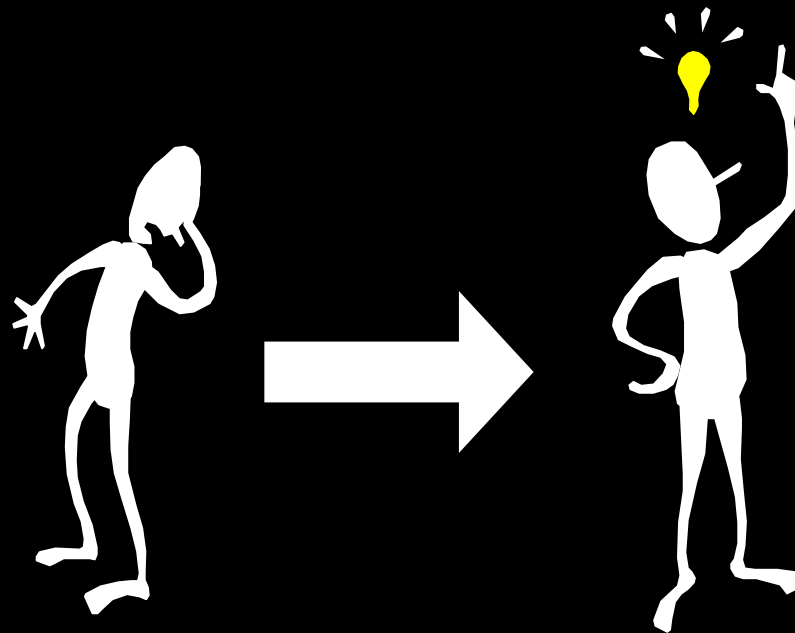


# 15-251

## Great Theoretical Ideas in Computer Science

# Bits of Wisdom on Solving Problems, Writing Proofs, and Enjoying the Pain: How to Succeed in This Class

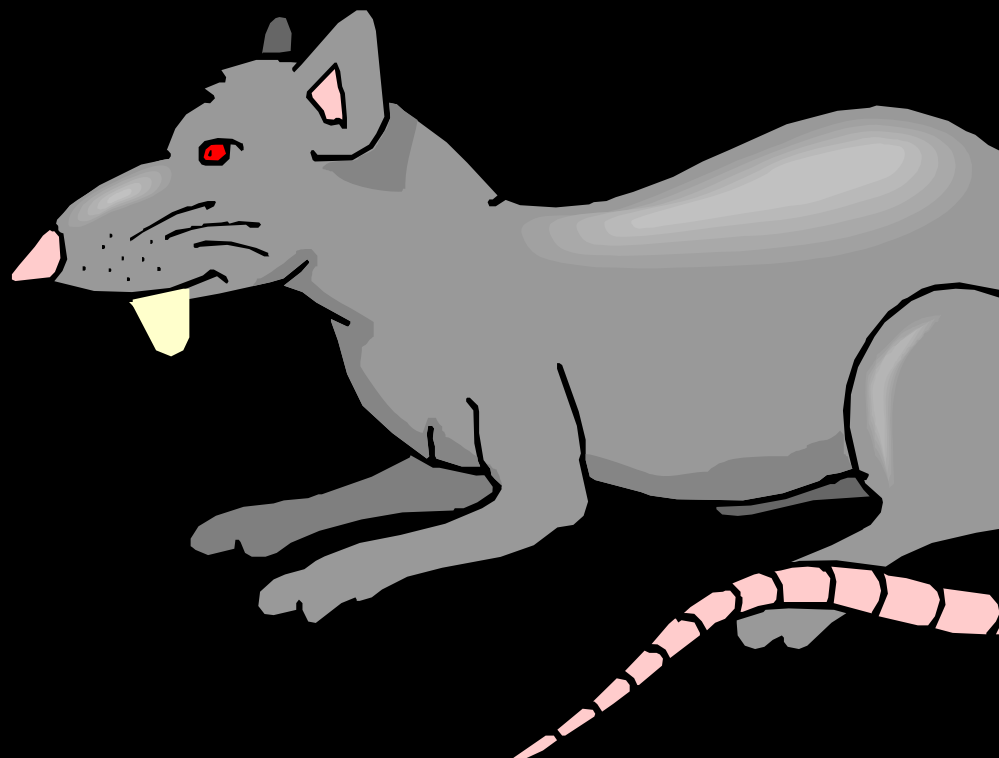
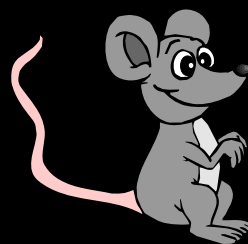
Lecture 4 (September 6, 2006)



**What did our brains evolve to do?**

**What were our brains  
designed to do?**





**The human mind was designed by evolution  
to deal with foraging in small bands on the  
African Savannah . . . faulting our minds for  
succumbing to games of chance is like  
complaining that our wrists are poorly  
designed for getting out of handcuffs**

**Steven Pinker  
“How the Mind Works”**

# Our brains did NOT evolve to do math!

Over the last 30,000 years, our brains  
have essentially stayed the same!

Of course, this doesn't mean  
you can't teach it to do math really well...

Our brains can perform simple,  
concrete tasks very well

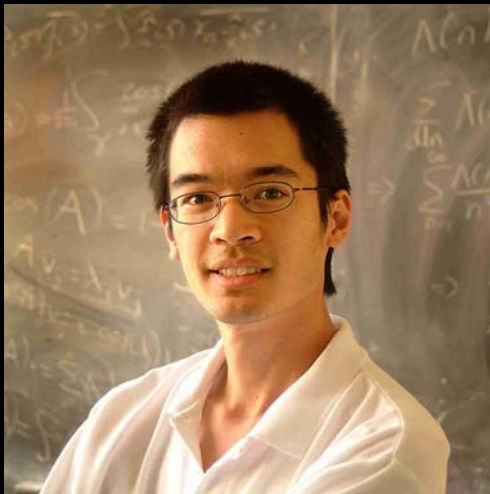
And that's how math  
should be approached!

Substitute concrete values for the  
variables:  $x=0$ ,  $x=100$ , ...

Draw simple pictures

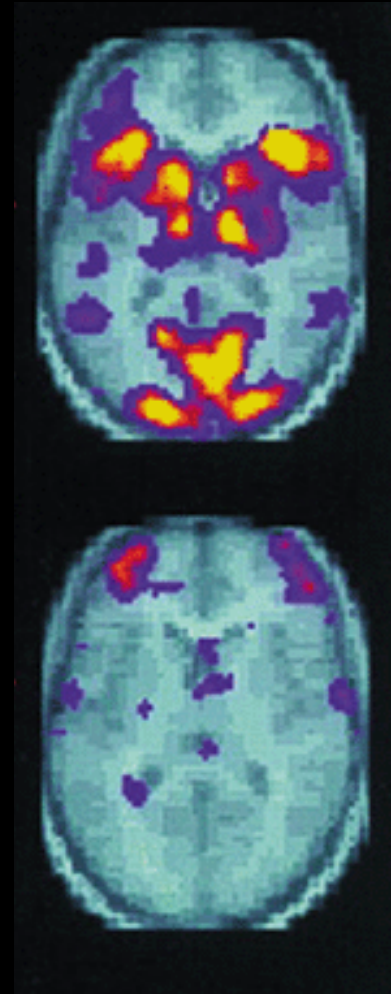
Try out small examples of the problem: What  
happens for  $n=1$ ?  $n=2$ ?

