1. In recitation a few weeks ago, we saw the K-MERGE procedure that merged \( k \) sorted lists (each of length \( n \)) into 1 sorted list. We managed to achieve a worst-case running time of \( \Theta(nk \log k) \) for K-MERGE by using a heap.

(a) Assume that the elements of all \( k \) lists are drawn from the set \{0, 1, ..., \( l \)\}, and that \( l = O(n) \). Describe an implementation of K-MERGE that achieves a better time bound than \( \Theta(nk \log k) \).

(b) Does your new algorithm for K-MERGE require that the \( k \) input lists be sorted beforehand?

2. Show how to sort \( n \) integers in the range 0 to \( n^2 - 1 \) in \( O(n) \) time. HINT: Think about what you heard in lecture about RADIX-SORT.