CS103 Handout 06
Spring 2015 April 3, 2015

Problem Set 1

Here we are – it's the first problem set of the quarter! This problem set is designed to give you practice writing proofs on a variety of different topics like set theory, binary operators, puzzles, games, and even logic itself. We hope that this problem set gives you a sense of what it's like to work with proof-based mathematics and solidifies your understanding of set theory.

Before you start this problem set, please do the following:

- Review Handout #03, "How to Succeed in CS103," for advice about how to approach the problem sets in CS103. In particular, ***make sure to start early!***
- Review Handout #04, "Problem Set Policies," to make sure you understand our late policy and how to submit your work.
- Review Handout #05, "CS103 and the Stanford Honor Code," to make sure you understand our collaboration and citation policies.

As always, please feel free to drop by office hours, post on Piazza, or send us emails if you have any questions. We'd be happy to help out.

This problem set has 32 possible points. It is weighted at 4% of your total grade.

Good luck, and have fun!

Checkpoint Questions Due Monday, April 6 at 12:50 PM
Remaining Questions Due Friday, April 10 at 12:50 PM

Write your solutions to the following checkpoint problem and submit them through Scoryst by Monday, April 3 at 12:50PM. These problems will be graded on a 0 / 1 / 2 scale as follows:

- Solutions that reasonably attempt to solve all of the problems, even if the attempts are incorrect, will receive two points.
- Solutions that reasonably attempt some but not all of the problems will receive one point.
- Solutions that do not reasonably attempt any of the problems – or solutions that are submitted after the deadline – will receive zero points.

Essentially, if you've made a good, honest effort to solve all of the problems and you submit on time, you should receive two points even if your solutions contain errors.

Please make the best effort you can when solving these problems. We want the feedback we give you on your solutions to be as useful as possible, so the more time and effort you put into them, the better we'll be able to comment on your proof style and technique. We will try to get these problems returned to you with feedback on your proof style by Wednesday, April 8. Submission instructions are included in the "Problem Set Policies" handout.

Checkpoint Problem: Multiples of Three (2 Points)

In class, we talked a fair amount about odd and even numbers. This question generalizes the idea of "even" and "odd" to similar terms that arise when dividing by three.

An integer is called a *multiple of three* if it can be written as $3k$ for some integer k. An integer is *congruent to one modulo three* if it can be written as $3k + 1$ for some integer k, and an integer is *congruent to two modulo three* if it can be written as $3k + 2$ for some integer k. For each integer n, exactly one of the following is true (you don't need to prove this):

- n is a multiple of three.
- n is congruent to one modulo three.
- n is congruent to two modulo three.

Now, suppose that we want to prove this result:

For every integer n, n is a multiple of three if and only if n^2 is a multiple of three.

To do this, we will prove the following two statements:

For any integer n, if n a multiple of three, then n^2 is a multiple of three.

For any integer n, if n^2 is a multiple of three, then n is a multiple of three.

i. Prove the first of these statements with a direct proof.

ii. Prove the second of these statements using a proof by contrapositive. Make sure that you state the contrapositive of the statement explicitly before you attempt to prove it.

iii. Prove, by contradiction, that $\sqrt{3}$ is irrational. Make sure that you explicitly state what assumption you are making before you derive a contradiction from it. Recall from lecture that a rational number is one that can be written as p / q for integers p and q where $q \neq 0$ and p and q have no common divisor other than ± 1.

The remainder of these problems should be completed and submitted online by Friday, April 10 at the start of class.

Problem One: Set Theory Warmup (2 Points)

This question is designed to help you get used to the notation and mathematical conventions surrounding sets. We strongly suggest working through this problem and checking your answers before starting Problem Two.

Consider the following sets:

$A = \{ 1, 2, 3, 4 \}$

$B = \{ 2, 2, 2, 1, 4, 3 \}$

$C = \{ 1, \{2\}, \{\{3, 4\}\} \}$

$D = \{ 1, 3 \}$

$E = \mathbb{N}$

$F = \{ \mathbb{N} \}$

Answer each of the following questions and briefly justify your answers. No formal proofs are necessary.

i. Which pairs of the above sets, if any, are equal to one another?

ii. Is $D \in A$? Is $D \subseteq A$?

iii. Is $D \in \wp(A)$? Is $D \subseteq \wp(A)$?

iv. What is $A \cap C$? How about $A \cup C$? How about $A \Delta C$?

v. What is $|B|$? What is $|E|$? What is $|F|$?