**“I don’t have any magical ability...I look at the problem, and it looks like one I’ve already done. When nothing’s working out, then I think of a small trick that makes it a little better. I play with the problem, and after a while, I figure out what’s going on.”**



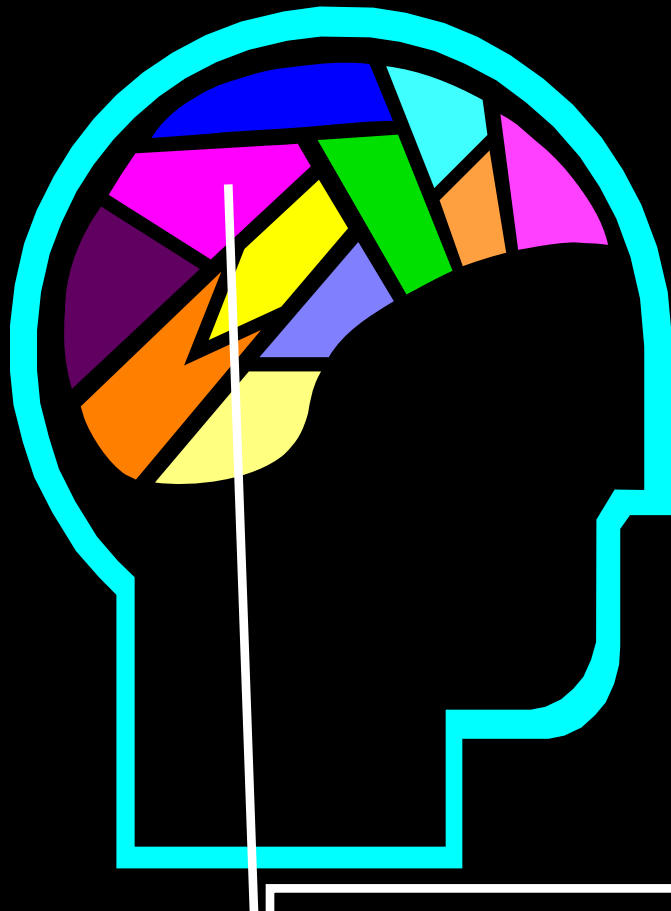
**Terry Tao (Fields Medalist,  
considered to be the best  
problem solver in the  
World)**





**Novice**

**Expert**



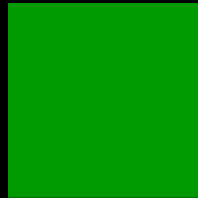
The better the  
problem solver, the  
less brain activity is  
evident.

The real masters  
show almost no brain  
activity!

