COL202: Discrete Mathematical Sturctures. I semester, 2017-18.
Quiz 3
10 August 2017
Maximum Marks: 4

Name	**Ent. No.**

Important: Keep your answer within the box. Anything written outside the box will be treated as rough work. Do your rough work on the free space on the flip side of this sheet.

Given two sorted (in ascending order) integer lists A and B, the *merge* operation creates a third list C that contains all the elements of A and B in sorted order. Let us tweak this and define a new operation *check-and-merge* as follows: if the largest element of A is smaller than the smallest element of B simply concatenate B behind A. If not check again with the positions of A and B reversed. If both conditions don't hold then perform normal merging. We assume that the checking and concatenating (if possible) take 1 unit of time, while normal merging takes time equal to the size of A plus the size of B.

Now, with this new operation, we define a new algorithm *check-and-merge sort* which is the same as the usual merge sort except that it uses *check-and-merge* as a subroutine where merge sort uses the normal merge. Recall that merge sort works as follows: Divide the list into two equal parts, recursively merge sort both parts, then merge the two recursively sorted lists. As the base case: a list with a single element is considered sorted.

Q1. (2 marks) If $T(n)$ is the time to sort an *already sorted* list using *check-and-merge sort*, find a recurrence for $T(n)$. An answer with no argument gets 0 marks.

Continued overleaf

Q2. (2 marks) Write a paragraph of 3-5 sentences proving that the time taken by *check-and-merge sort* on a sorted list of size n is cn where c is some number that doesn't depend on n. Do *not* use any recurrence solving formula. Make arguments. Failure to follow these instructions will get you 0 marks.