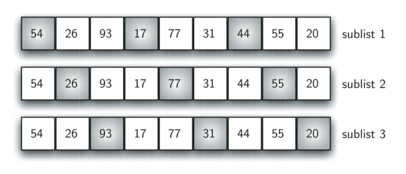
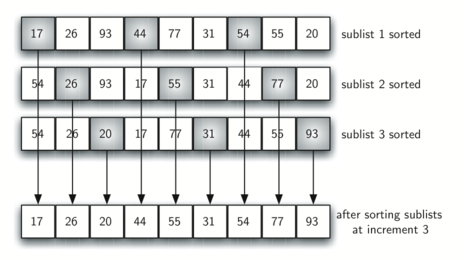
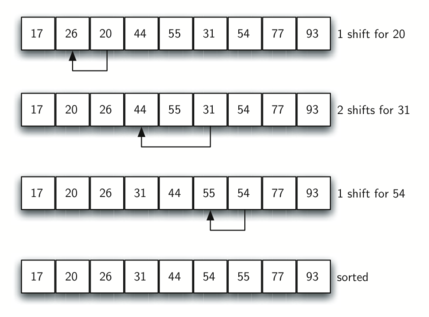
The **shell sort**, sometimes called the “diminishing increment sort,” improves on the insertion sort by breaking the original list into a number of smaller sublists, each of which is sorted using an insertion sort. The unique way that these sublists are chosen is the key to the shell sort. Instead of breaking the list into sublists of contiguous items, the shell sort uses an increment i, sometimes called the **gap**, to create a sublist by choosing all items that are i items apart.

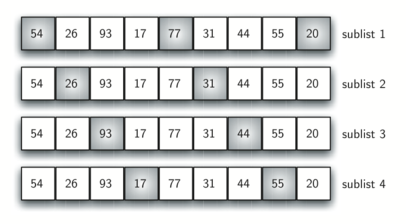
This can be seen in [Figure 6](http://interactivepython.org/runestone/static/pythonds/SortSearch/TheShellSort.html#fig-incrementsa). This list has nine items. If we use an increment of three, there are three sublists, each of which can be sorted by an insertion sort. After completing these sorts, we get the list shown in [Figure 7](http://interactivepython.org/runestone/static/pythonds/SortSearch/TheShellSort.html#fig-incrementsb). Although this list is not completely sorted, something very interesting has happened. By sorting the sublists, we have moved the items closer to where they actually belong.





[Figure 8](http://interactivepython.org/runestone/static/pythonds/SortSearch/TheShellSort.html#fig-incrementsc) shows a final insertion sort using an increment of one; in other words, a standard insertion sort. Note that by performing the earlier sublist sorts, we have now reduced the total number of shifting operations necessary to put the list in its final order. For this case, we need only four more shifts to complete the process.





We said earlier that the way in which the increments are chosen is the unique feature of the shell sort. The function shown in [ActiveCode 1](http://interactivepython.org/runestone/static/pythonds/SortSearch/TheShellSort.html" \l "lst-shell) uses a different set of increments. In this case, we begin with n2n2sublists. On the next pass, n4n4 sublists are sorted. Eventually, a single list is sorted with the basic insertion sort. [Figure 9](http://interactivepython.org/runestone/static/pythonds/SortSearch/TheShellSort.html#fig-incrementsd) shows the first sublists for our example using this increment.

The following invocation of the shellSort function shows the partially sorted lists after each increment, with the final sort being an insertion sort with an increment of one.