Self-Assessment Midterm

Read this page and fill in your name, pledge, and email ID now.
Do not open past this page until you have read the instructions and are ready to begin.

Name: _____

UVA Email ID: _____

This self-assessment midterm is intended to help prepare you to do well on the final by giving you experience taking an exam, but without any grade pressure or logistical challenges. This exam will not be graded, but you do need to **upload an image of your exam** by **2:59pm Wednesday, 13 October**. You are encouraged to print out the exam to do on paper, and then just take pictures of your exam for the upload. It is not necessary to typeset your exam (and doing the exam on paper better simulates what you will do for the final).

This exam covers material from Weeks 1–5 of the course, including the assigned reading and video materials. Most of the questions on the exam will cover topics that have been covered in the assigned cohort problems. You should prepare a one-page (letter-size, two-sided) reference sheet for use during the exam, but all other resources are forbidden (no internet, textbook, other humans, magnification instruments, etc.). We expect that students will benefit from thinking about what to put on your reference sheet in preparing for the exam, and you may work with anyone you want (including your cohort-mates and other students in the class) to prepare a reference sheets together.

For this exam, you must **work alone**. You are not permitted to obtain help from people other than asking clarifying questions of the course staff. You are not permitted to provide help to others taking the exam. **You may not use any resources other than your brain and body, the one page of notes you prepared, and a simple writing implement like a pen or pencil.**

Binary Representation

1. [10 points] Better Buses

Recall the UTS buses from Week 1. To celebrate the student Covid positivity rate dropping below 1%, the displays are being upgraded. The new displays are 128 pixels wide and 32 pixels tall. Whereas before pixels could be either orange or black, now each pixel can be in one of three states: orange, green, or black.

Answer these questions about the length of binary strings necessary to represent the bus display.

a) A simple encoding scheme uses two bits to represent each state: `00` is black, `01` is orange, and `11` is green. Assuming all configurations are represented with the same number of bits, what is the number of bits required to represent all configurations using this representation? Justify your answer by demonstrating a bijection between all strings of the length you indicate, and all possible configurations of the display.

b) The above encoding is inefficient. Describe a more efficient encoding and argue that it is the most efficient encoding that can represent all possible configurations with a fixed-length bitstring.

Countability

2. [10 points] It's about to get odd

Prove that the set of the odd natural numbers (i.e., $\{1, 3, 5, 7, \ldots\}$) is *countably infinite*.

3. [10 points] Toes

Prove that the set of all toes on all the feet of all current UVA students is countable.

Uncountability

4. [10 points] Fingers

The Cantorvanian creatures from the planet Cantorvania have only one hand, but it has a countably infinite number of fingers. A human glove has only 5 holes for fingers, so when a Cantorvanian wears one it will put many fingers into the same finger hole. To wear a glove Cantorvanians do not need to put their fingers into the glove's holes contiguously. For example, fingers 4 and 7 may go into hole 2, with finger 5 going into hole 5. To It is ok if some glove holes have no fingers, but all fingers must be in a hole.[1]

Show that there is an uncountable number of ways for a Cantorvanian to wear a human glove.

[1] I don't know about you, but after reading this question "finger" no longer sounds like a word, and I'm strangely aware of the way mine move. Happy Halloween!

Counting Gates

5. [10 points] n-bit XOR

In class we discussed the function $\texttt{XOR} : \{0,1\}^2 \to \{0,1\}$ defined such that $\texttt{XOR}(a,b)$ is 1 provided exactly one of a or b is 1. For this question we have implemented an n-bit XOR function, which we denote

$$\texttt{XOR}_n : \{0,1\}^n \to \{0,1\}$$

for $n \geq 2$. It returns 1 exactly when an odd number of its inputs are 1. Our implementation uses a subroutine for doing XOR with one fewer bit (i.e., it will invoke an implementation of $\texttt{XOR}_{n-1}$), then perform XOR on that resulting bit with the first input bit. We will have a base case of $\texttt{XOR}_2$, which is normal 2-bit XOR.

As a straightline program, $\texttt{XOR}_n$ for all $n > 2$ will be implemented as follows:

```
def XOR_n(x1, ..., xn):
    rest = XOR_n_minus_one(x2, ..., xn)
    return XOR(rest, x1)
```

Show using induction that the number of NAND gates needed to represent a circuit for $\texttt{XOR}_n$ is no more than $5n$ gates (hint: XOR requires 4 NAND gates).

Universality

6. [10 points] XOR, AND, 1

Show that the operations `XOR`, `AND` and `1` together are a universal gate set.

Incremental Universality

7. [10 points] Increment and Add

Consider the *INCADD*-straightline language where programs must be straightline code using these two operations:

```
def INC(a):
    return (a + 1) % 2
```

```
def ADD(a,b):
    return (a + b) % 2
```

The `a` and `b` inputs are a single bit. The `%` operator (as in Java, Python, and Rust) is modulo (remainder after division). So, for example, `(1 + 0) % 2 = 1` and `(1 + 1) % 2 = 0`.

Asymptotics

8. [10 points] Big Omicron

Recall these definitions of asymptotic operators from class:

Definition 1 (O) A function $f : \mathbb{N} \to \mathbb{R}_+$ is in the set $O(g(n))$, defined for any function $g : \mathbb{N} \to \mathbb{R}_+$ if and only if there exist two constants $c \in \mathbb{R}_+, n_0 \in \mathbb{N}$ such that: $\forall n > n_0 . f(n) \leq cg(n)$.

Definition 2 (Ω) A function $f : \mathbb{N} \to \mathbb{R}_+$ is in the set $\Omega(g(n))$, defined for any function $g : \mathbb{N} \to \mathbb{R}_+$ if and only if there exist two constants $c \in \mathbb{R}_+, n_0 \in \mathbb{N}$ such that: $\forall n > n_0 . f(n) \geq cg(n)$.

Definition 3 (Θ) A function $f : \mathbb{N} \to \mathbb{R}_+$ is in the set $\Theta(g(n))$, defined for any function $g : \mathbb{N} \to \mathbb{R}_+$ if and only if $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$.

Let $f(n) = 13n$ and $g(n) = 7n^2$, which of the following are true? Support your answer to each part with a convincing argument that uses the definitions above.

a. $f \in O(g)$

b. $f \in \Omega(g)$

c. $f \in \Theta(g)$

Asymptotics

9. [10 points] Omicron and Agemo

If $f \in O(g)$ is $g \in \Omega(f)$?

Use the definitions (in the previous page) to either prove that for any functions f and g, $f \in O(g) \implies g \in \Omega(f)$, or show that there is at least one case where $f \in O(g)$ but $g \notin \Omega(f)$.