Sorting a Sequence

→ Insertion Sort:

- start w/ first element
- if next element smaller than first, then swap
- consider 3rd element, swap w/ first 2 if necessary
- continue until all in place

InsertionSort(A):

→ Input: array A of n comparable elements

→ Output: array A w/ elements rearranged in nondecreasing order

for k from 1 to n-1

Insert A[k] at proper location

↓

```
def insertion_sort(A):
    for k in range(1, len(A)):          # 1 to n-1
        curr = A[k]                     # curr el to insert
        j = k                           # find correct index
        while j > 0 and A[j-1] > curr:  # A[j-1] must be after curr
            A[j] = A[j-1]
            j -= 1
        A[j] = curr                     # curr in right place
```

MultiDimensional Datasets

→ lists, tuples, and strings in Python are 1D

→ 2D array = matrix, where i and j are used to refer to cells in the matrix

→ Constructing: to properly init a 2D list, we ensure each cell of primary list refers to an independent instance of a secondary list.

```
data = [[0] * c for j in range(r)]
```