Project 20A Radix Integer Sort(Using ArrayList)

Replace the two arrays in the example above (*zeros* and *ones*) with two *ArrayList* objects and then run the *Tester* class on page 40-1 to test the sorting of positive integers.

In both the example and project above, the **keys** used to sort were the binary digits (ones and zeros). If *String*s are to be sorted, the keys to be used are the characters that comprise the *String*s (or equivalently, their ASCII codes).

In the example above, only two arrays were needed to store the temporary results since the binary keys could result in only one of two values (1 or 0). However, if we assume the *String*s to be sorted consist only of capital letters (A-Z), then the corresponding 26 ASCII codes range from 65 to 90. Clearly, 26 storage areas are needed as opposed to just two as was the case with binary keys.

Project 20B Radix String Sort(Using ArrayList)

Create a radix type *sort* method that receives an array of *String*s and modifies the code in the example above by replacing the *zeros* and *ones* arrays with an **array** of 26 *ArrayList* objects.

Create a *Tester* class having a *main* method that calls the *sort* method and passes to it the following *String* array consisting of only **capital letters** (A-Z), and with each *String* having **exactly 4** characters:

{"DELL", "HELP", "ALSO", "BEAR", "BACK", "IPAD", "IPOD", "AGRO"}

The *sort* method should sort this array in descending order and then after the completion of the call to *sort* from the *Tester* class, the properly sorted array should be printed.

After completion of this project, consider how the code would be modified to accommodate Strings of variable length. Also, consider how to modify the code to sort in ascending order.