CSC236 term test #2

November 2016

Last name:

First name:

utoronto.ca email:

Fill in your name and UofT email **clearly** and wait for the invigilator to announce the start before turning this page. You will receive 20% of any question you leave blank or say that you cannot answer. On the other hand, you will lose marks for false statements, or statements you make without justification.

1. (10 pts) Consider the following recurrence relation:

$$T(n) = \begin{cases} 3 & n = 2 \\ T\left(\left\lceil \frac{n}{2} \right\rceil\right) * T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) & n > 2 \end{cases}$$

Use repeated substitution (aka unwinding) to make a conjecture of a closed-form expression for $T(n)$ in the special case where n is a power of 2 (i.e.,$\exists k > 0 \in \mathbb{N}, n = 2^k$). Then, prove your conjecture is true for n of the form 2^k.

This page is left (mostly) blank, in case you need it.

2. (8 pts) Assume we know that when $n = 2^{(2^k)}$ for some $k \in \mathbb{N}$, $S(n) = \lg \lg n + 3$. Show that $S(n)$ is in $\Omega(\lg \lg n)$ for all $n > 1 \in \mathbb{N}$, not just special cases. **Hint:** you may use

i. S is monotonic non-decreasing

ii. $\forall\, n > 1 \in \mathbb{N}, \exists k \in \mathbb{N}$ such that $\sqrt{2^{(2^k)}} \leq n \leq 2^{(2^k)}$

iii. $\sqrt{2^{(2^k)}} = 2^{(2^{k-1})}$

iv. Since $n = 2^{(2^k)}$, $2^k = \lg n$ and $k = \lg \lg n$

This page is left (mostly) blank, in case you need it.

Note. *This question may take more time than the number of points assigned suggest.*

3.

a) (4 pts) Consider the following algorithm, and prove if the loop iterates at least c times, the following loop invariant holds at end of the c-th iteration.

$$LI(i_c, sum_c): 0 \leq i_c \leq \frac{n}{2}, i_c \in \mathbb{N} \text{ and } sum_c = \sum_{j=0}^{i_c-1} A[2j]$$

Note that sum of an empty list is zero, *i.e.*, $\sum_{j=0}^{-1} A[2j] = 0$.

```
Algorithm avg(A)
pre-: A is a list of real numbers, its index starts from 0 and
its size, n, is 2k, ∃k > 0 ∈ ℕ
post-: return the average of numbers in positions divisible by 2
i = 0
sum = 0
m = length(A)/2
while i < m
    sum = sum + A[2 * i]
    i = i + 1
a = sum / i
return a
```

b) (1 pts) Partial correctness: use the loop invariant above and prove the algorithm is correct, assuming it terminates.

This page is left (mostly) blank, in case you need it.

This page is left (mostly) blank, in case you need it.