Problem Two: Much Ado About Nothing (2 Points)

It can take a bit of practice to get used to the empty set. This problem will ask you to think about a few different sets related to Ø.

Answer each of the following questions. No justification is necessary.

i. What is $\emptyset \cup \{\emptyset\}$? How about $\emptyset \cap \{\emptyset\}$?

ii. What is $\{\emptyset\} \cup \{\{\emptyset\}\}$? How about $\{\emptyset\} \cap \{\{\emptyset\}\}$?

iii. What is $|\{\emptyset, \{\emptyset\}\}|$?

iv. Let $S = \{\{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$. What is $\wp(S)$?

Problem Three: Properties of Sets (4 Points)

The basic operations from set theory can be combined together in lots of different ways. This question asks you to look at various claims about sets and to determine whether they're correct or not.

Determine whether the following claims about sets are true or false. If they're true, prove them. If they're false, disprove them (that is, write out their negation, then prove the negation.)

i. For all sets A, B, and C, if $A \in B$ and $B \in C$, then $A \in C$.

ii. For all sets A and B, if $\wp(A) = \wp(B)$, then $A = B$.

iii. For all sets A, B, and C, if $A \cup C = B \cup C$, then $A = B$.

iv. There exists a set A where $\wp(A) = \{A\}$.

Problem Four: Two Is Irrational? (2 Points)

In lecture, we proved that $\sqrt{2}$ is irrational, and in the checkpoint problem you proved that $\sqrt{3}$ is irrational. Below is a purported proof that $\sqrt{4}$ is irrational:

***Theorem*:** $\sqrt{4}$ is irrational.

Proof: Assume for the sake of contradiction that $\sqrt{4}$ is rational. Then there must exist integers p and q where $q \neq 0$, where $p / q = \sqrt{4}$, and where p and q have no common factors other than 1 and -1.

Starting with $p / q = \sqrt{4}$ and squaring both sides tells us that $p^2 / q^2 = 4$. We can then cross-multiply by q^2 to see that $p^2 = 4q^2$. Since q^2 is an integer and $p^2 = 4q^2$, we see that p^2 is a multiple of four, and therefore that p is a multiple of four. This tells us that $p = 4n$ for some integer n.

Since $4q^2 = p^2$ and $p = 4n$, we can use some algebraic substitutions to show that $4q^2 = (4n)^2 = 16n^2$, so $q^2 = 4n^2$. Since n^2 is an integer and $q^2 = 4n^2$, we see that q^2 is a multiple of four, so q is a multiple of four as well. But since both p and q are multiples of four, we see that p and q share a common divisor other than ±1, contradicting our initial assumption. We have reached a contradiction, so our assumption must have been incorrect. Thus $\sqrt{4}$ is irrational. ■

This proof has to be wrong, because $\sqrt{4} = 2 = {}^2/_1$, so it is indeed rational!

What error does this proof make that lets it conclude $\sqrt{4}$ is irrational? Why doesn't this error occur in the similar proofs that $\sqrt{2}$ and $\sqrt{3}$ are irrational?

Problem Five: Modular Arithmetic (4 Points)

Many different numbers yield the same remainder when divided by some number. For example, the numbers 2, 5, 8, 11, 14, and 17, all leave a remainder of two when divided by three, while the numbers 1, 12, 23, 34, and 45 all leave a remainder of one when divided by eleven. To formalize this relationship between numbers, we'll introduce a relation $\equiv_k$ that, intuitively, indicates that two numbers leave the same remainder when divided by k. For example, we'd say that $1 \equiv_{11} 12$ and that $8 \equiv_3 11$.

To be more rigorous, we'll formally define $\equiv_k$. For any integer k, define $a \equiv_k b$ as follows:

We say that $a \equiv_k b$ if there exists an integer q such that $a - b = kq$

For example, $7 \equiv_3 4$, because $7 - 4 = 3 = 3 \cdot 1$, and $13 \equiv_4 5$ because $13 - 5 = 8 = 4 \cdot 2$. If $x \equiv_k y$, we say that *x is congruent to y modulo k*, hence the terminology in the checkpoint problem. In this problem, you will prove several properties of modular congruence.

i. Prove that for any integer x and any integer k that $x \equiv_k x$.

