University of Virginia 16 November 2011
cs1120: Introduction of Computing
Explorations in Language, Logic, and Machines

Class 36: Computability

Upcoming (Remaining!) Schedule

- **Tonight (11:59pm):** Problem Set 7 due
- Help hours: Today 5-6:30pm (Jiamin in Davis Commons)
- **Exam 2 Review Sessions** (Today, 7:30pm and Thursday, 6:30pm in Rice 442)
- My office hours Thursday are 4-5pm (instead of normal morning time)
- **Friday's class: Rice Hall Dedication**
- **Tuesday, 22 November (11:59pm):** PS8, Part 1 (see below)
- **Wednesday, 30 November:** Exam 2 due (will be handed out on **Monday, 21 November**)
- **Monday, 5 December (last class):** PS8, Final Submission due
- **Monday, 12 December (1:00pm):** Final Exam due

Problem Set 8 Part 1

For Option J, see the Problem Set posted on the course site. Part 1 is due (electronic submission only) by Tuesday, 22 November.

For Options C and W, you need to submit Part 1 of Problem Set 8 by sending me an email. If you have partners, the email should contain all partners as cc:'s in the email. The content of the email should clearly describe what you plan to do including:

1. The goal of your project.
2. Who your target audience is.
3. Your plan for completing the project. This should state clearly what the main steps are, and how you intend to achieve them.
4. If you are a team (more than one person), also explain how you will distribute and manage the work amongst all team members.

Computability

Terms to remember:

Procedure: A precise description of a series of **steps that can be followed mechanically**.

Algorithm: A procedure that is guaranteed to always finish.

Solving a Problem: An algorithm that for all possible inputs always produces the correct output.

Computable: A problem is *computable* if there exists an algorithm that solves the problem.

Non-Computable: A problem is **non-***computable* if no algorithm exists that solves the problem.

These definitions are about *existence* - it is not necessary to actually know the algorithm to know a problem is computable, just to argue that it must exist.

HALTING Problem (Python version)

Input: a string representing a Python program

Output: if evaluating the input program would ever finish, output **true**. Otherwise, **false**.

Can we define a procedure **halts(s)** that correctly implements the HALTING problem? (Trick question)

Proving non-existence of *A* by contradiction

1. Show *X* is non-sensical (cannot exist).
2. Show a way to construct _____ using _____.
3. Therefore, *A* must not exist.

What are *X* and *A* for Gödel's Proof:

What are *X* and *A* for Turing's Proof:

Prove the *Prints-37 Problem* is non-computable:

Input: A string, *s*, representing a Python program.

Output: True if *s* will ever print 37; otherwise, **False**.

Is the *Find-a-Proof* problem computable?

Input: A proposition *p* expressed in the PM system.

Output: If a proof of *p* exists in PM, output a proof of p. Otherwise, output **false**.

Is the *Find-a-Short-Proof* problem computable?

Input: A proposition *p* expressed in the PM system and a number *n*.

Output: If a proof of *p* of less than length *n* exists in PM, output a proof of p. Otherwise, output **false**.

Are there more Turing Machines or real numbers?