Homework 5: Number Theory and Induction

Due date: Wednesday February 8th at 10 PM
If you work with others (and you should!), remember to follow the collaboration policy outlined in the syllabus.
In general, you are graded on both the clarity and accuracy of your work. Your solution should be clear enough that someone in the class who had not seen the problem before would understand it.
We sometimes describe approximately how long our explanations are. These are intended to help you understand approximately how much detail we are expecting. You are allowed to have longer explanations, but explanations significantly longer than necessary may receive deductions.

Be sure to read the grading guidelines on the assignments page for more information on what we're looking for.

In order to assist with the transition from formal proofs to English proofs, we've published a style guide on the website containing some tips. This guide contains references to proof materials that we haven't taught yet, so don't worry if some of these terms are unfamiliar.

This homework comes in two parts. Part one is practice with modular arithmetic; part two is practice with induction.

We will have two separate gradescope submission boxes. Using one late day allows you to submit **both** parts one day later (e.g. one late day lets you submit both parts on Thursday May 5).

The staff will focus on grading part 2 first. If you don't use any late days, we will get you feedback on part two before the midterm ends (we want to be really sure you get feedback on at least one induction problem in time). We will likely not get the part 1 feedback returned before the midterm ends.

Part I

1. Backwards Proofs

(a) Do the concept check for this week on gradescope.

(b) On the concept check, problem 4 is an incorrect proof that $3|(9^2 - 4^2)$, which is the base case in an induction proof of $3|(9^n - 4^n)$ for integers $n \geq 2$. Write a correct proof of **only** the base case (this should be very short).

2. Like -3 but better! [16 points]

(a) In normal arithmetic, $a+3+(-3) = a$ for any integer a. So we say that -3 "undoes" 3. In modular arithmetic, a similar statement might be that $a+3+2 \equiv a \pmod 5$, so 2 "undoes 3 for $\pmod n$ addition." More generally, we say that for any integer n (where $n > 3$), an integer b "undoes 3 for $\pmod n$ addition" if and only if for all integers a, $a + 3 + b \equiv a \pmod n$.

Show that for any integer n (where $n > 3$), there exists some integer b, where $1 \leq b \leq n$, which undoes 3 for $\pmod n$ addition. [8 points]

(b) In this problem, you'll show that for any integer n (where $n > 3$), for all integers b, b' where both b and b' undo 3 for $\pmod n$ addition, that $b \equiv b' \pmod n$. Note that we've gotten rid of the $1 \leq b \leq n$ requirement in this part! [8 points]

- Write the statement above in predicate logic. Use the predicate $\texttt{Undoes3}(b, n)$ for "b undoes 3 for $\pmod n$ arithmetic."
- Now write an English proof of the statement.

(c) For similar concepts in modular arithmetic, people will say things like "There is a unique number that undoes 3 mod n." Ponder why this use of "unique" makes sense, but also why this is a little different from the example of "unique" we saw in class. You do not have to write anything for this part [0 points]

3. GCD proof [12 points]

Let x, y, z be arbitrary integers such that $x|y$ and $y|z$. For the following questions, if the statement is true, write a proof. If it is false, disprove it (you will provide a counterexample in that case).

(a) Is it true that $xy|z$?

(b) Is it true that $x|z$?

Extra Credit: Exponentially increasing fun [0 point]

Since $a \% n \equiv a \pmod{n}$, we know that we can reduce the base of an exponent in $\pmod{n}$ arithmetic. That is:

$$a^k \equiv (a \% n)^k \pmod{n}.$$

But the same is **not** true of the exponent! That is, we cannot say that $a^k \equiv a^{k\%n} \pmod{n}$. Consider, for instance, that $2^{10} \% 3 = 1$ but $2^{10\%3} \% 3 = 2^1 \% 3 = 2$. The correct way to simplify exponents is quite a bit more subtle. In this problem you'll prove it in steps.

For these proofs you may use any theorem on the number theory reference sheet, even the ones we haven't proven yet in class.

(a) Let $R = \{t \in \mathbb{Z} : 1 \leq t \leq n-1 \wedge \gcd(t, n) = 1\}$. Define the set $aR = \{ax \% n : x \in R\}$. Prove that $aR = R$ for every integer $a > 0$ with $\gcd(a, n) = 1$.

(b) Consider the product of all elements in R (taken $\% n$) and consider the product of all the elements in aR (again, taken $\% n$). By comparing these two expressions, conclude that for all $a \in R$ we have $a^{\varphi(n)} \equiv 1 \pmod{n}$ where $\varphi(n) = |R|$.

(c) Use the previous part to show that for any $b \geq 0$ and $a \in R$ we have $a^b \equiv a^{b\%\varphi(n)} \pmod{n}$.

(d) Now suppose that $y = x^e \pmod{n}$ for some x with $\gcd(x, n) = 1$ and e some integer ≥ 0 such that $\gcd(e, \varphi(n)) = 1$. Let $d = e^{-1} \pmod{\varphi(n)}$. Prove that $y^d \equiv x \pmod{n}$.

(e) Prove the following two facts about φ: First, if p is prime then $\varphi(p) = p - 1$. Second, for any positive integers a and b with $\gcd(a, b) = 1$, we have $\varphi(ab) = \varphi(a)\varphi(b)$.

These facts together are the basis for the most-widely used "public key encryption system." One chooses $n = pq$ for large primes p and q, and a value of e. The numbers n and e are made public to anyone who wants to send a message securely. To send a message x, the sender computes $y = x^e \% n$ and sends y (the "encrypted text"). To decrypt, one computes $y^d \% n$ (note that the recipient must be the one who chose p, q so they can calculate d). The security of the system relies on it being hard to compute d from just e and m.

Part II

4. Induction Divides[20 points]

Prove that $4 \mid (9^n - 1)$ for all $n \in \mathbb{N}$, by induction on n.

Hint: In your inductive step, you'll need to be creative to apply your inductive hypothesis. Focus on forcing the right expression to appear.

5. Induction Code [20 points]

Consider the following code snippet.

```
public int Mystery(int n){
    if(n < 0)
        throw new InvalidInputException();
    if(n == 0)
        return 30;
    if(n == 1)
        return 33;
    return Mystery(n - 1) + 2 * Mystery(n - 2);
}
```

In this problem, we will use `Mystery(n)` to refer to the value returned by the code snippet above when run on input `n`.
For example, `Mystery(2)` $= 33 + 2 \cdot 30 = 93$.

Use induction to show that `Mystery(n)`$= 21 \cdot 2^n + 9 \cdot (-1)^n$ for all integers $n \geq 0$.

6. Feedback [1 point]

Please keep track of how much time you spend on this homework and answer the following questions. This can help us calibrate future assignments and future iterations of the course, and can help you identify which areas are most challenging for you.

- How many hours did you spend working on this assignment (excluding any extra credit questions, if applicable)? Report your estimate to the nearest hour.
- Which problem did you spend the most time on?
- Any other feedback for us?