Bucket Sort

- uses hashing – the values we are sorting are hashed

- Makes assumptions the data, like radix and counting sort

- because it makes assumptions, can sort in O(n) time

- performs best when hashed values of items being sorted are evenly distributed, so there aren't many collisions

- Not in-place algorithm

- Stability will depend on sort algorithm used to sort the buckets – ideally, we want a stable sort

- to achieve O(n), must have only one item per bucket

- Insertion sort is often used to sort the buckets, because it is fast when the number of items is small

How it works

A generalization of Counting Sort

1. Distribute the items into buckets based on their hashed values (scattering phase)
2. Sort the items each bucket
3. Merge the buckets – can just concatenate them (gathering phase)

The value in bucket X must be greater than the values in bucket X - 1 and less than the values in bucket X + 1 – this means that the hash function we us must met this requirement

1) Values

54	46	83	66	95	92	43

2) Place items into Buckets (For example : Hash function uses the value of tens to hash them)

				46 -> 43	54	66		83	95 -> 92

3) Sort items in Buckets

				43 -> 46	54	66		83	92 -> 95

4) Merge Buckets

43	46	54	66	83	92	95