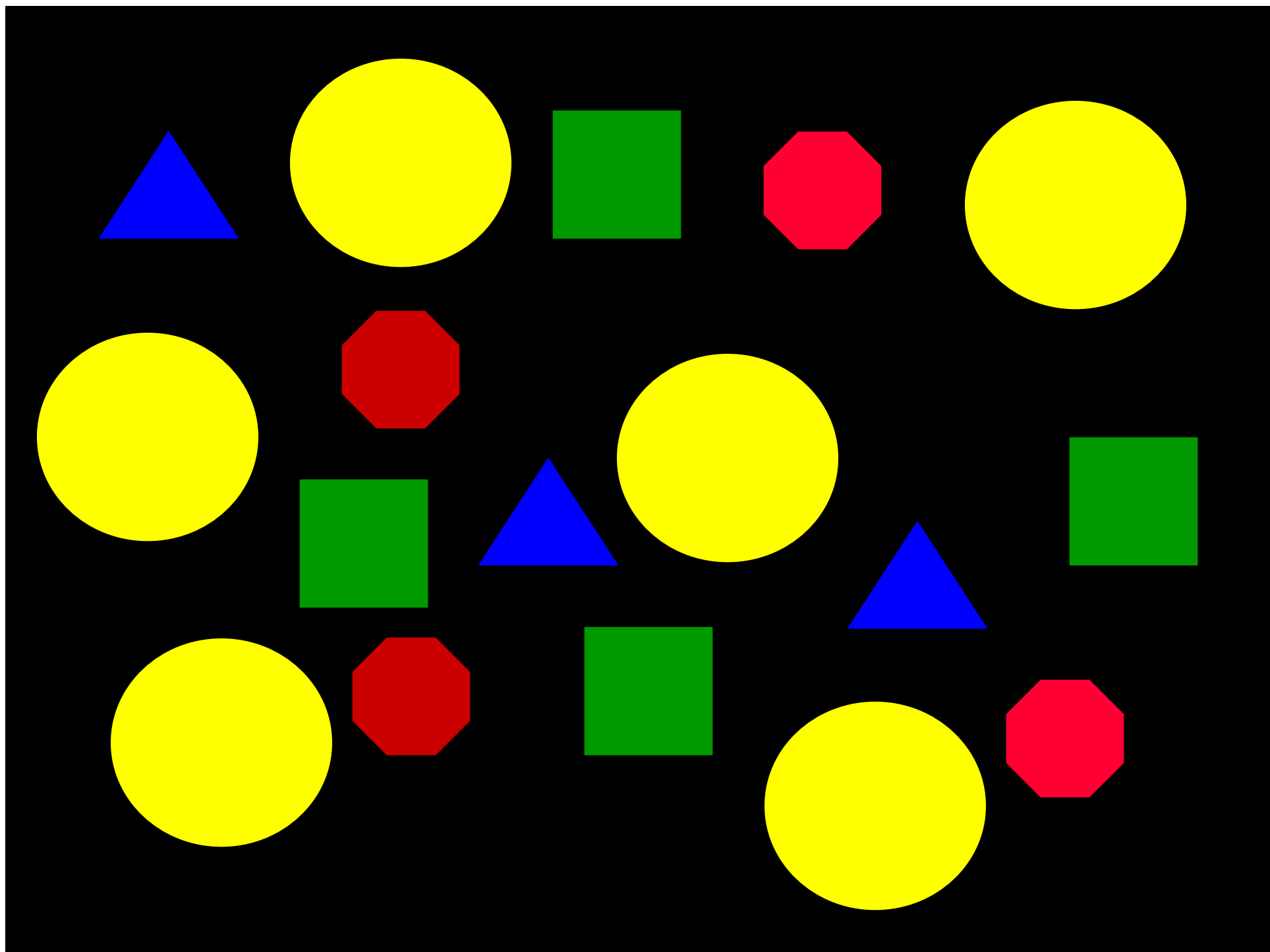
Simple and to the point

**Use a lot of paper,  
or a board!!!**

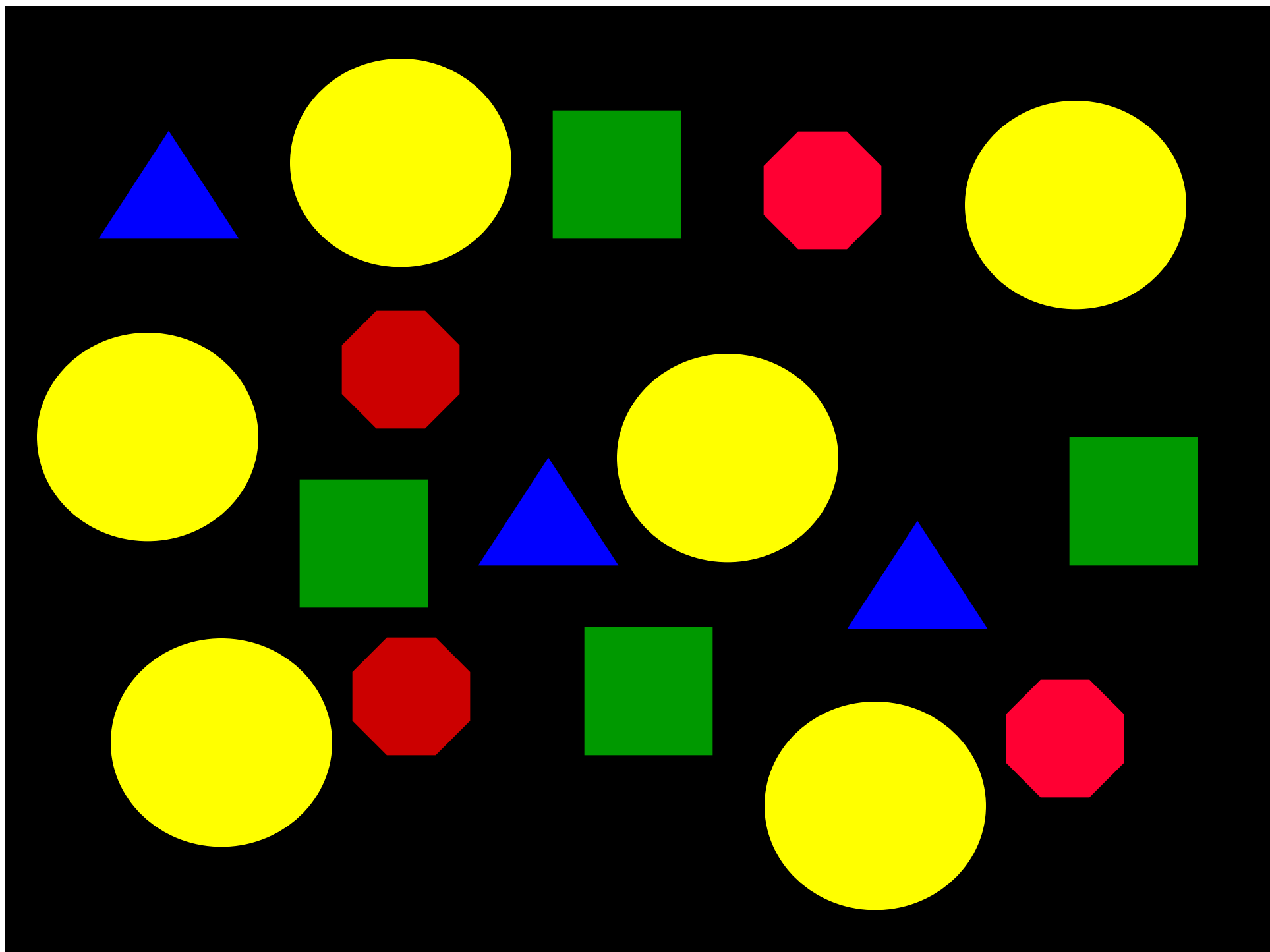
# Quick Test...



Count the green squares...  
you will have three seconds...



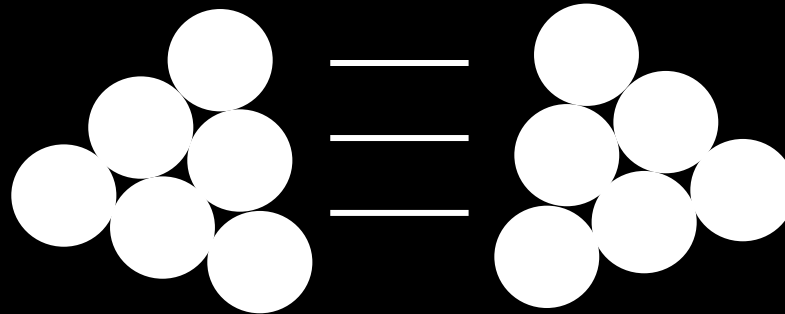
**How many were there?**



$$n(n+1) - 1 \quad ??$$

## Magnets

$$\begin{aligned} & (n-1) + (n-2) \\ & + \dots + 1 \\ & = \frac{n(n-1)}{2} \end{aligned}$$



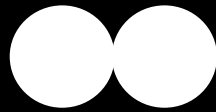
$n$  magnets are clumped

When the clump is split up into  $n_1$  and  $n_2$  pieces, the energy spent is  $n_1 n_2$

What's the least amount of energy necessary to split the clump into single magnets?



# Two Magnets



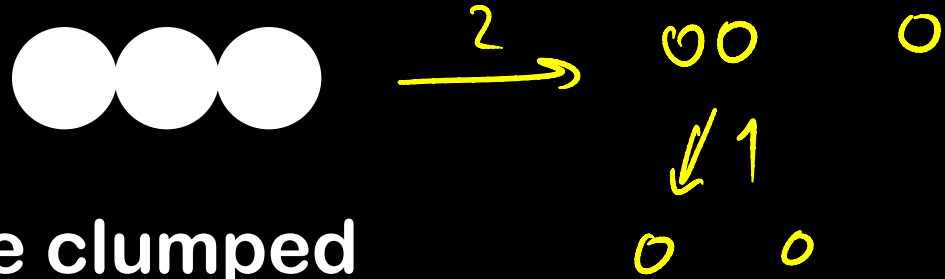
Two magnets are clumped

When the clump is split up into 1 and 1 pieces, the energy spent is 1

What's the least amount of energy necessary to split the clump into single magnets?

