Sorting in Linear Time

In the previous sections, we just expended a good bit of effort to show that the best comparison sort possible would have performance $\Theta(n \lg n)$. But the header of this section suggests we are now about to discuss sorting algorithms with $O(n)$ complexity. How can this be possible? The "obvious" answer (although it really is not very obvious) is that we are going to consider sorting algorithms that do not fit the model of "comparison-based sorting." In the following we will examine three different algorithms.

- Counting sort
- Radix sort
- Bucket sort

Warning: A common mistake when applying these algorithms is to assume they are applicable in all situations and yield linear performance in all situations. **This is not true!!** Careful consideration of the assumptions and conditions for linearity must be included in any decision to use one of the following algorithms.