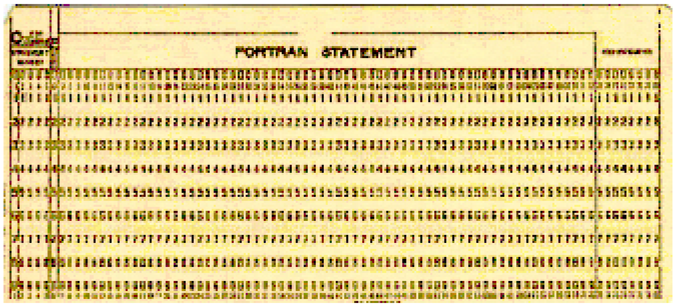
Radix Sort History

Radix sort is another linear sorting algorithm, but this one makes use of a secondary sorting function to work. The idea is motivated by an old piece of equipment created for the US Census Bureau and used extensively in other organizations such as banks—the card sorting machine. Many years ago, data (and programs) were "stored" on what were called Hollerith cards, such as the following:



This particular card was printed to help FORTRAN programmers, but the idea was applied in general. Specifically, each card had 80 columns, and a card punch machine would punch patterns into the cards corresponding to Hollerith codes to represent characters. An example of a card punch machine (and one your instructor used while an undergraduate) is as follows:

This picture is from http://www.computinghistorymuseum.org/museum/punch_zoom.htm at the computing history museum at American University.

Data was punched on cards using a machine such as the above, and then the data was sorted using a special "card sorting" machine, such as the following:

This picture is taken from http://www-03.ibm.com/ibm/history/exhibits/attic3/attic3_136.html and represents the IBM Type 80 electric punch card sorting machine, created in 1925.

The way the card sorter worked was in multiple steps. First, the cards were placed in a “hopper” at one end of the machine, and the machine was set to scan the right-most column. The cards were then scanned and dropped into slots based on what holes were punched. The cards were then extracted from the machine in slot order and placed back in the hopper. The machine was then set to scan the next column to the left, and the process was repeated. This continued until the cards were completely scanned, and the resulting deck was sorted.

Radix sort implements this exact idea in software. Key to radix sort working is the use of a secondary sort function to put the elements into their slots. This secondary sort must be a “stable” sort.

Definition: A sorting algorithm is said to be stable if elements of the same value appear in the same relative order in the output array as they did in the input array.

Consider the six comparison-based sorts we discussed above, we see that bubble sort, insertion sort, and merge sort are all stable. Unfortunately, heap sort, quicksort, and selection sort are not. Note, however, that counting sort is stable! We will use this to our advantage later.