Generalized Continuum hypothesis.

There is no infinite set whose cardinality is between the cardinality of an infinite set and its power set.

The powerset of a set is the set of all subsets.

Recall: powerset of the naturals is not countable.

Changing Axioms?

Goedel:
Any set of axioms is either
inconsistent (can prove false statements) or
incomplete (true statements cannot be proven.)

Concrete example:
Continuum hypothesis: “no cardinatity between reals and naturals.”
Continuum hypothesis not disprovable in ZFC
(Goedel 1940.)

Continuum hypothesis not provable.
(Cohen 1963: only Fields medal in logic)

BTW:
Cantor ..bipolar disorder..
Goedel ..starved himself out of fear of being poisoned..
Russell .. was fine.....but for ...two schizophrenic children..
Dangerous work?

See Logicomix by Doxiaidis, Papadimitriou (was professor here), Papadatos, Di Donna.

Resolution of hypothesis?

Gödel. 1940.
Can’t use math!
If math doesn’t contain a contradiction.

This statement is a lie.

Is the statement above true?

The barber shaves every person who does not shave themselves.

Who shaves the barber?

Self reference.

Can a program refer to a program?

Can a program refer to itself?

Uh oh....

Is it actually useful?

Write me a program checker!

Check that the compiler works!

How about.. Check that the compiler terminates on a certain input.

$HALT(P, I)$
  P - program
  I - input.

Determines if $P(I)$ (P run on I) halts or loops forever.

Notice:
Need a computer
...with the notion of a stored program!!!!
(not an adding machine! not a person and an adding machine.)

Program is a text string.
Text string can be an input to a program.
Program can be an input to a program.

Russell’s Paradox.

Naive Set Theory: Any definable collection is a set.

$$\exists y \forall x (x \in y \iff P(x)) \qquad (1)$$

y is the set of elements that satifies the proposition $P(x)$.

$P(x) = x \notin x$. Definable set.

There exists a y that satisfies statement 1 for $P(\cdot)$.

Take $x = y$.

$$y \in y \iff y \notin y.$$

Oops! Not Definable.

What type of object is a set that contain sets?

Axioms changed.

Implementing HALT.

$HALT(P, I)$
  P - program
  I - input.

Determines if $P(I)$ (P run on I) halts or loops forever.

Run P on I and check!

How long do you wait?

Something about infinity here, maybe?

Halt does not exist.

$HALT(P, I)$
  P - program
  I - input.

Determines if $P(I)$ (P run on I) halts or loops forever.

Theorem: There is no program HALT.

Proof: Yes! No! Yes! No! No! Yes! No! Yes! .. $\square$

What is he talking about?
(A) He is confused.
(B) Fermat's Theorem.
(C) Diagonalization.
(D) Professor is just strange.

(C). Maybe (D).

Halt and Turing.

Proof: Assume there is a program $HALT(\cdot, \cdot)$.

Turing(P)
1. If HALT(P,P) ="halts", then go into an infinite loop.
2. Otherwise, halt immediately.

Assumption: there is a program HALT.
There is text that "is" the program HALT.
There is text that is the program Turing.
Can run Turing on Turing!

Does Turing(Turing) halt?

Turing(Turing) halts
$\Longrightarrow$ then HALTS(Turing, Turing) $=$ halts
$\Longrightarrow$ Turing(Turing) loops forever.

Turing(Turing) loops forever
$\Longrightarrow$ then HALTS(Turing, Turing) $\neq$ halts
$\Longrightarrow$ Turing(Turing) halts.

Contradiction. Program HALT does not exist! $\square$
Questions?

Another view of proof: diagonalization.

Any program is a fixed length string.
Fixed length strings are enumerable.
Program halts or not any input, which is a string.

	P_1	P_2	P_3	$\cdots$
P_1	H	H	L	$\cdots$
P_2	L	L	H	$\cdots$
P_3	L	H	H	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Halt - diagonal.
Turing - is not Halt.
and is different from every P_i on the diagonal.
Turing is not on list. Turing is not a program.
Turing can be constructed from Halt.
Halt does not exist! $\square$

Proof play by play.

Assumed HALT(P, I) existed.

What is P? Text.
What is I? Text.

What does it mean to have a program HALT(P, I).
  You have *Text* that is the program HALT(P, I).

