Lecture 12: Some Interesting Problems Related to Set Theory

Russell's Paradox:
Form set $S$ as the set of all sets that do not contain themselves.
$$S = \{ A | \}$$
Should $S$ contain itself?
If $S \in S$,

If $S \notin S$,

The Barber Puzzle:
The male barber in Zooville shaves all the male inhabitants of Zooville who do not shave themselves. He shaves no one else. Who shaves the barber?

As a statement: "Given the scenario above, the barber shaves himself."

Is this statement true or false?

If the barber shaves himself,

If the barber does not shave himself,

Implication of the Paradox: $S \in S \rightarrow S \notin S$ and $S \notin S \rightarrow S \in S$

To Avoid the Paradox: $S = \{ A |$
Prove $S \notin S \rightarrow S \in S$.
Suppose $S \notin S$.

**Statement:** A sentence which (in our universe) can be judged to be either true or false.

ex. If Loyola is in Buffalo, then cats fly.

ex. If Loyola is in Maryland, then all computers are powered by gerbils.

ex. The sentence in this box is a lie.

**Halting Problem:** Does a program loop forever?

Is it possible to write a program that could analyze any computer program with a certain set of input data for the program and determine whether the program halts (stops) or gets caught in an infinite loop when that data is used?

Is this possible?

ex. Suppose this is the program, P.

```java
x = myScan.nextInt();
while ( x != 0 )
{
    System.out.println (x);
    x = x - 2;
}
```

Let $(P, D)$ represent the program $P$ with input data $D$.

Does $(P, 12)$ halt?

Does $(P, 13)$ halt?

Could we have a program, CheckHalt, so that CheckHalt $(P, D)$ sends back “halts” or “loops”?

Let’s assume that CheckHalt can be written.
(i.e., we can build a program that given another program $P$ and a set of data $D$ will answer “halts” or “loops” to the question "Will $P$ halt with input $D$?")
Then let's build another program, Test. Test will accept a program P as its input and then do three things:

1. Make a copy of the input program P.
2. Call CheckHalt sending it P as the program AND P also as the data. In other words it calls CheckHalt (P, P).
3. When it gets the results of CheckHalt, it acts as follows:
   - If CheckHalt (P, P) answers “loops”, Test stops.
   - If CheckHalt (P, P) answers “halts”, Test enters an infinite loop (doesn't halt).

Now, Test is itself a program, i.e., just a sequence of characters in a file.
We could call Test and send it a copy of itself. In other words call Test (Test).
Then CheckHalt (Test, Test) would be called.

If CheckHalt (Test, Test) answers “loops”, Test stops. (But when CheckHalt answers "loops", it means…

If CheckHalt (Test, Test) answers “halts”, Test enters an infinite loop. (But when CheckHalt answers "halts"…

HUH?