CS200 - Problem Set 4

1. In this class, we learn how to prove certain types of code/algorithms work correctly. However, for some algorithms, like Artificial Intelligence and Machine Learning algorithms, we haven't figured out ways to prove they are always correct, and in fact we know they are often incorrect and tend to be biased in their results. (In other words, they tend to be more incorrect for certain types of inputs than others.) Additionally, even if we can prove an algorithm is correct, it might be biased, and even if it is correct and unbiased it still might not be moral to use such an algorithm. Please read the two articles in Canvas Files in the folder "Bias in Algorithms," and write a brief reflection on the following themes:

    - If a company can't prove that it's algorithm is always correct or always unbiased, is it wrong for the company to deploy that algorithm? When is it OK for a company to deploy such an algorithm?
    - Both humans and algorithms can be biased. Do you think it is better to give decision making power to a biased person or a biased algorithm?
    - Suppose you are working for a company, and they ask you to design an algorithm. Before you start work, do you think it is important to ask questions like: what potential harm can this algorithm do if it is used? What other questions do you think are important to ask before embarking on creating an algorithm?
    - If you think an algorithm could potentially cause harm but also could do good, how would you weight those concerns?
    - Are all algorithms biased?
    - Who do algorithms benefit? Who is excluded from those benefits?
    - Recently, algorithms have been used to help decide prison sentences. Let's assume that all bias has been removed from these algorithms. In that case, is it OK to use such an algorithm? Are there certain applications that you think we should never use algorithms for?

2. In CS302, you might study an algorithm called "Closest Points," which takes as input a set of points on a plane, and returns the pair of points that are closest to each other. The following is a small part of the larger proof of correctness of the algorithm. Suppose you know that no two points are closer than distance δ. Further suppose that you fill the plane with an imaginary grid of squares, where the side of each square is $\delta/2$. Prove using a proof by contradiction that any square in the grid can contain at

most one point. (Note: if a point p has x-coordinate p_x and y-coordinate p_y and a point q has x-coordinate q_x and y-coordinate q_y, then the distance between p and q is $\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$.)

3. Explain what is wrong with *the logic* of each of the following proofs. (You just need to write a sentence or two for each proof describing the logical error. All errors are with logic and structure, not with style or lack of explanation.)

   (a) **[2 points]** DM 2.5.14 (Click on the link in a pdf to go to the textbook section 2.5, then go to problem 14)

   (b) **Proof:**
   Let $P(n)$ be the predicate that you can make n cents of postage using 3-cent and 7-cent stamps. We will prove using induction that $P(n)$ is true for all $n \geq 14$.

   Base case: When $n = 14$, note that you can make 14 cents using two 7-cent stamps.

   Inductive case: Let $k \geq 14$. Suppose $P(k)$ is true. Then $\exists x, y \in \mathbb{Z} : x, y \geq 0$ where x corresponds to the number of 3-cent stamps, and y corresponds to the number of 7-cent stamps) such that

   $$k = 3x + 7y. \tag{1}$$

   Now we can remove two 7-cent stamps and add five 3-cent stamps. Then we have

   $$3(x + 5) + 7(y - 2) = 3x + 15 + 7y - 14 = k + 1. \tag{2}$$

   Thus we can create $k + 1$-cents worth of postage using 7-cent stamps and 3-cent stamps.

   Therefore, by induction $P(n)$ is true for all $n \geq 14$.

   (c) **Proof:**
   Let $P(n)$ be the predicate that the sum of the first n odd numbers is n^2. We will prove $P(n)$ is true for all $n \geq 1$.

   Base case: When $n = 1$, the sum of the first odd number is $1 = 1^2$, so $P(1)$ is true.

   Inductive step: For $k = 2$, we look at $1 + 3$, which equals 2^2. For $k = 3$, we look at $1 + 3 + 5$, which equals 3^2. Continuing in this way, we see that sum of the first k odd numbers is k^2.

   Thus for all $n \geq 1$, $P(n)$ is true.

   (d) **Proof:**
   Let $P(n)$ be the predicate that the sum of the first n numbers is $(n^2 + n)/2$. We will prove $P(n)$ is true for all $n \geq 1$.

   Base case: When $n = 1$, the sum of the first number is 1, which equals $(1^2 + 1)/2$.

Inductive step: Assume $P(k)$ is true for $k \geq 1$. This means

$$P(k) = (k^2 + k)/2. \tag{3}$$

Now if we add $k + 1$ to both sides, we have

$$P(k) + (k + 1) = (k^2 + k)/2 + (k + 1). \tag{4}$$

Doing some math, we find that the right hand side is equal to $((k+1)^2+(k+1))/2$, so $P(k + 1)$ is true.

Thus for all $n \geq 1$, $P(n)$ is true.

(e) **Proof:**
Let $P(n)$ be the predicate that the sum of the first n numbers is $(n^2 + n)/2$. We will prove $P(n)$ is true for all $n \geq 1$.

