In-Class Problems Week 5, Wed.

Problem 1. (a) Several students felt the proof of Lemma 7.1.7 was worrisome, if not circular. What do you think?

Lemma 7.1.7. Let A be a set and $b \notin A$. If A is infinite, then there is a bijection from $A \cup \{b\}$ to A.

Proof. Here's how to define the bijection: since A is infinite, it certainly has at least one element; call it a_0. But since A is infinite, it has at least two elements, and one of them must not be equal to a_0; call this new element a_1. But since A is infinite, it has at least three elements, one of which must not equal a_0 or a_1; call this new element a_2. Continuing in the way, we conclude that there is an infinite sequence $a_0, a_1, a_2, \ldots, a_n, \ldots$ of different elements of A. Now we can define a bijection $f : A \cup \{b\} \to A$:

$$\begin{aligned} f(b) &::= a_0, \\ f(a_n) &::= a_{n+1} && \text{for } n \in \mathbb{N}, \\ f(a) &::= a && \text{for } a \in A - \{a_0, a_1, \ldots\}. \end{aligned}$$

■

(b) Use the proof of Lemma 7.1.7 to show that if A is an infinite set, then A surj $\mathbb{N}$, that is, every infinite set is "as big as" the set of nonnegative integers.

Problem 2.
Prove that if there is a surjective function ([≤ 1 out, ≥ 1 in] mapping) $f : \mathbb{N} \to S$, then S is countable.

Hint: A Computer Science proof involves filtering for duplicates.

Problem 3.
The rational numbers fill the space between integers, so a first thought is that there must be more of them than the integers, but it's not true. In this problem you'll show that there are the same number of positive rationals as positive integers. That is, the positive rationals are countable.

(a) Define a bijection between the set, $\mathbb{Z}^+$, of positive integers, and the set, $(\mathbb{Z}^+ \times \mathbb{Z}^+)$, of all pairs of positive integers:

$$\begin{array}{l} (1,1), (1,2), (1,3), (1,4), (1,5), \ldots \\ (2,1), (2,2), (2,3), (2,4), (2,5), \ldots \\ (3,1), (3,2), (3,3), (3,4), (3,5), \ldots \\ (4,1), (4,2), (4,3), (4,4), (4,5), \ldots \\ (5,1), (5,2), (5,3), (5,4), (5,5), \ldots \\ \vdots \end{array}$$

(b) Conclude that the set, $\mathbb{Q}^+$, of all positive rational numbers is countable.

Hint: Use Problem 2.

Problem 4.

Let's refer to a programming procedure (written in your favorite programming language —C++, or Java, or Python, ...) as a *string procedure* when it is applicable to data of type **string** and only returns values of type **boolean**. When a string procedure, P, applied to a **string**, s, returns **True**, we'll say that P *recognizes* s. If $\mathcal{R}$ is the set of strings that P recognizes, we'll call P a *recognizer* for $\mathcal{R}$.

(a) Describe how a recognizer would work for the set of strings containing only lowercase Roman letters —`a,b,...,z` —such that each letter occurs twice in a row. For example, `aaccaabbzz`, is such a string, but `abb`, `00bb`, `AAbb`, and `a` are not. (Even better, actually write a recognizer procedure in your favorite programming language).

A set of strings is called *recognizable* if there is a recognizer procedure for it. So the program you described above proves that the set of strings with doubled letters from part (a) is recognizable.

When you actually program a procedure, you have to type the program text into a computer system. This means that every procedure is described by some string of typed characters. If a string, s, is actually the typed description of some string procedure, let's refer to that procedure as P_s. You can think of P_s as the result of compiling s.[1]

In fact, it will be helpful to associate every string, s, with a procedure, P_s. So if string s is not the typed description of a string procedure, we will define P_s to be some fixed string procedure —say one that always returns **False**; so if s is an ill-formed string, P_s will be a recognizer for the empty set of strings.

The result of this is that we have now defined a total function, f, mapping every string, s, to the set, $f(s)$, of strings recognized by P_s. That is we have a total function,

$$f : \textbf{string} \to \text{pow}(\textbf{string}). \tag{1}$$

(b) Explain why range(f) is the set of all recognizable sets of strings.

This is exactly the set up we need to apply the reasoning behind Russell's Paradox to define a set that is not in the range of f, that is, a set of strings, $\mathcal{N}$, that is *not* recognizable.

(c) Let

$$\mathcal{N} ::= \{s \in \textbf{string} \mid s \notin f(s)\}.$$

Prove that $\mathcal{N}$ is not recognizable.

Hint: Similar to Russell's paradox or the proof of Theorem 7.1.11.

(d) Discuss what the conclusion of part (c) implies about the possibility of writing "program analyzers" that take programs as inputs and analyze their behavior.

[1] The string, s, and the procedure, P_s, have to be distinguished to avoid a type error: you can't apply a string to string. For example, let s be the string that you wrote as your program to answer part (a). Applying s to a string argument, say `aabbccdd`, should throw a type exception; what you need to do is apply the procedure P_s to `aabbccdd`. This should result in a returned value **True**, since `aabbccdd` consists of consecutive pairs of lowercase roman letters.