Insertion Sort O(n^2)

Input: array A[1 .... n]

Goal: sort in increasing order

Algo in words: Scan the array from left to right. For each position j, insert A[j] to its correct postion in A[1 ... j] by moving all elements larger than it one position to the right.

```
for j = 2 to n:
    key <- A[j]
    i = j - 1
    while i > 0 and A[i] > key:
        A[i+1] <- A[i]
        i <- i - 1
    A[i+1] = key
```

Proof of correctness:

Loop invariant: at beginning of the jth iteration (or end of j-1 st iteration), A[1 ... j-1] contains the same elements it initially contained, but in sorted order.

If true, then at the end of the nth iteration, A is sorted, as required.

Claim: Loop invariant holds for j = 2, 3, ..., n+1.

Proof: By induction

When j = 2 (base of induction), A[1] is just one number and is therefore sorted.

Next, assume holds for j, and let us show that holds for j+1. By assumption, A[1 ... j-1] is sorted. During the jth iteration, we insert A[j] into its correct position. Therefore, the invariant holds at the end of the jth iteration, as required.

Running time: O(n^2)

$\Sigma_2 j(c + c' c_j) = c^2/2n^2 + n - () = O(n^2)$