UNIVERSITY OF TORONTO
Faculty of Arts and Science

term test #2, Version 2
CSC165H1S

Date: Tuesday November 28, 6:10–7:00pm
Duration: 50 minutes
Instructor(s): Danny Heap
No Aids Allowed

Name:
utorid:
U of T email:

Please read the following guidelines carefully!

• Please write your name on both the front and back of this exam.
• This examination has 4 questions. There are a total of 8 pages, DOUBLE-SIDED.
• Answer questions clearly and completely. Provide justification unless explicitly asked not to.
• All formulas must have negations applied directly to propositional variables or predicates.
• In your proofs, you may always use definitions of predicates from the course. You may not use any external facts about rates of growth, divisibility, primes, or greatest common divisor unless you prove them, or they are given to you in the question.

Take a deep breath.
This is your chance to show us
How much you’ve learned.

Good luck!
1. [5 marks] Induction. Use induction on \( n \) to prove:

\[ \forall n \in \mathbb{N}, n \geq 5 \Rightarrow 2^n > n^2 \]

**Solution**

**Proof (induction on \( n \)):** Define \( P(n) : n \geq 5 \Rightarrow 2^n > n^2 \).

**Base case:** \( 2^5 = 32 > 25 = 5^2 \), which verifies \( P(5) \).

**Inductive step:** Let \( n \in \mathbb{N} \), and assume \( P(n) \). I will show that \( P(n+1) \) follows, that is if \( n + 1 \geq 5 \), then \( 2^{n+1} > (n + 1)^2 \). Assume that \( n + 1 \geq 5 \). Then

\[
2^{n+1} = 2 \times 2^n > 2n^2 \quad \text{(by the inductive hypothesis)}
\]

\[
= n^2 + n^2 \geq n^2 + 5n = n^2 + 2n + 3n \quad \text{(since } n \geq 5) \]

\[
> n^2 + 2n + 1 = (n + 1)^2 \quad \text{(since } 3n \geq 15 \land 15 > 1) \]
2. [5 marks] Properties of Big-Oh. Recall the following definitions:

- For all functions \( f, g : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0} \), we say \( f \in \Omega(g) \) when:
  \[
  \exists c, n_0 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_0 \Rightarrow f(n) \geq cg(n)
  \]

- For any real number \( x \), \( \lfloor x \rfloor \) is the largest integer that is no larger than \( x \), and we may use the following characterization of \( \lfloor x \rfloor \):
  \[
  x - 1 < \lfloor x \rfloor \leq x
  \]

  Define the function \( \lfloor f \rfloor(n) \) as \( \lfloor f(n) \rfloor \).

- Function \( f \) eventually dominates 1 if:
  \[
  \exists n_1 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_1 \Rightarrow f(n) \geq 1
  \]

Use these definitions (you may not use any of the properties of big-Oh from the course notes) to prove that if \( f \in \Omega(g) \) and \( f \) eventually dominates 1, then \( \lfloor f \rfloor \in \Omega(g) \). Begin by writing a statement, in predicate logic, of what you aim to prove.

**Solution**

**Claim:**
\[
\forall f, g : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}, [f \in \Omega(g) \land (\exists n_1 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_1 \Rightarrow f(n) \geq 1)] \Rightarrow \lfloor f \rfloor \in \Omega(g)
\]

**Proof:** Let \( f, g : \mathbb{N} \rightarrow \mathbb{R}_{\geq 0} \). Assume \( f \in \Omega(g) \), that is \( \exists c, n_0 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_0 \Rightarrow f(n) \geq cg(n) \). Let \( c \) and \( n_0 \) be such values. Also assume \( \exists n_1 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_1 \Rightarrow f(n) \geq 1 \), and let \( n_1 \) be such a value. Let \( n_2 = \max(n_0, n_1) \) and \( c_1 = c/2 \). I will show that \( \forall n \in \mathbb{N}, n \geq n_2 \Rightarrow \lfloor f(n) \rfloor \geq c_1 g(n) \).

