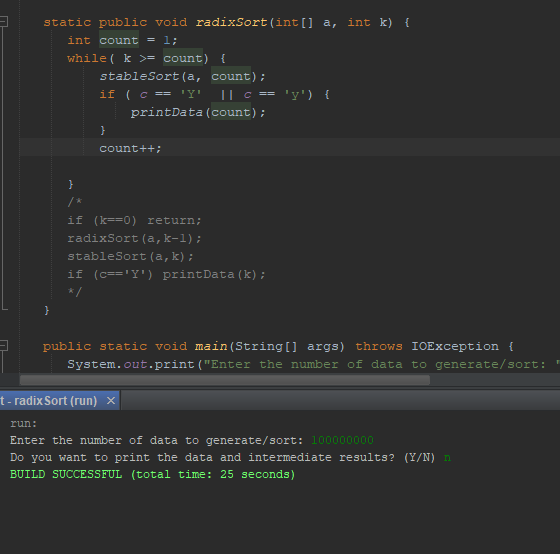
Task 2)  
 

Task 3) I ended up getting the iterative code by looking at the recursion given. All that is needed to is take a loop which is which K is not >= count which count is a temp variable to hold the base case then sort the numbers and using stableSort. If the user says they are wanting the print then print values created, then increment the count.

Task 4) When looking at the recursive version of radixSort you can see that the recursion is correct. When looking at the base case of the recursion which is when K (which is an integer which at most is the value of 9) is equal to 0. If the value of K is not the recursive call will occur showing until K-1 is equal to 0, at this point the last instance of the recursion will return causing the previous instance to call stableSort(A, K) where A is the size of an integer array. For example when K = 2 and A = 10 the recursive call at stableSort(10, 2-1) at this point the next instance of the recursive call of Once this occurs that instance of the recursion will complete moving on to the previous instance stableSort(10, 1-1) will be called and the base case will return. The instance in which K = 1 will sort the array A than the original call when K = 2 will sort its number than radixSort will complete which assuming that the stableSort will always sort the values of K correctly. Once until the last instance has been completed which will prove the recursive version of radixSort is correct.