Base case: When $n = 1$, the sum of the first number is 1, which equals $(1^2 + 1)/2$.

Inductive step: Assume $P(k)$ is true for $k \geq 1$. Then if we consider $P(k + 1)$, we have

$$1 + 2 + \cdots + k + 1 = ((k + 1)^2 + (k + 1))/2. \tag{5}$$

Now if we subtract $k + 1$ from both sides and do some algebra, we have

$$1 + 2 + \cdots + k = (k^2 + k)/2, \tag{6}$$

which is $P(k)$, which we know is true.

Thus for all $n \geq 1$, $P(n)$ is true.

(f) **[Challenge]** Explain what is wrong with the following inductive proof that all Middlebury students have the same eye color. I find it easiest to describe the issue by using the "ladder" analogy from class.

Proof: Let $P(n)$ be the predicate that any group of n Middlebury students have the same eye color. We will prove $P(n)$ is true for all $n \geq 1$.

Base case: $P(1)$ is true because any one Middlebury student has the same eye color as themselves.

Inductive case: Let $k \geq 1$. Assume for induction that any set of k Middlebury students have the same eye color. Now let's consider any set of $k + 1$ Middlebury students. If we look at the first k of those $k + 1$ students, by our inductive assumption they must all have the same eye color. However, if we look at the last k of those $k + 1$ students, by our inductive assumption, they must also all have the same eye color. Now the second student must be part of the first set of k and the last set of k, so all $k + 1$ students must have the same eye color as this second student. Thus, any set of $k + 1$ Middlebury students have the same eye color, so $P(k + 1)$ is true.

Therefore, by induction, $P(n)$ is true for all $n \geq 1$.

(g) Why is it important to be able to find errors in inductive proofs?

4. Using PS3 #5, rewrite the inductive step of PS4 #3b (on this problem set) to make it correct.

5. Prove using induction that $2^n > n^2$ whenever n is an integer, and $n \geq 5$.

6. **[Optional]** I'm not assigning any more mathematical induction proofs on this pset, but if you'd like more practice, there are plenty more in the textbooks!

7. We can use induction to prove a recursive algorithm is correct. However, it involves a couple of stylistic ingredients that are a little different from a mathematical proof, as we saw in class. I will prove `Square` is correct for you, and highlight some important parts. Then you should prove `Log` is correct.

Algorithm 1: `Square`(m)

Input : Non-negative integer m
Output: m^2

```
/* Base Case                                                    */
1 if m equals 0 then
2 |   return 0;
3 else
  |   // Recursive step
4 |   return 2 × m − 1 + Square(m − 1);
5 end
```

Proof: Let $P(n)$ be the predicate that `Square`(n) correctly outputs n^2. We will prove using induction that $P(n)$ is true for all $n \geq 0$. *[NOTE: n is the variables we will use for induction, while m is the variable we will use when referencing the pseudocode. You can use the same variables, but it can get confusing in more complicated proofs.]*

For the base case, let $n = 0$, so we need to analyze `Square`(0). In this case, the `If` statement is true at line 1, and the algorithm returns 0. Since $0 = 0^2$, the output is correct. *[NOTE: I've analyzed what happens in the code in the base case, and compared to what I wanted the output to be. I reference line numbers to make it clear what I'm talking about.]*

For the inductive step, let $k \geq 0$ and assume $P(k)$ is true. Now let's consider what happens when we run `Square`(k+1). Since $k \geq 0$, then $k + 1 \geq 1$, so the algorithm does not trigger the base case but goes to the `else` case in line 4. *[NOTE: You need to explain why the algorithm does not trigger the base case in the code]* and returns $2(k+1)-1+$`Square`$((k+1)-1) = 2(k+1)-1+$`Square`(k). By inductive assumption, `Square`(k) returns k^2. *[NOTE: This is the critical inductive step! You need to replace the output of the recursive call with whatever you can assume the output of the function should be by inductive assumption. Then once you have done this replacement, you normally need to do some math, as follows:]* Using this assumption, we have that at

line 4 the algorithm returns

$$2(k+1) - 1 + k^2 = k^2 + 2k + 1 = (k+1)^2. \tag{7}$$

This is the correct output for input $m = k+1$, so $P(k+1)$ is true.

Therefore, by induction, $P(n)$ is true for all $n \geq 0$.

Prove the following algorithm evaluates the log (base 2) of the input. Depending on your background, you may need to learn or review logarithms and exponentials. Khan Academy has some useful videos, or there are many other online resources that have basic info on logarithms and exponentials. Think about what your induction variable "n" is, before starting.

Algorithm 2: `Log`(m)

Input : Positive integer m of the form 2^g
Output: g

```
/* Base Case                                                    */
1 if m equals 1 then
2 |   return 0;
3 else
  |   // Recursive step
4 |   return 1 + Log(m/2);
5 end
```

8. How long did you spend on this homework?