Let \( n \in \mathbb{N} \) and assume \( n \geq n_2 \). Then

\[
2 \lfloor f(n) \rfloor = \lfloor f(n) \rfloor + \lfloor f(n) \rfloor \geq \lfloor f(n) \rfloor + 1 \quad (f(n) \geq 1 \Rightarrow \lfloor f(n) \rfloor \geq 1 \text{ by definition of } \lfloor x \rfloor)
\]

\[
> f(n) \quad \text{(characterization of } \lfloor x \rfloor) \]

\[
\geq cg(n) \quad \text{(since } n \geq n_0) \]

\[
\lfloor f(n) \rfloor \geq c/2g(n) = c_1 g(n)
\]

\[\blacksquare\]
3. [6 marks] Worst-case runtime

Consider the following algorithm:

```python
def algor(L):
    # assume L is a non-empty list of True and False
    n = len(L)
    verity = L[0]
    un_switch = 0
    for i in range(n):
        # loop 1
        if verity == L[i]:
            un_switch = un_switch + 1
            verity = not L[i]
    for j in range(un_switch * un_switch):
        # loop 2
        for k in range(j):
            # loop 3
            print("boop!")
```

Define \( n = \text{len}(L) \) and \( WC(n) \) as the worst-case runtime function of \( \text{algor} \). You may find the following formula useful:

\[
\sum_{g=0}^{h} g = \frac{h(h+1)}{2}
\]

(a) [4 marks] Find, and prove, a tight upper bound on \( WC(n) \). By “tight” we mean that if you choose \( f \) so that \( WC \in O(f) \) you should be convinced (but no need to prove) that \( WC \in \Omega(f) \) also. Begin by writing a statement, in predicate logic, of what you aim to prove.

**Solution**

Claim: Define \( I_{\text{algor},n} = \{\text{lists of length } n \text{ consisting only of } 0s \text{ and } 1s\} \) and \( RT(x) : \text{“steps to execute algor(x)” for } x \in I_{\text{algor},n} \). Let \( U(n) = 5n^4 \). I will show that \( U(n) \) is an upper bound on \( WC_{\text{algor}}(n) \) by showing that:

\[
\forall n \in \mathbb{N}, \forall x \in I_{\text{algor},n}, RT(x) \leq 5n^4
\]

Proof: Let \( n \in \mathbb{N} \) and \( x \in I_{\text{algor},n} \). Then \( RT(x) \) costs at most:

- 3 steps for lines 3–5,
- 4 steps for lines 6–9 for each \( i \), for \( 4n \) steps altogether (if we count each line as producing 1 step)
- \((n^2 - 1)^2 = n^4 - 2n^2 + 1\) steps for lines 11–13, since \( j < (\text{un_switch} \times \text{un_switch}) \) is at most \( n^2 - 1 \) and \( k \) is never more than \( j \), so the inner loop iterates at most \( n^2 - 1 \) times for each \( j \), and there are no more than \( n^2 - 1 \) values of \( j \).
In total there are no more than (since $n \geq 1$):

$$n^4 - 2n^2 + 1 + 4n + 3 = n^4 - 2n^2 + 4n + 4 \leq n^4 + 4n + 4 \leq 9n^4$$

(b) [2 marks] Describe an input family for alg which whose runtime is in big-Theta of the upper bound from the previous part. Explain your conclusion. No proof is necessary.

Solution

sample solution: Consider the family of lists with alternating Trues and Falses, e.g. $x_n = [\text{True}, \text{False}, \ldots, \text{True}, \text{False}]$. Then un_switch has value $n$ on line 10. Loop 3 takes $j$ steps for each fixed $j$, and $j$ ranges from 0 to $n^2 - 1$, so lines 11–13 perform:

$$\sum_{j=0}^{n^2-1} j = \frac{(n^2 - 1)n^2}{2} = \frac{n^4 - n^2}{2}$$

... steps, which is at least $\frac{n^2}{4}$ provided $n$ is at least two. This is in $\Omega$ of $U(n)$, and the previous part showed it was in big-Oh of $U(n)$. The steps contributed by lines 1–10 need not be considered, since they will not change this $\Omega$ bound.

4. [3 marks] Describe an input family for alg whose runtime is in $O(n)$. Explain your conclusion. No proof is necessary.

Solution

sample solution: Consider the family of lists with only Trues as elements. Then lines 11–13 contribute no steps, since un_switch has value 1 on line 10, so the runtime consists of 3 steps for lines 3–5 and $4n$ steps for lines 6–9, for a total of $4n + 3$ steps, which is in $O(n)$. 
This page is left nearly blank for rough work. If you want work on this page to be marked, please indicate this clearly at the location of the original question.
This page is left nearly blank for rough work. If you want work on this page to be marked, please indicate this clearly at the location of the original question.
Name:

<table>
<thead>
<tr>
<th>Question</th>
<th>Grade</th>
<th>Out of</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Q3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>19</strong></td>
<td></td>
</tr>
</tbody>
</table>

never, ever, write below this line...