Have Text that is the program TURING.
Here it is!!
Turing(P)
  1. If HALT(P,P) ="halts", then go into an infinite loop.
  2. Otherwise, halt immediately.

Turing "diagonalizes" on list of program.
It is not a program!!!!
$\Longrightarrow$ HALT is not a program.

Questions?

We are so smart!

Wow, that was easy!

We should be famous!

No computers for Turing!

In Turing's time.

No computers.

Adding machines.
e.g., Babbage (from table of logarithms) 1812.

Concept of program as data wasn't really there.

Turing machine.

A Turing machine.
– an (infinite) tape with characters
– be in a state, and read a character
– move left, right, and/or write a character.

Universal Turing machine
– an interpreter program for a Turing machine
– where the tape could be a description of a ... Turing machine!

Now that's a computer!

Turing: AI, self modifying code, learning...

Turing and computing.

Just a mathematician?

"Wrote" a chess program.

Simulated the program by hand to play chess.

It won! Once anyway.

Involved with computing labs through the 40s.

Church, Gödel and Turing.

Church proved an equivalent theorem. (Previously.)

Used λ calculus....which is... Lisp (Scheme)!!!
.. functional part. Scheme's lambda is calculus's λ!

Programming languages! javascript, ruby, python....

Gödel: Incompleteness theorem.

Any formal system either is inconsistent or incomplete.
Inconsistent: A false sentence can be proven.
Incomplete: There is no proof for some sentence in the system.

Along the way: "built" computers out of arithmetic.
Showed that every mathematical statement corresponds to
.... a natural number! ! ! ! Same cardinality as...Text.
Today:Programs can be written in ascii.

Computing on top of computing...

Computer, assembly code, programming language, browser, html, javascript..

We can't get enough of building more Turing machines.

Undecidable problems.

Does a program, P, print "Hello World"?
How? What is P? Text!!!!!!

Find exit points and add statement: **Print** "Hello World."

Can a set of notched tiles tile the infinite plane?
Proof: simulate a computer. Halts if finite.

Does a set of integer equations have a solution?
Example: " $x^n + y^n = 1$?"
Problem is undecidable.

Be careful!

Is there an integer solution to $x^n + y^n = 1$?
(Diophantine equation.)

The answer is yes or no. This "problem" is not undecidable.

Undecidability for Diophantine set of equations
$\implies$ no program can take any set of integer equations and
always corectly output whether it has an integer solution.

More about Alan Turing.

- Brilliant codebreaker during WWII, helped break German Enigma Code (which probably shortened war by 1 year).
- Seminal paper in numerical analysis: Condition number.
  Math 54 doesn't really work.
  Almost dependent matrices.
- Seminal paper in mathematical biology.
  Person: embryo is blob. Legs, arms, head.... How?
  Fly: blob. Torso becomes striped.
  Developed chemical reaction-diffusion networks that break symmetry.
- Imitation Game.

Turing: personal.

Tragic ending...

- Arrested as a homosexual, (not particularly closeted)
- given choice of prison or (quackish) injections to eliminate sex drive;
- took injections.
- lost security clearance...
- suffered from depression;
- (possibly) suicided with cyanide at age 42 in 1954.
  (A bite from the apple....) accident?
- British Government apologized (2009) and pardoned (2013).

Back to technical..

This statement is a lie. Neither true nor false!

Every person who doesn't shave themselves is shaved by the barber.

Who shaves the barber?

```
def Turing(P):
 if Halts(P,P): while(true): pass
 else:
  return
```

...Text of Halt...

Halt Progam $\implies$ Turing Program. ($P \implies Q$)

Turing("Turing")? Neither halts nor loops! $\implies$ No Turing program.

No Turing Program $\implies$ No halt program. ($\neg Q \implies \neg P$)

Program is text, so we can pass it to itself,
or refer to self.

Summary: decidability.

Computer Programs are an interesting thing.
Like Math.
Formal Systems.

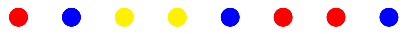
Computer Programs cannot completely "understand" computer programs.

Computation is a lens for other action in the world.

Probability

What's to come? Probability.

A bag contains:



What is the chance that a ball taken from the bag is blue?

Count blue. Count total. Divide.

Next Up: Probability.