CS 341, Spring 2020
Semih Salihoglu

Assignment 1 Part 2 (due Sunday, May 31, midnight EST)

Instructions:

- Hand in your assignment using Crowdmark. Detailed instructions are on the course website.
- Give complete legible solutions to all questions.
- Your answers will be marked for clarity as well as correctness.
- For any algorithm you present, you should justify its correctness (if it is not obvious) and analyze the complexity.

1. [*15 marks*] A group of hackers from an enemy organization has attempted to install a virus to n of your company's computers. Your software engineers have designed a test, called `TEST-EACH-OTHER`, that takes two computers c_A and c_B, where each input computer tests the other and outputs whether the other one is infected with the virus (+) or not infected with the virus (−). If a computer is actually − than it will always output a correct result. Unfortunately, if it is +, its reply is unrelated to the real state of the other computer and hence cannot be trusted. In other words, a computer c_A that is infected with the virus can be "dishonest" and output the correct or the incorrect state of c_B.

   The following table summarizes the four possible outcomes of running `TEST-EACH-OTHER` on two computers c_A and c_B, and what we can conclude from it. Please review the table to ensure that these are indeed the possible outcomes.

c_A**'s output**	c_B**'s output**	**Conclusion**
c_B is −	c_A is −	either both − or both +
c_B is −	c_A is +	at least one is +
c_B is +	c_A is −	at least one is +
c_B is +	c_A is +	at least one is +

   Luckily your security experts have told you that more than $n/2$ computers were not infected (so they are −). Your goal is to identify all the + and − computers. Below, running one instance of `TEST-EACH-OTHER` constitutes one test.

   (a) [*12 marks*] Describe an algorithm to find a single − phone by performing $O(n)$ tests. [Hint: Think of how you can use $O(n)$ tests to reduce the problem size by a constant factor.]

   (b) [*3 marks*] Using part (a), show how to identify the condition of each computer by performing $O(n)$ tests.

2. [*12 marks*] Consider the recurrence:

$$T(n) = 2T(\lfloor n/9 \rfloor) + \sqrt{n} \qquad \text{if } n \geq 9$$

$$T(n) = 5 \qquad \text{if } n < 9$$

Prove $T(n) = O(\sqrt{n})$ by induction (i.e., guess-and-check or substitution method). Show what your c and n_0 are in your big-oh bound. Note that depending on the choice of your n_0, you might have to cover multiple base cases in your inductive proof.

3. [*16 marks*] Give tight asymptotic (Θ) bounds for the solution to the following recurrences by using the recursion-tree method or the induction method (your choice). You may assume that n is a power of 10 in (a), or a power of 3 in (b). Show your work.

   (a) [*8 marks*]

$$T(n) = \begin{cases} 2\,T(n/10) + \sqrt{n} & \text{if } n > 1 \\ 7 & \text{if } n \leq 1 \end{cases}$$

   (b) [*8 marks*]

$$T(n) = \begin{cases} 10\,T(n/3) + n^2 & \text{if } n > 1 \\ 1 & \text{if } n \leq 1 \end{cases}$$

4. [*6 marks*]

   (a) Solve part (a) of the previous question by the master method.

   (b) Solve part (b) of the previous question by the master method.