University of Toronto
Faculty of Arts and Science

CSC165H1S Midterm 2, Version 2

Date: March 20, 2019 Duration: 75 minutes **Instructor(s)**: David Liu, François Pitt

No Aids Allowed

Name:

Student Number:

- This examination has **4** questions. There are a total of **8 pages, DOUBLE-SIDED**.
- All statements in predicate logic must have negations applied directly to propositional variables or predicates.
- In your proofs, you may always use definitions we have covered in this course. However, you may **not** use any external facts about these definitions unless they are given in the question.
- For algorithm analysis questions, you can jump immediately from an exact step count to an asymptotic bound without proof (e.g., write "the number of steps is $3n + \lceil \log n \rceil$, which is $\Theta(n)$").

Take a deep breath.

This is your chance to show us
how much you've learned.
We **WANT** to give you the credit
that you've earned.
A number does not define you.

Good luck!

Question	Grade	Out of
Q1		7
Q2		5
Q3		6
Q4		9
Total		27

1. **[7 marks] Short answer.** You do **not** need to show your work for any parts of this question.

(a) **[1 mark]** Write down the balanced ternary representation of the decimal number 100. The representation should not have any leading zeros. HINT: $3^3 = 27$, $3^4 = 81$. You do not need any higher powers of 3.

(b) **[1 mark]** Let $n \in \mathbb{Z}^+$. What is the *largest* number that can be expressed by an n-digit binary representation? (Your answer should be in terms of n and can be in the form of a summation.)

(c) **[2 marks]** Let $f(n) = \dfrac{n^2 - n + 7}{4n + 3}$ and $g(n) = 5\sqrt{n}$. For each statement below, check one box to indicate whether the statement is true or false.

	TRUE	FALSE		TRUE	FALSE		TRUE	FALSE
$f(n) \in \mathcal{O}(n)$	☐	☐	$g(n) \in \Omega(n)$	☐	☐	$f(n) \in \Omega(g(n))$	☐	☐
$f(n) \in \Theta(g(n))$	☐	☐	$g(n) \in \Theta(\log_3 n)$	☐	☐	$f(n) + g(n) \in \Theta(f(n))$	☐	☐

(d) **[1 mark]** Consider the following algorithm.

```python
def f(n: int) -> None:
    """Precondition: n >= 0."""
    i = 2
    while i + i < n:
        i = i * i * i
```

Find a formula for i_k, the value of variable i after k iterations (where $k \in \mathbb{N}$).

(e) **[2 marks]** Use your answer from the previous part to find the exact number of iterations this function's loop will run. Use floor or ceiling to ensure that the number of iterations is an integer.

2. **[5 marks]** **Induction.** Prove the following statement using induction.

$$\forall n \in \mathbb{N},\ n \geq 2 \Rightarrow \prod_{i=1}^{n} \frac{2i-1}{2i} \geq \frac{1}{2n}$$

(Recall that $\prod$ is *product notation*, similar to summation except each term is multiplied rather than added.)

3. **[6 marks]** **Asymptotic analysis.** You may refer to the following definitions for this question.

$$\begin{aligned} g \in \mathcal{O}(f): &\quad \exists c, n_0 \in \mathbb{R}^+,\ \forall n \in \mathbb{N},\ n \geq n_0 \Rightarrow g(n) \leq c \cdot f(n) \\ g \in \Omega(f): &\quad \exists c, n_0 \in \mathbb{R}^+,\ \forall n \in \mathbb{N},\ n \geq n_0 \Rightarrow g(n) \geq c \cdot f(n) \\ g \in \Theta(f): &\quad g \in \mathcal{O}(f) \wedge g \in \Omega(f) \end{aligned}$$

Disprove the following statement. Begin by writing its negation; you may, but are not required to, expand the definitions of Big-Oh, Omega, and/or Theta in the negated statement.

$$\exists a \in \mathbb{R}^+,\ an^2 + 1 \in \Theta(n^4)$$

The next page is left blank for rough work and/or to continue your proof.

Use this page for rough work. If you want work on this page to be marked, please indicate this clearly *at the location of the original question.*

4. **[9 marks] Running time analysis.**

   (a) **[3 marks]** Consider the following algorithm.

```python
def f(n: int) -> None:
    """Precondition: n >= 0."""
    for i in range(n, 0, -1):   # Loop 1  (i = n, n-1, ..., 1)
        j = 0
        while j < i * i:        # Loop 2
            print(2 * j)
            j = j + 2
```

Find the **exact total number of iterations of Loop 2** when `f` is run, in terms of its input n. To simplify your calculations, you may ignore floors and ceilings. Use the following formula to simplify any summations you find in your expression (valid for all $m \in \mathbb{N}$):

$$\sum_{i=1}^{m} i^2 = \frac{m(m+1)(2m+1)}{6}$$

Note: make sure to explain your work in English, rather than writing only calculations.

(b) **[6 marks]** Consider the following algorithm, which takes as input a list of integers.

```
def my_alg(lst: List[int]) -> None:
    n = len(lst)
    for i in range(n):                 # Loop 1
        if lst[i] % 2 == 0:
            for j in range(i + 1, n):   # Loop 2
                lst[j] = lst[j] + 1
```

Prove matching upper (Big-Oh) and lower (Omega) bounds on the worst-case running time of `my_alg`. Clearly label which part of your solution is a proof of the upper bound, and which part is a proof of the lower bound. You may use properties of divisibility (e.g., about even and odd numbers) in your analysis without proving them. You may also use the summation formula $\sum_{i=0}^{m} i = \frac{m(m+1)}{2}$.

HINT: when Loop 2 runs, all the elements of `lst` from indexes `i+1` to `n-1` switch from even to odd, or vice versa.

If you need more space, please continue your answer on the next page.

Use this page for rough work. If you want work on this page to be marked, please indicate this clearly *at the location of the original question*.