CSCI 3104 Fall 2021
Instructors: Profs. Grochow and Waggoner

Midterm 2- Standard 16

Due Date ..... TODO
Name ..... **Abeal Sileshi**
Student ID **104326364**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You **may not collaborate with other students**. **Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Discord, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

Problem 1.

- My submission is in my own words and reflects my understanding of the material.
- I have not collaborated with any other person.
- I have not posted to external services including, but not limited to Chegg, Discord, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

Agreed (signature here). I, Abeal Sileshi, agree to the above. □

3 Standard 16- Analyzing Code III: Writing Recurrences

Problem 2. Write down a recurrence for the runtime complexity of this algorithm. Clearly justify your answer. You are **not** being asked to solve the recurrence.

Algorithm 1 Recurrences

```
1: procedure Foo1(Integer n)
2:     if n < 5 then return
3:     Foo1(n/7)
4:     Foo1(n/7)
5:     Foo1(n/7)
6:     Foo1(n/7)
7:     Foo1(n/7)
8:
9:     for i ← 1; i ≤ n; i ← i * 8 do
10:        print (2 * i)
```

Answer.

$$T(n) = \begin{cases} O(1) & : n < 5, \\ 5T(\frac{n}{7}) + \log_8 n & : n \geq 5. \end{cases}$$

If n is less than 5, the program returns. That's why a constant time of 1 is given for it. If the input is greater or equal to 5, the algorithm makes 5 recursive calls to itself with input n/7 - this is where the 5T(n/7) comes from. The non-recursive work is in the loop. We can see the loop runs until upper bound of n and i is multiplied by 8 in each iteration hence giving us $\log_8 n$. □