Istanbul Technical University
Faculty of Computer and Informatics

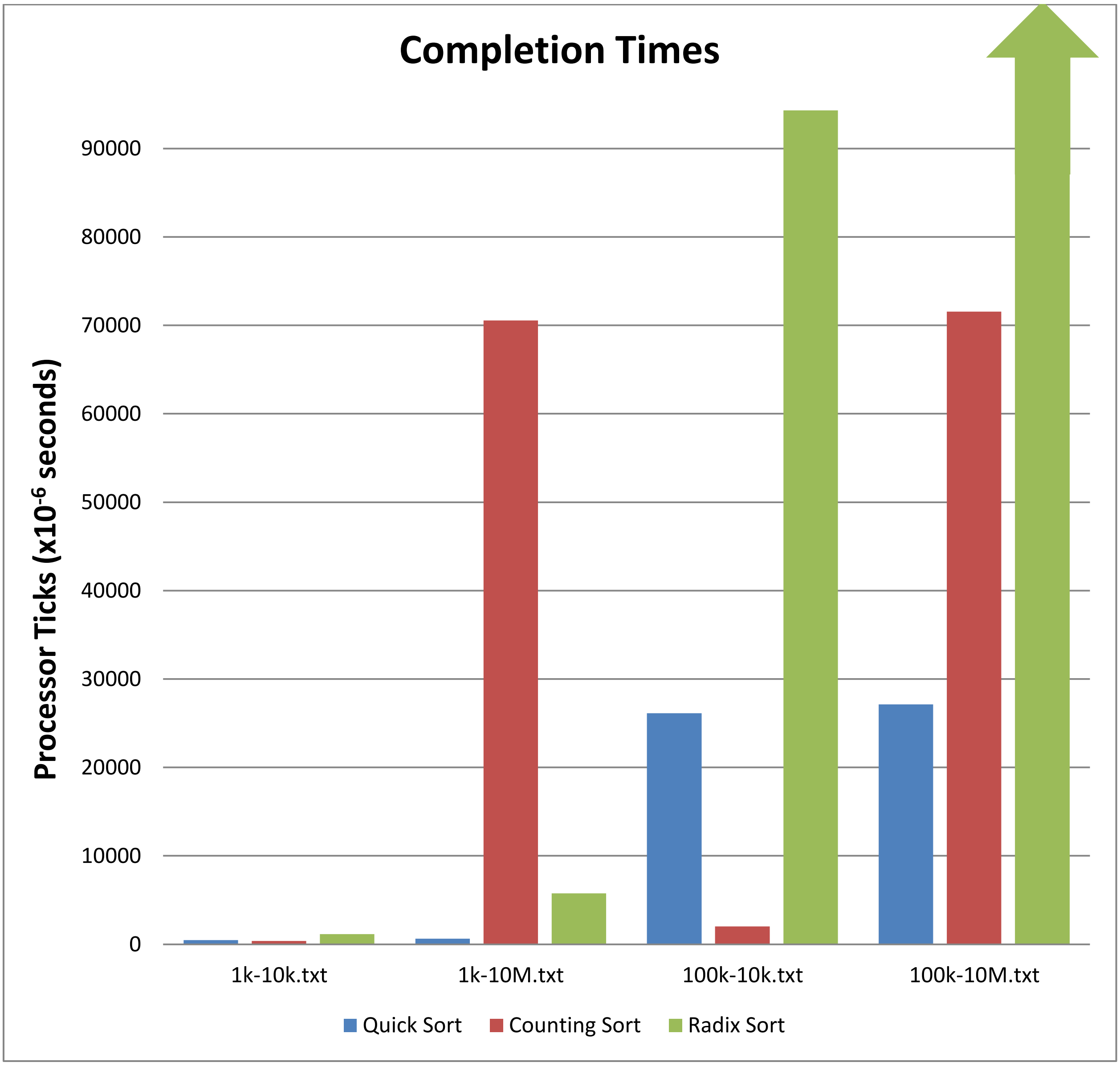
BLG335E Analysis of Algorithms I
Project 3

Cem Yusuf Aydoğdu
150120251

Part B

1. Completion times of algorithms for each dataset is shown below

	Quick Sort	Counting Sort	Radix Sort
1k-10k.txt	486 ticks, 0.000486 seconds	371 ticks, 0.000371 seconds	1137 ticks, 0.001137 seconds
1k-10M.txt	638 ticks, 0.000638 seconds	70548 ticks, 0.070548 seconds	5773 ticks, 0.005773 seconds
100k-10k.txt	26129 ticks, 0.026129 seconds	2009 ticks, 0.002009 seconds	94302 ticks, 0.094302 seconds
100k-10M.txt	27129 ticks, 0.027129 seconds	71545 ticks, 0.071545 seconds	236831 ticks, 0.236831 seconds



2.

a. Pivot selection affects partitioning in quick sort. In case of ordered or reverse ordered input, selecting the pivot from always first or last element will cause an unbalanced partitioning which corresponds to time completixy of $O(n^2)$, therefore pivot selection should be randomized for better(balanced) partitioning.

b. Worst case in quick sort is getting the input as ordered or reverse ordered, which causes unbalanced partitioning. Time complexity of this case is $O(n^2)$.

3. k denotes the maximum number in the array and the size of histogram array (*counts* in pseudocode). Time complexity of loops about array is $O(n)$ and completixy of loops about histogram is $O(k)$, so total time complexity is $O(n + k)$.
k must be specified because when $k \gg n$, k determines overall compexity

4. In this project, radix sort is implemented to operate counting sort according to one binary digit in each pass, which corresponds to 32 pass for integer data type. Complexity is $\theta(n + 2^{32})$, which is $\theta(2^{32})$ since 2^{32} grows much faster than n.