# CS 430 – FALL 2023
# INTRODUCTION TO ALGORITHMS
# HOMEWORK #4

1. 1a) (5 points) Show the red-black trees that result after successively inserting the keys 41, 38, 31, 12, 19, 8 into an initially empty red-black tree.

   1b) (5 points) In 1a) you found the red-black tree that results from successively inserting the keys 41, 38, 31, 12, 19, 8 into an initially empty tree. Now show the red-black trees that result from the successive deletion of the keys in the order 8, 12, 19, 31, 38, 41.

2. (7 points) Describe a red-black tree on $n$ keys that realizes the largest possible ratio of red internal nodes to black internal nodes. What is this ratio? What tree has the smallest possible ratio, and what is the ratio?

3. (7 points) Show that the longest simple path from a node $x$ in a red-black tree to a descendant leaf has length at most twice that of the shortest simple path from node $x$ to a descendant leaf.

4. (10 points) BT&T (Big Telephone & Telegraph) has 256 million customers. Their current telephone directory consists of many heavy volumes typeset in small point size, which are expensive to print and inconvenient to use. To overcome the above problems, BT&T has decided to set up an on-line computerized directory, and their software engineers are debating what is the most efficient data structure for the purpose. Assume that the BT&T computer can compare two names in one microsecond (1/1,000,000 or $10^{-6}$ of a second). You do not need to simplify your calculations.

   4a) One of the engineers suggests implementing the on-line directory as an unsorted linked list. With this implementation, give an estimate of the worst-case search and insertion times.

   4b) Another engineer wants to implement the on-line directory as an red-black tree. With this implementation, give an estimate of the worst-case search and insertion times.

   4c) A third engineer proposes the use of a sorted array. With this implementation, give an estimate of the worst-case search and insertion times.

5. (6 points) Write a non-recursive version of OS-SELECT($x$, $i$) to find the ith largest node in an order-statistic tree (initial call OS-SELECT(root, $i$))