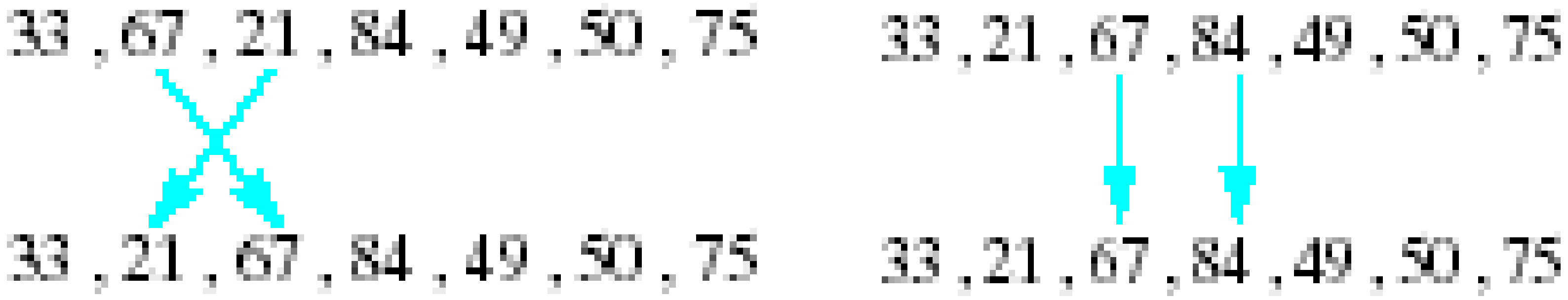
- Array-based
- Linked-list based
- Which one do you prefer?

Insertion Sort

- Can we write a recursive insertion sort?

Bubble Sort

- Systematically interchange pairs of elements which are out of order
- Bubble sort does this



Out of order, exchange

In order, do not exchange

Bubble Sort: First Shot – Naïve

- void bubble_sort(int a[], int n)
    - For naïve implementation, how many iterations should be made in order to make the list in order?

Bubble Sort: Optimization!

- Can we improve the naïve implementation?
  - Detect partially sorted sublist!
  - Leave it alone!
  - How to detect partially sorted sublist?

Bubble Sort

```
void bubble_sort(int a[], int n) {
   int num_compares = n-1; //first should do n-1 comparisons
   while (num_compares > 0) {
      int last = 0;  //why need this?
      for (int i=0; i<num_compares; i++) {
         if (a[i] > a[i+1]) {
            swap(a[i], a[i+1]);    last = i;
         } //end if
      num_compares = last;
   } //end while
}   // thinking: why need last = i???  The purpose of last?
```

Bubble Sort

- Disadvantage?
  - Swap of data items, but if data item is large, swap could be very inefficient
- Advantage over selection sort?
  - It can detect partially sorted sublist.

Bubble Sort Algorithm

What is the worst case for Bubble Sort?

Bubble Sort Algorithm

What is the worst case for Bubble Sort?

The list of items are in decreasing order.

T(n) = O(n^2)

Bubble Sort Algorithm

What is the best case for Bubble Sort?

Bubble Sort Algorithm

What is the best case for Bubble Sort?

$T(n) = O(n)$

Can we have better Sorting Algorithms

- We seek improved computing times for sorts of large data sets, better than O(n^2)
- Chapter presents schemes (e.g. heapsort) which can be proven to have average computing time

$$O(\,n \log_2 n\,)$$

- Must be said, no such thing as a universally good sorting scheme
  - Results may depend just how out of order list is

Indirect Sorts

- Possible that the items being sorted are large structures
  - Data transfer/swapping time unacceptable
- Alternative is indirect sort
  - Uses index table to store positions of the objects
  - Manipulate the index table for ordering
- Where will this technique be used? Name one ☺

Review

- Selection sort
- Insertion sort
  - Efficient on small input sizes!
- Bubble sort