**Energy = 1**

# Three Magnets



Three magnets are clumped

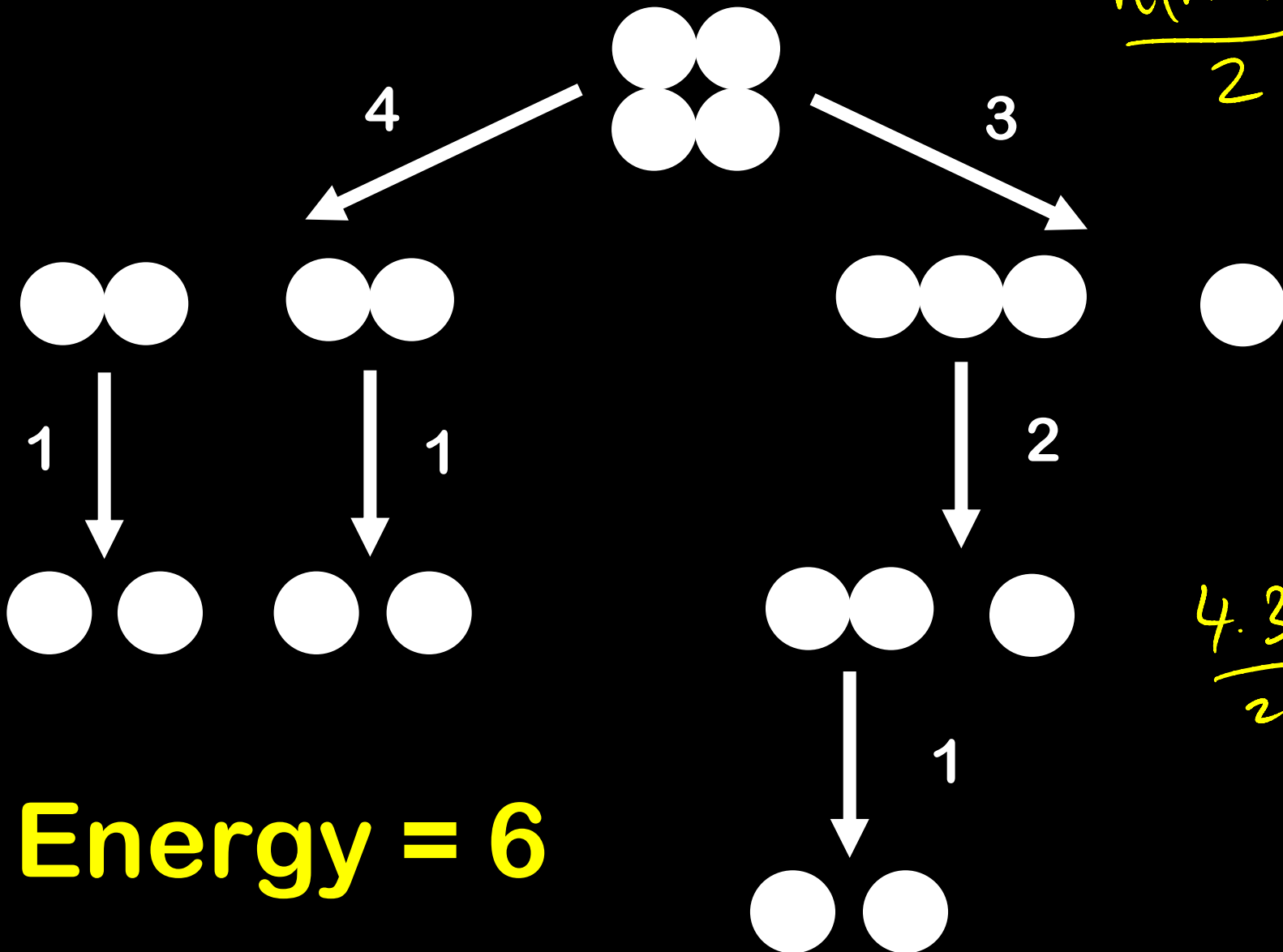
When the clump is split up into 2 and 1 pieces, the energy spent is 2

What's the least amount of energy necessary to split the clump into single magnets?

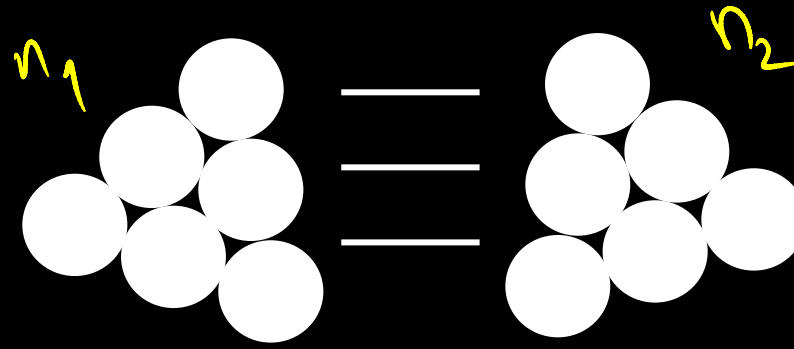
**Energy = 3**

# Four Magnets

$$\frac{n(n-1)}{2}$$



**Energy = 6**



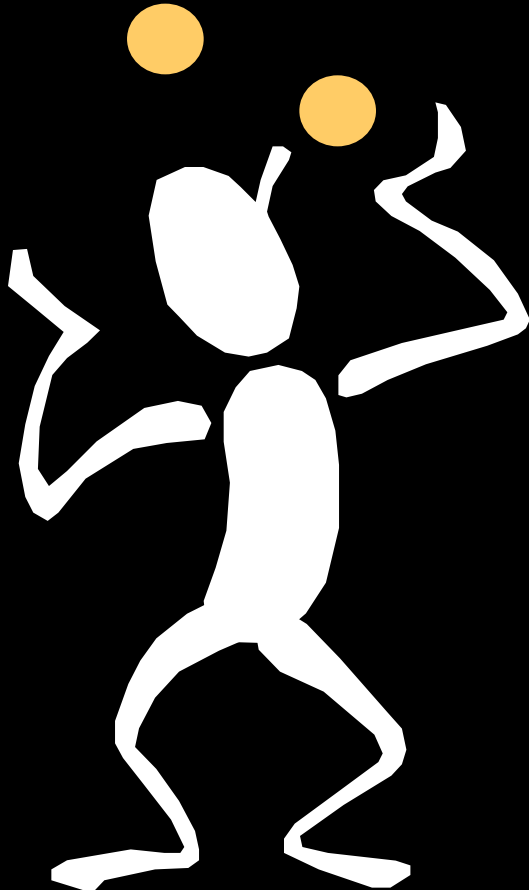
Every time you split a magnet from  $n$  other magnets, you must spend  $n$  units of energy

Every magnet must be separated from  $n-1$  other magnets

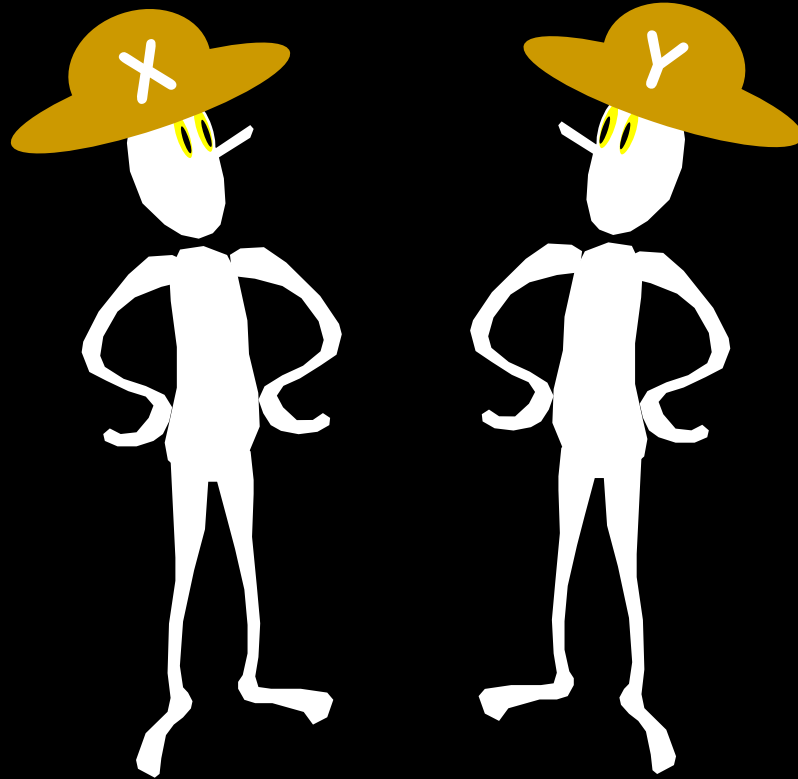
$$\text{Energy} = \frac{n(n-1)}{2} \quad \text{pairs}$$

## Exemplification:

Try out a problem or solution on small examples. Look for the patterns.



