CSC 142

Sorting

[Reading: chapter 11]

The problem

Given `a[0], a[1], ..., a[n-1]`

reorder entries so that

`a[0] <= a[1] <= ... <= a[n-1]`

→ a very common operation

→ lots of application
(search in a database...)

First: Look at a simpler problem

Suppose

`a[0] <= a[1] <= ... <= a[i-1]`

How to add `a[i]` so that

`a[0] <= a[1] <= ... <= a[i]`

e.g : 1 4 5 8 9 6

add 6 → 1 4 5 6 8 9

Insertion algorithm

1. Store `a[i]` in a temporary variable `temp`
2. Shift all elements > `temp` to the right by one position
3. Insert `temp` in the space created

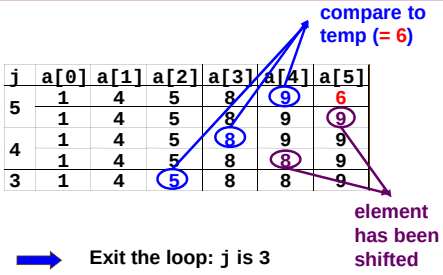
Requirement:
a is sorted up through `a[i-1]`
(if not, does NOT work)

Outcome:
after insertion, a is sorted up through `a[i]`

Insert as a method

```
public void insert(int[] a, int i)
{
   int j, temp;
   temp = a[i];
   for(j=i; j>0 && a[j-1]>temp; j--)
      a[j] = a[j-1];
   a[j] = temp;
}
```


How does it work?



j	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
5	1	4	5	8	9	6
	1	4	5	8	9	9
4	1	4	5	8	9	9
	1	4	5	8	8	9
3	1	4	5	8	8	9

→ Exit the loop: `j` is 3

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
1	4	5	6	8	9

Insertion Sort

→ Use `insert`

```
public void sort (int[] a)
{
     int i;
     for(i=1; i<a.length; i++)
         insert(a,i);
}
```

start at 1

Example: `sort(a)`

i	a[0]	a[1]	a[2]	a[3]	a[4]
initially	10	20	15	5	3
1	10	20	15	5	3
2	10	15	20	5	3
3	5	10	15	20	3
4	3	5	10	15	20

Bubble Sort

Idea :

- Compare `a[i]` and `a[i+1]`
- if `a[i] > a[i+1]`,
  exchange `a[i]` and `a[i+1]`

Loop over :

`i=0` → `dim - 2`
`i=0` → `dim - 3`
⋮
`i=0` → `0`

Example: first pass of bubble sort (i=0 to 3)

i	a[0]	a[1]	a[2]	a[3]	a[4]
initially	10	20	15	5	3
0	10	20	15	5	3
1	10	15	20	5	3
2	10	15	5	20	3
3	10	15	5	3	20

20 drifts to the top (like a bubble in a liquid)

For all of the passes

Pass	a[0]	a[1]	a[2]	a[3]	a[4]
initially	10	20	15	5	3
1	10	15	5	3	20
2	10	5	3	15	20
3	5	3	10	15	20
4	3	5	10	15	20

Efficiency is the key

Bubble Sort is less efficient than Insert Sort

→ **Check it out**

There are many sorting algorithms :
to find the best algorithm, theory + practice