The Barber!

The barber shaves all and only people who do not shave themselves.

(A) Barber not Mark. Barber shaves Mark.
(B) Mark shaves the Barber.
(C) Barber doesn't shave themself.
(D) Barber shaves themself.

Its all true. It's all a problem.

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.

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! .. □

Yes! No!...

What is he talking about?

(A) He is confused.
(B) Diagonalization.
(C) Welch-Berlekamp
(D) Professor is just strange.

(B) and (D) maybe? and maybe (A).

Professor does me some love Welch-Berlekamp though!

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

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

Contradiction. Program HALT does not exist! □
Questions?

Another view of proof: diagonalization.

Any program is a fixed length string.
Fixed length strings are enumerable.
Program halts or not on 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! □

Programs?

What are programs?

(A) Instructions.
(B) Text.
(C) Binary String.
(D) They run on computers.

All are correct.

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!!
from fancystuff import halt

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.

Helped Break the enigma code.
The polish machine...the *bomba.*

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.

For now: Counting!

Later: Probability.

The future in this course.

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. **Chances?**

(A) Red Probability is $3/8$
(B) Blue probability is $3/9$
(C) Yellow Probability is $2/8$
(D) Blue probability is $3/8$

Today: Counting!

Outline: basics

1. Counting.
2. Tree
3. Rules of Counting
4. Sample with/without replacement where order does/doesn't matter.

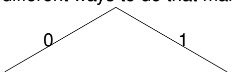
Probability is soon..but first let's count.

Count?

How many outcomes possible for k coin tosses?
How many poker hands?
How many handshakes for n people?
How many diagonals in a n sided convex polygon?
How many 10 digit numbers?
How many 10 digit numbers without repetition?
How many ways can I divide up 5 dollars among 3 people?

Using a tree..

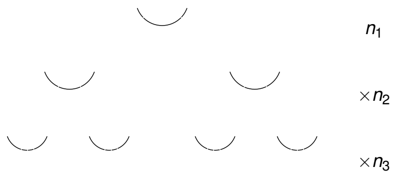
How many 3-bit strings?
How many different sequences of three bits from $\{0,1\}$?
How would you make one sequence?
How many different ways to do that making?



8 leaves which is $2 \times 2 \times 2$. One leaf for each string.
8 3-bit strings!

First Rule of Counting: Product Rule

Objects made by choosing from n_1, then n_2, ..., then n_k
the number of objects is $n_1 \times n_2 \cdots \times n_k$.



In picture, $2 \times 2 \times 3 = 12$!

Poll

Mark whats corect.

(A) $|10 \text{ digit numbers}| = 10^{10}$
(B) $|k \text{ coin tosses}| = 2^k$
(C) $|10 \text{ digit numbers}| = 9 * 10^9$
(D) $|n \text{ digit base } m \text{ numbers}| = m^n$
(E) $|n \text{ digit base } m \text{ numbers}| = (m-1)m^{n-1}$

(A) or (C)? (D) or (E)? (B) are correct.

Using the first rule..

How many outcomes possible for k coin tosses?

2 ways for first choice, 2 ways for second choice, ...
$2 \times 2 \cdots \times 2 = 2^k$

How many 10 digit numbers?

10 ways for first choice, 10 ways for second choice, ...
$10 \times 10 \cdots \times 10 = 10^k$

How many n digit base m numbers?

m ways for first, m ways for second, ...
m^n

(Is 09, a two digit number?)

If no. Then $(m-1)m^{n-1}$.

Functions, polynomials.

How many functions f mapping S to T?

$|T|$ ways to choose for $f(s_1)$, $|T|$ ways to choose for $f(s_2)$, ...
....$|T|^{|S|}$

How many polynomials of degree d modulo p?

p ways to choose for first coefficient, p ways for second, ...
...p^{d+1}

p values for first point, p values for second, ...
...p^{d+1}

Questions?

Permutations.

How many 10 digit numbers **without repeating a digit**?

10 ways for first, 9 ways for second, 8 ways for third, ...

... $10 * 9 * 8 \cdots * 1 = 10!$.[1]

How many different samples of size k from n numbers **without replacement.**

n ways for first choice, $n-1$ ways for second,
$n-2$ choices for third, ...

... $n * (n-1) * (n-2) \cdot * (n-k+1) = \frac{n!}{(n-k)!}$.

How many orderings of n objects are there?
Permutations of n objects.

n ways for first, $n-1$ ways for second,
$n-2$ ways for third, ...

... $n * (n-1) * (n-2) \cdot * 1 = n!$.

[1] By definition: 0! = 1.

One-to-One Functions.

How many one-to-one functions from $|S|$ to $|S|$.

$|S|$ choices for $f(s_1)$, $|S|-1$ choices for $f(s_2)$, ...

So total number is $|S| \times |S|-1 \cdots 1 = |S|!$
A one-to-one function is a permutation!

Counting sets..when order doesn't matter.

How many poker hands?

$52 \times 51 \times 50 \times 49 \times 48$???