# Hats with Consecutive Numbers



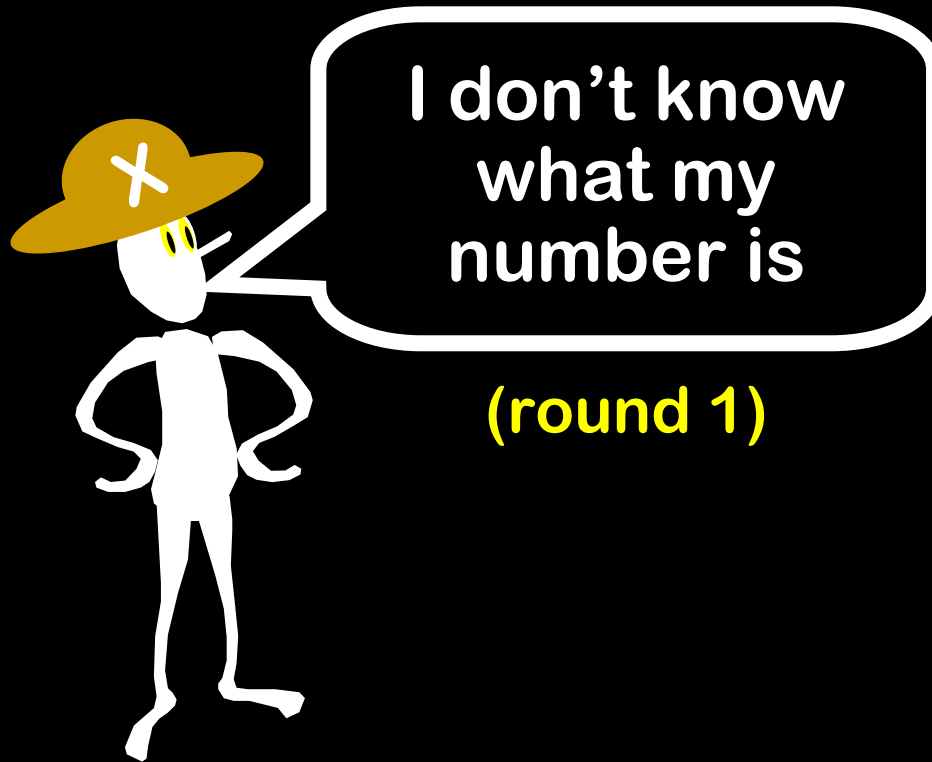
Alice

Bob

$$|X - Y| = 1$$

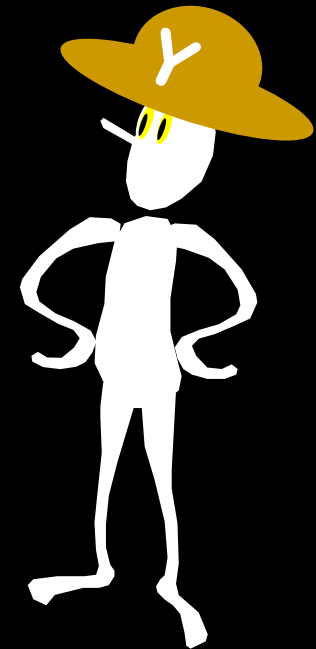
Alice starts: ...

# Hats with Consecutive Numbers



(round 1)

Alice

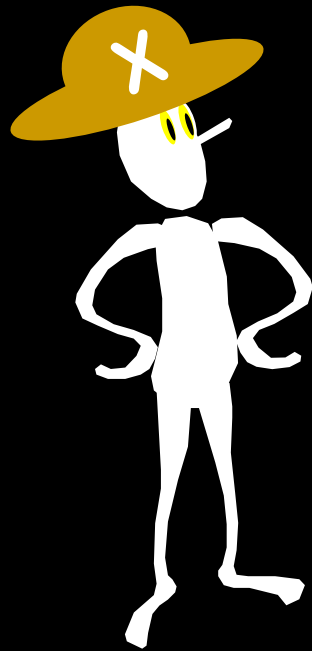


Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

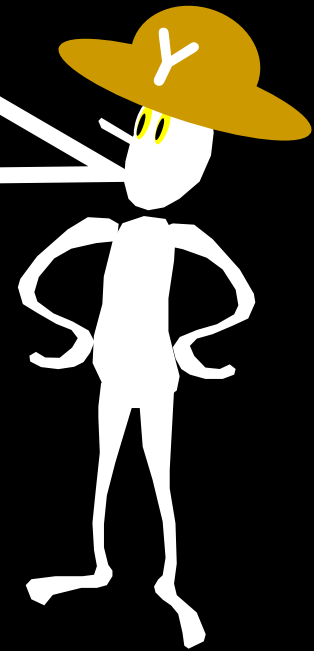
# Hats with Consecutive Numbers



Alice

I don't know  
what my  
number is

(round 2)



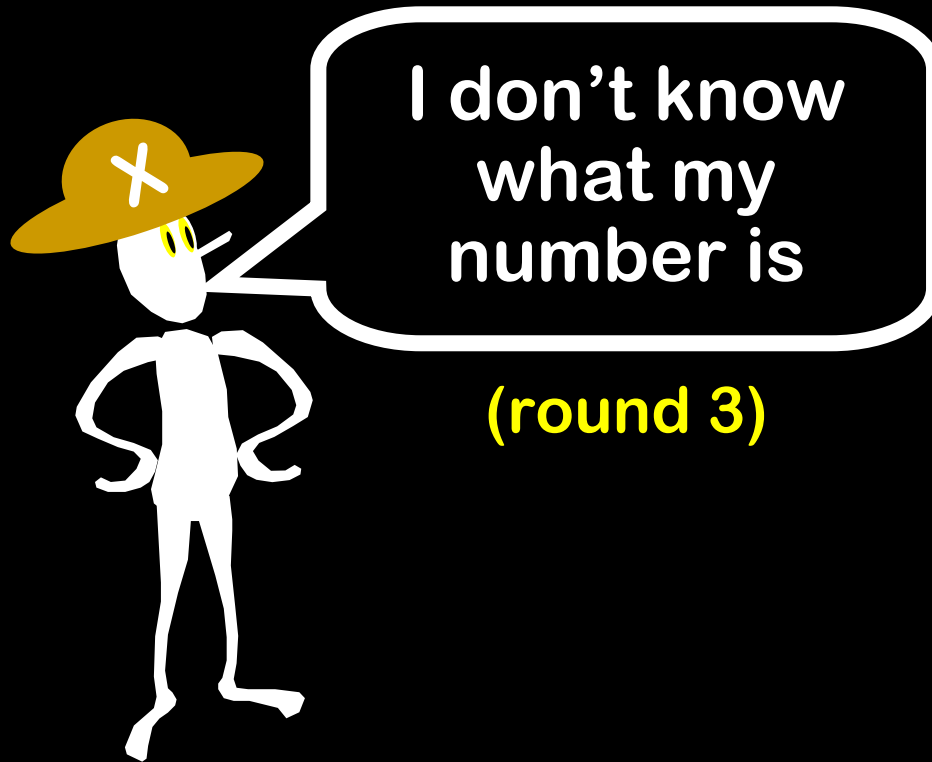
Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

