3. The distribution counting sort requires a lot of pre-known information to run. It does this by increasing a count on a counting integer that has a spot for every number between the high and low values. This means that if you have a large range of numbers, the operating power used is going to be much bigger, as a larger array will need to be created. Then for each number in the original array, there is a read through to determine the next value that is needed for that indexed count. Once the counts are complete, a new array is formed at the same size as the original and the value at each spot is determined by the counts at each number. Because this number can change based on the size and range, the larger the data set the bigger the number of required save locations. Overall, this sort can be a very useful tool, if you have an average amount of data, especially with a lot of repeats, and a small range of possible numbers to fill. The smaller the range, the more efficient this sort method will be.

4. distribution counting is a stable sort, because once the data is removed from the original array into the counting array, it can be placed into a new array, making it stable, or it can just be placed back in the original for sake of saving resources. In general however, the sort does not move elements around, making it stable.