Are $A, K, Q, 10, J$ of spades
and $10, J, Q, K, A$ of spades the same?
Second Rule of Counting: If order doesn't matter count ordered objects and then divide by number of orderings.[2]

Number of orderings for a poker hand: "5!"
(The "!" means factorial, not Exclamation.)

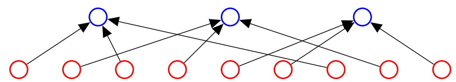
Can write as...
$$\frac{52 \times 51 \times 50 \times 49 \times 48}{5!}$$
$$\frac{52!}{5! \times 47!}$$

Generic: ways to choose 5 out of 52 possibilities.

[2]When each unordered object corresponds equal numbers of ordered objects.

Ordered to unordered.

Second Rule of Counting: If order doesn't matter count ordered objects and then divide by number of orderings.



How many red nodes (ordered objects)? 9.

How many red nodes mapped to one blue node? 3.

How many blue nodes (unordered objects)? $\frac{9}{3} = 3$.

How many poker deals? $52 \cdot 51 \cdot 50 \cdot 49 \cdot 48$.

How many poker deals per hand?
Map each deal to ordered deal: 5!

How many poker hands? $\frac{52 \cdot 51 \cdot 50 \cdot 49 \cdot 48}{5!}$

Questions?

..order doesn't matter.

Choose 2 out of n?
$$\frac{n \times (n-1)}{2} = \frac{n!}{(n-2)! \times 2}$$

Choose 3 out of n?
$$\frac{n \times (n-1) \times (n-2)}{3!} = \frac{n!}{(n-3)! \times 3!}$$

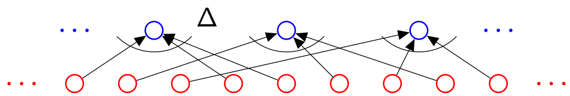
Choose k **out of** n?
$$\frac{n!}{(n-k)! \times k!}$$

Notation: $\binom{n}{k}$ **and pronounced "**n **choose** k**."**

Familiar? Questions?

Example: Visualize the proof..

First rule: $n_1 \times n_2 \cdots \times n_3$**. Product Rule.**
Second rule: when order doesn't matter divide...

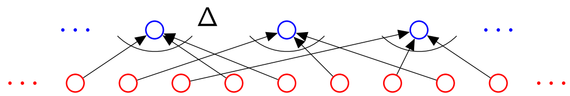


3 card Poker deals: $52 \times 51 \times 50 = \frac{52!}{49!}$. First rule.
Poker hands: Δ?
Hand: Q, K, A.
Deals: Q, K, A : Q, A, K : K, A, Q : K, A, Q : A, K, Q : A, Q, K.
$\Delta = 3 \times 2 \times 1$ First rule again.
Total: $\frac{52!}{49!3!}$ Second Rule!

Choose k out of n.
Ordered set: $\frac{n!}{(n-k)!}$ Orderings of one hand? $k!$ (By first rule!)
$\implies$ Total: $\frac{n!}{(n-k)!k!}$ Second rule.

Example: Anagram

First rule: $n_1 \times n_2 \cdots \times n_3$**. Product Rule.**
Second rule: when order doesn't matter divide...



Orderings of ANAGRAM?
Ordered Set: 7! First rule.
A's are the same!
What is Δ?
ANAGRAM
$A_1NA_2GRA_3M$, $A_2NA_1GRA_3M$, ...
$\Delta = 3 \times 2 \times 1 = 3!$ First rule!
$\implies \frac{7!}{3!}$ Second rule!

Poll

Mark what's correct.

(A) $|\text{Poker hands}| = \binom{52}{5}$
(B) Orderings of ANAGRAM $= 7!/3!$
(C) Orderings of "CAT". $= 3!$
(D) Orders of MISSISSIPPI $= 11!/4!4!2!$
(E) Orderings of ANAGRAM $= 7!/4!$
(F) Orders of MISSISSIPPI $= 11!/10!$

(A)-(E) are correct.

Some Practice.

How many orderings of letters of CAT?

3 ways to choose first letter, 2 ways for second, 1 for last.

$\implies 3 \times 2 \times 1 = 3!$ orderings

How many orderings of the letters in ANAGRAM?

Ordered, except for A!

total orderings of 7 letters. 7!
total "extra counts" or orderings of three A's? 3!

Total orderings? $\frac{7!}{3!}$

How many orderings of MISSISSIPPI?

4 S's, 4 I's, 2 P's.
11 letters total.
$11!$ ordered objects.
$4! \times 4! \times 2!$ ordered objects per "unordered object"

$\implies \frac{11!}{4!4!2!}$.

Summary.

First rule: $n_1 \times n_2 \cdots \times n_3$.

k Samples with replacement from n items: n^k.
Sample without replacement: $\frac{n!}{(n-k)!}$

Second rule: when order doesn't matter (sometimes) can divide...

Sample without replacement and order doesn't matter: $\binom{n}{k} = \frac{n!}{(n-k)!k!}$.
"n choose k"

One-to-one rule: equal in number if one-to-one correspondence.
pause Bijection!

Sample k times from n objects with replacement and order doesn't matter: $\binom{k+n-1}{n-1}$.