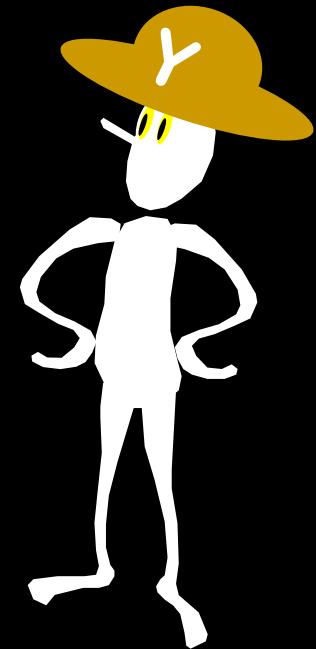


# Hats with Consecutive Numbers



(round 3)

Alice

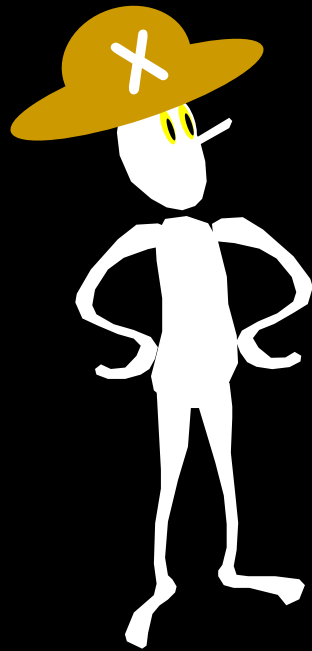


Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

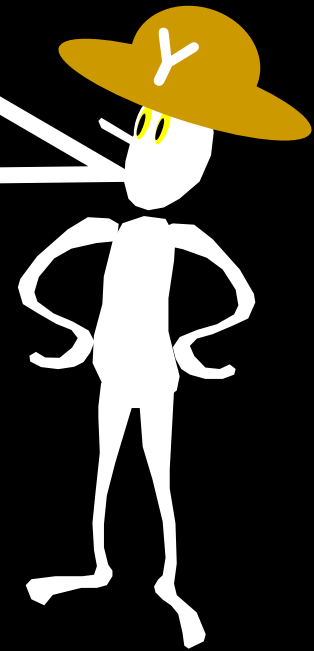
# Hats with Consecutive Numbers



Alice

I don't know  
what my  
number is

(round 4)



Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

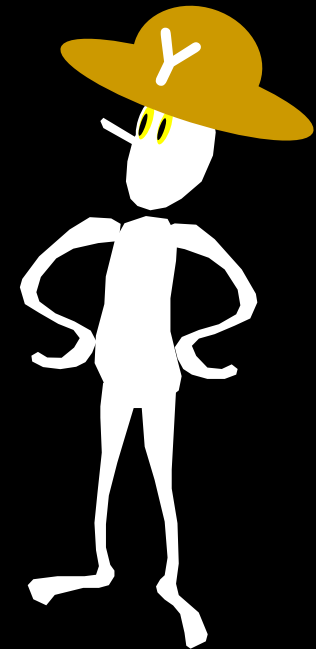


# Hats with Consecutive Numbers



Alice

(round 251)

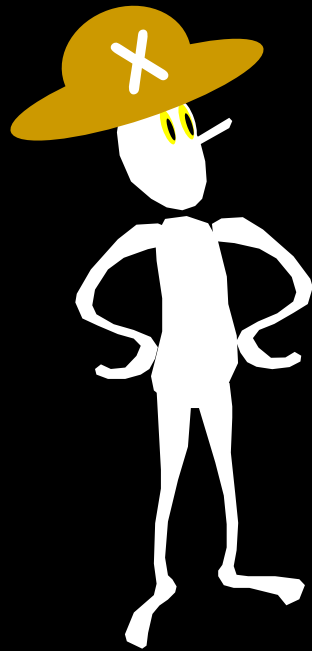


Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

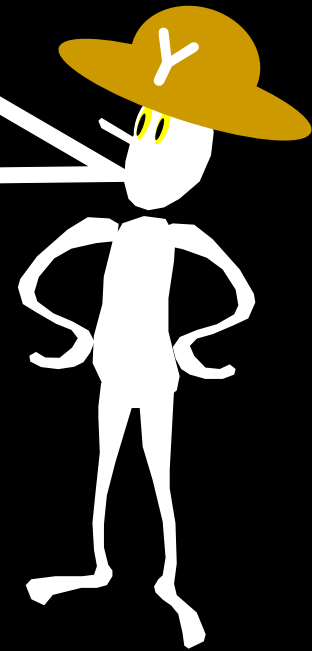
# Hats with Consecutive Numbers



Alice

I know what  
my number  
is!!!!!!!

(round 252)



Bob

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Alice starts: ...

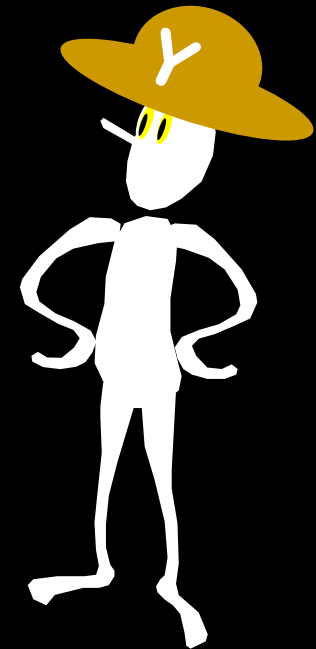
**Question: What are Alice  
and Bob's numbers?**

# Imagine Alice Knew Right Away



(round 1)

Alice



Bob

125

$$|X - Y| = 1 \text{ and } X, Y > 0$$

Then  $X = 2$  and  $Y = 1$

1,2

N,Y

N,y

2,1

Y

2,3

N,Y

N

3,2

N,N,Y

N,N,y

3,4

N,N,N,Y

N,N,N,

4,3

N,N,Y

4,5

N,N,N,Y



# Inductive Claim

**Claim:** After  $2k$  NOs, Alice knows that her number is at least  $2k+1$ .

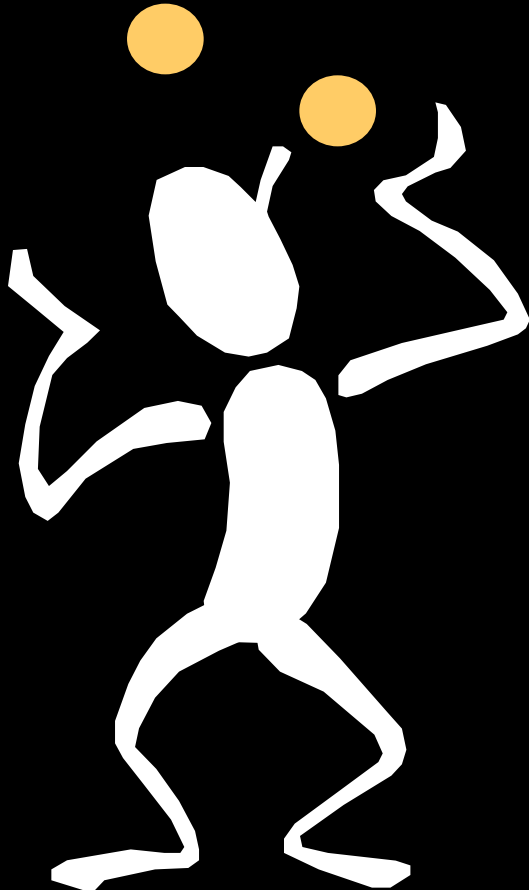
After  $2k+1$  NOs, Bob knows that his number is at least  $2k+2$ .

