Sorting Algorithms

There are many situations in programming where you need to put a sequence of data values into some kind of order. The list type in Python has a method sort() that will put the data in the list into increasing order. This works for simple data like numbers or strings, but not for more general kinds of data. Python has to know how to compare two elements before it can sort a list of them.

Here are a few algorithms you can use if you ever need to sort a list yourself, as you will in Lab 7.

BubbleSort: Walk through the list comparing each element and the next, interchanging them if they are in the wrong order. Keep doing this until you can make a pass through the list without finding any elements out of order.

SelectionSort: Make a pass through the list to find the smalllest element; interchange it with the element at index [0]. Then make a pass through the list starting at index [1] to find the smallest remaining element; interchange it with the element at index [1]. Do this through the entire list.

InsertionSort: Keep a sorted portion of the list and an unsorted portion. At each step take the next element out of the unsorted portion and put it at the right spot in the sorted portion.

MergeSort: There is even a recursive sorting algorithm. If the length of the list is 0 or 1 do nothing; it is already sorted. Otherwise split the list in half, recursively sort each half, then merge the two sorted halves into one completely sorted list.