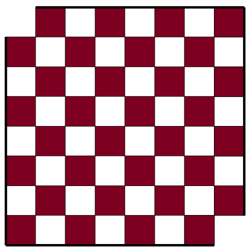
ii. Prove that for any integers x and y and any integer k that if $x \equiv_k y$, then $y \equiv_k x$.

iii. Prove that for any integers x, y, and z and any integer k that if $x \equiv_k y$ and $y \equiv_k z$, then $x \equiv_k z$.

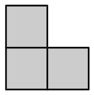
The three properties you have just proven show that modular congruence is an *equivalence relation*. Equivalence relations are important throughout mathematics, and we'll see more examples of them later in the quarter.

Problem Six: Tiling a Chessboard (4 Points)

Suppose you have a standard 8×8 chessboard with two opposite corners removed, as shown here:



In the course notes (pages 62 - 63), there's a proof that it's impossible to tile this chessboard using 2×1 dominoes. This question considers what happens if you try to tile the chessboard using *right triominoes*, L-shaped tiles that look like this:



i. Prove that it is impossible to tile an 8×8 chessboard missing two opposite corners with right triominoes.

ii. For $n \geq 3$, is it *ever* possible to tile an $n \times n$ chessboard missing two opposite corners with right triominoes? If so, find a number $n \geq 3$ such that it's possible and show how to tile that chessboard with right triominoes. If not, prove that for every $n \geq 3$, it's impossible to tile an $n \times n$ chessboard missing two opposite corners with right triominoes.

Problem Seven: Malleable Encryption (4 Points)

At the tail end of Wednesday's lecture on XOR, we talked about *XOR encryption*, a way of using the XOR operator to share secret information. As a refresher, XOR encryption works as follows. Two people (traditionally, named Alice and Bob) want to share a secret piece of information M with one another. Let's assume that M is a bitstring of length n. In advance, Alice and Bob meet in private and share a secret, randomly-generated encryption key K with one another (K is itself also a bitstring of length n). When Alice decides to send the secret message M to Bob, she computes $M \oplus K$ by XORing each bit of M with the corresponding bit of K. When Bob receives $M \oplus K$, he then computes the value $(M \oplus K) \oplus K = M$ and can read the message.

As mentioned in class, the string $M \oplus K$ appears to be truly random, so anyone who saw the bitstring $M \oplus K$ and didn't know the secret encryption key K would have absolutely no way of learning anything about M or K. Unfortunately, while the system guarantees *secrecy* (no one can read the message without the key), it doesn't guarantee *integrity*. Specifically, it's possible to tamper with the encrypted message so that the recipient receives the wrong information.

Let's imagine that some third person (we'll call her Eve, as is tradition) wants to interfere with the communications between Alice and Bob. We'll assume that Eve can grab any message that Alice sends to Bob and modify it before Bob receives it, and can do so without Bob knowing that anything unusual has happened.

Suppose Eve knows for sure that Alice is about to send Bob exactly one of two possible messages, which we'll call M_1 and M_2. Eve knows what these two possible messages are, but doesn't know which one Alice is going to send and doesn't know the secret key K that Alice and Bob are using to communicate. Eve wants to interfere with the communication so that Bob receives the opposite message that Alice intended. That is, if Alice sends M_1 to Bob, Eve wants to make it seem as though Alice sent M_2, and if Alice sends M_2 to Bob, Eve wants to make it seem as though Alice sent M_1. Amazingly, even though Eve has no way of knowing which message Alice sent to Bob and doesn't know the encryption key K, she can still tamper with the encrypted message to accomplish this.

i. Describe a procedure Eve can follow that will make it seem as though Alice sent the opposite of the intended message to Bob. Remember that Eve doesn't know which of the two messages M_1 and M_2 Alice chose to send to Bob, doesn't know the encryption key K, and (since XOR encryption guarantees secrecy) has no way to learn either of these pieces of information. She only knows that Alice definitely sent one of M_1 and M_2.

ii. Prove that the procedure that you developed in part (ii) works correctly – namely, that Bob will believe Alice sent the opposite of the message she actually sent.

In practice, to prevent message tampering from going undetected, messages transmitted over a network include a *message authentication code* (or *MAC*) that makes it extremely difficult for tampering to occur. If you're curious about how this works, take CS255!