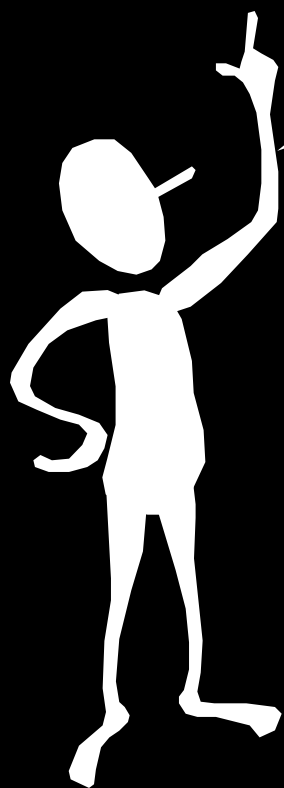
Hence, after 250 NOs, Alice knows her number is at least 251. If she says YES, her number is at most 252.

If Bob's number is 250, her number must be 251.  
If his number is 251, her number must be 252.

## Exemplification:

Try out a problem or solution on small examples. Look for the patterns.

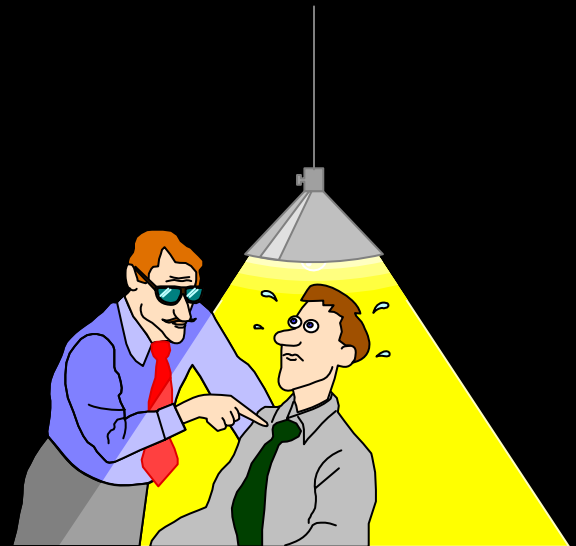




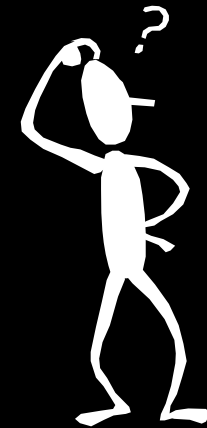
A volunteer, please

# Relax

I am just going to ask you a  
Microsoft interview question



Four guys want to cross a bridge that can only hold two people at one time. It is pitch dark and they only have one flashlight, so people must cross either alone or in pairs (bringing the flashlight). Their walking speeds allow them to cross in 1, 2, 5, and 10 minutes, respectively. Is it possible for them to all cross in 17 minutes?

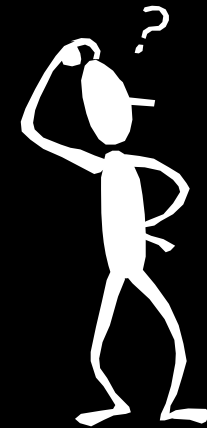


# Get The Problem Right!

Given any context you should double check that you read/heard it correctly!

You should be able to repeat the problem back to the source and have them agree that you understand the issue

Four guys want to cross a bridge that can only hold two people at one time. It is pitch dark and they only have one flashlight, so people must cross either alone or in pairs (bringing the flashlight). Their walking speeds allow them to cross in 1, 2, 5, and 10 minutes, respectively. Is it possible for them to all cross in 17 minutes?



## Intuitive, But False

“ $10 + 1 + 5 + 1 + 2 = 19$ , so the four guys just can't cross in 17 minutes”

“Even if the fastest guy is the one to shuttle the others back and forth – you use at least  $10 + 1 + 5 + 1 + 2 > 17$  minutes”



# Vocabulary Self-Proofing

As you talk to yourself, make sure to tag assertions with phrases that denote degrees of conviction

Keep track of what you actually know  
– remember what you merely suspect

“ $10 + 1 + 5 + \cancel{1} + 2 = \overset{18}{\cancel{19}}$ , so it would be weird if the four guys could cross in 17 minutes”

“~~even~~ if we use the fastest guy to shuttle the others, they take too long.”



**If it is possible, there must be more than one guy doing the return trips: it must be that someone gets deposited on one side and comes back for the return trip later!**