Problem Eight: Yablo's Paradox (4 Points)

A *logical paradox* is a statement that results in a contradiction regardless of whether it's true or false. One of the simplest paradoxes is the *Liar's paradox*, which is the following:

This statement is false.

If this statement is true, then by its own admission, it must be false – a contradiction. On the other hand, if the statement is false, then it must be true – a contradiction. Since this statement results in a contradiction regardless of whether it's true or false, it's a paradox.

Paradoxes often arise as a result of *self-reference*. In the Liar's Paradox, the paradox arises because the statement directly refers to itself. However, it's not the only paradox that can arise from self-reference. This problem explores a paradox called *Yablo's paradox* that gives rises to many paradoxes, each of which arises from *indirect* self-reference.

Consider the following collection of infinitely many statements numbered S_0, S_1, S_2, …, where there is a statement S_n for each natural number n. These statements are ordered in a list as follows:

(S_0): All statements in this list after this one are false.
(S_1): All statements in this list after this one are false.
(S_2): All statements in this list after this one are false.
...

Interestingly, the interplay between these statements makes every statement in the list a paradox.

i. Prove that *every* statement in this list is a paradox.

Now, consider the following modification to this paradox. Instead of having infinitely many statements, suppose that there are "only" 10,000,000,001 of them. Specifically, suppose we have these statements:

(T_0): All statements in this list after this one are false.
(T_1): All statements in this list after this one are false.
(T_2): All statements in this list after this one are false.
...
($T_{10,000,000,000}$): All statements in this list after this one are false.

There's still a lot of statements here, but not infinitely many of them. Interestingly, these statements are all perfectly consistent with one another and do not result in any paradoxes.

ii. For each statement in the above list, determine whether it's true or false and explain why your choices are consistent with one another.

Most logical paradoxes arise because of self-reference. In the Liar's paradox, there's immediate self-reference: the statement says that it's false. Yablo's paradox also contains self-reference: each statement talks about statements "after this one," which means that the statements refer to their own properties. Going forward, don't worry about paradoxical statements in CS103. We won't talk about any more self-referential statements without first warning you that they're self-referential.

Problem Nine: Symmetric Latin Squares (4 Points)

A *Latin square* is an $n \times n$ grid filled with the numbers 1, 2, 3, …, n such that every number appears in every row and every column exactly once. For example, the following are Latin squares:

1	2	3
3	1	2
2	3	1

4	2	1	3
1	3	2	4
3	1	4	2
2	4	3	1

1	3	5	2	4
2	4	1	3	5
3	5	2	4	1
4	1	3	5	2
5	2	4	1	3

A *symmetric Latin square* is a Latin square that is symmetric across the main diagonal. That is, the elements at positions (i, j) and (j, i) are always the same. For example:

1	2	3
2	3	1
3	1	2

4	2	3	1
2	3	1	4
3	1	4	2
1	4	2	3

1	2	3	4	5
2	4	5	3	1
3	5	2	1	4
4	3	1	5	2
5	1	4	2	3

Prove that in any $n \times n$ symmetric Latin square where n is odd, every number 1, 2, 3, …, n must appear at least once on the diagonal from the upper-left corner to the lower-right corner. As a hint, split the Latin square into three regions – the main diagonal and the two regions above and below the main diagonal.

Extra Credit Problem: The Mouse and the Cheese (1 Point Extra Credit)[*]

On each problem set, we'll provide an optional extra credit problem. When we compute final grades at the end of the quarter, we compute the grading curve without any extra credit factored in, then recompute grades a second time to factor in extra credit. This way, you're not at any disadvantage if you decide not to work through these problems. If you do complete the extra credit problems, you may get a slight boost to your overall grade.

As a matter of course policy, we don't provide any hints on the extra credit problems – after all, they're supposed to be challenge problems! However, we're happy to chat about them after the problem sets come due.

Suppose that you have a 3” × 3” × 3” cube of cheese subdivided into twenty-seven 1” × 1” × 1” smaller cubes of cheese. A mouse wants to eat the entire cube of cheese and does so as follows: she first picks any small cube to eat first, then moves to an adjacent small cube of cheese (i.e. a cube that shared a face with the cube that was just eaten) to eat next, then repeats this process.

Prove that the mouse can't eat the centermost cube of cheese last. (A note: for the purposes of this problem, a computer program doesn't count as a proof. ☺)

[*] Adapted from Problem 4E of *A Course in Combinatorics, Second Edition* by Lint and Wilson.