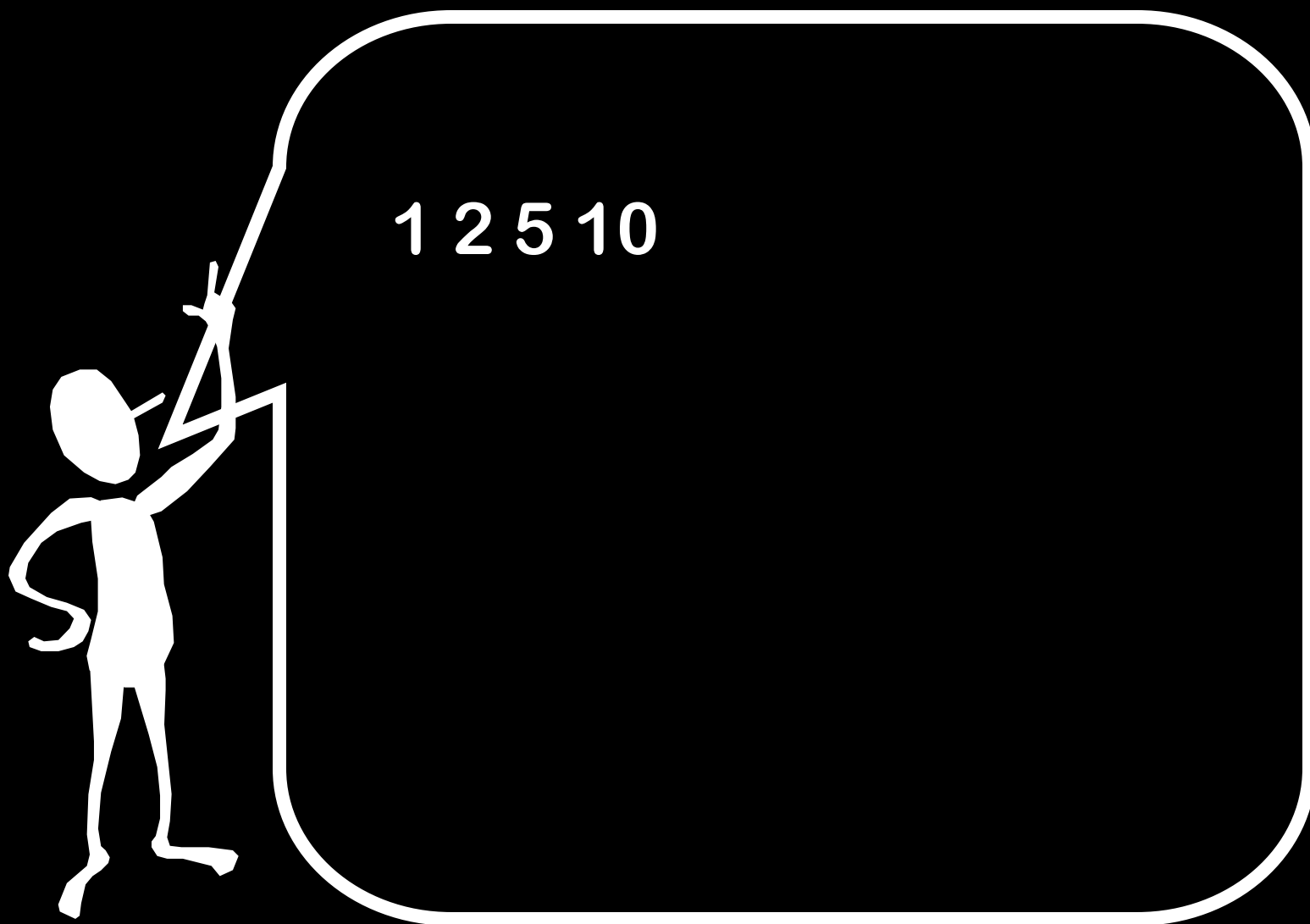


**Suppose** we leave 1 for a  
return trip later

We start with 1 and X and  
then X returns

Total time:  $2X$

Thus, we start with  
1,2 go over and  
2 comes back....



1 2 5 10





1 2 5 10

5 10

2 1



1 2 5 10

5 10

2 1





1 2 5 10

5 10

2 5 10

2 1

1



1 2 5 10

5 10

2 5 10

2 1

1



1 2 5 10

5 10

2 5 10

2

2 1

1

1 5 10



1 2 5 10

5 10

2 5 10

2

2 1

1

1 5 10



1 2 5 10

5 10

2 5 10

2

1 2

2 1

1

1 5 10

5 10



1 2 5 10

5 10

2 5 10

2

1 2

2 1

1

1 5 10

5 10



1 2 5 10

5 10

2 5 10

2

1 2

2

2 1

2

1

10

1 5 10

1

5 10

2

1 2 5 10



## **5 and 10 “Load Balancing”:**

**Handle our hardest  
work loads in parallel!  
Work backwards by  
assuming 5 and 10  
walk together**





1 2 5 10

5 10

2 5 10

2

1 2

2 1

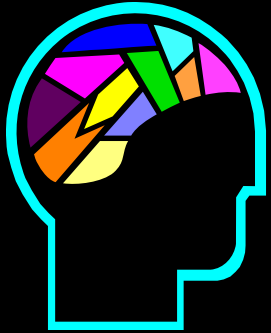
1

1 5 10

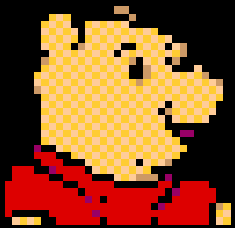
5 10

1 2 5 10

# Words To The Wise



Keep It Simple



Don't Fool Yourself

# That really was a Microsoft question



Why do you think that they ask such questions, as opposed to asking for a piece of code to do binary search?

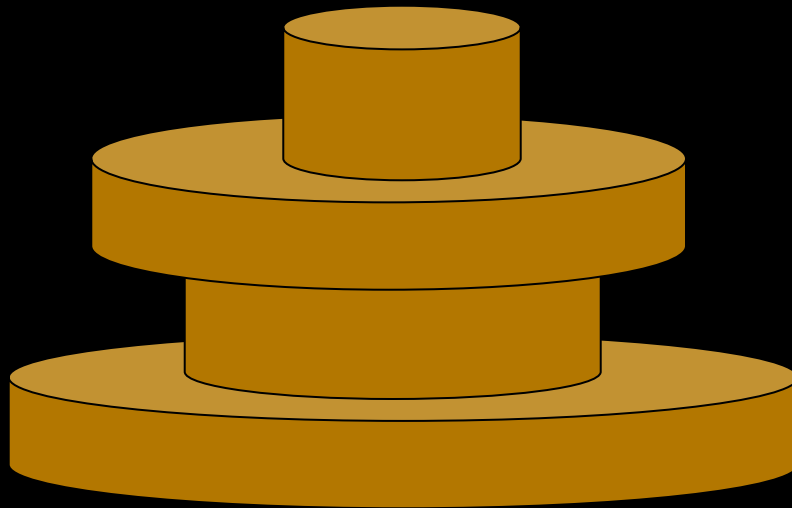
# The future belongs to the computer scientist who has

- **Content:** An up to date grasp of fundamental problems and solutions
- **Method:** Principles and techniques to solve the vast array of unfamiliar problems that arise in a rapidly changing field



## Representation:

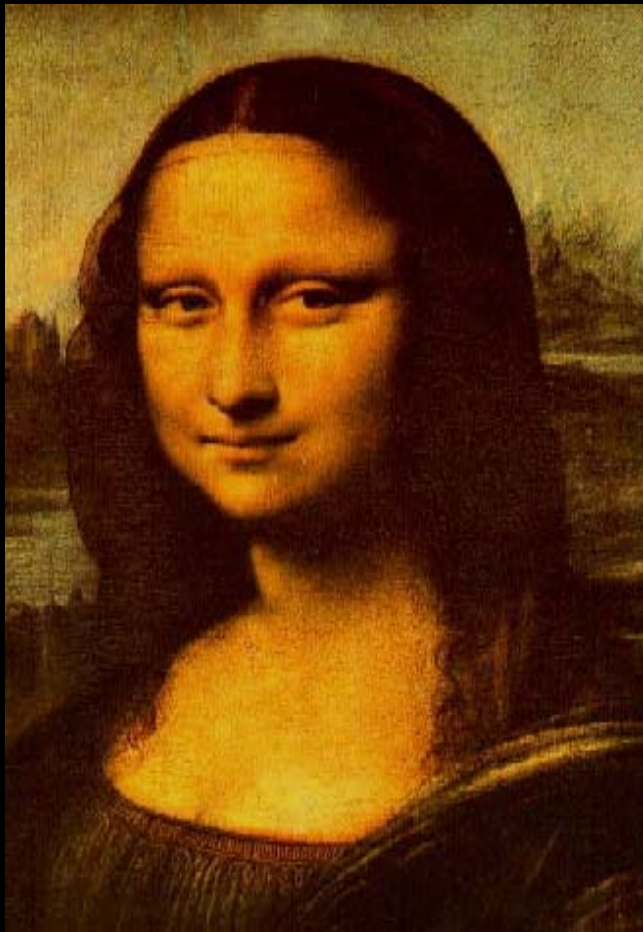
Understand the relationship between different representations of the same information or idea



1  
3  
2  
4

## Abstraction:

Abstract away the inessential  
features of a problem

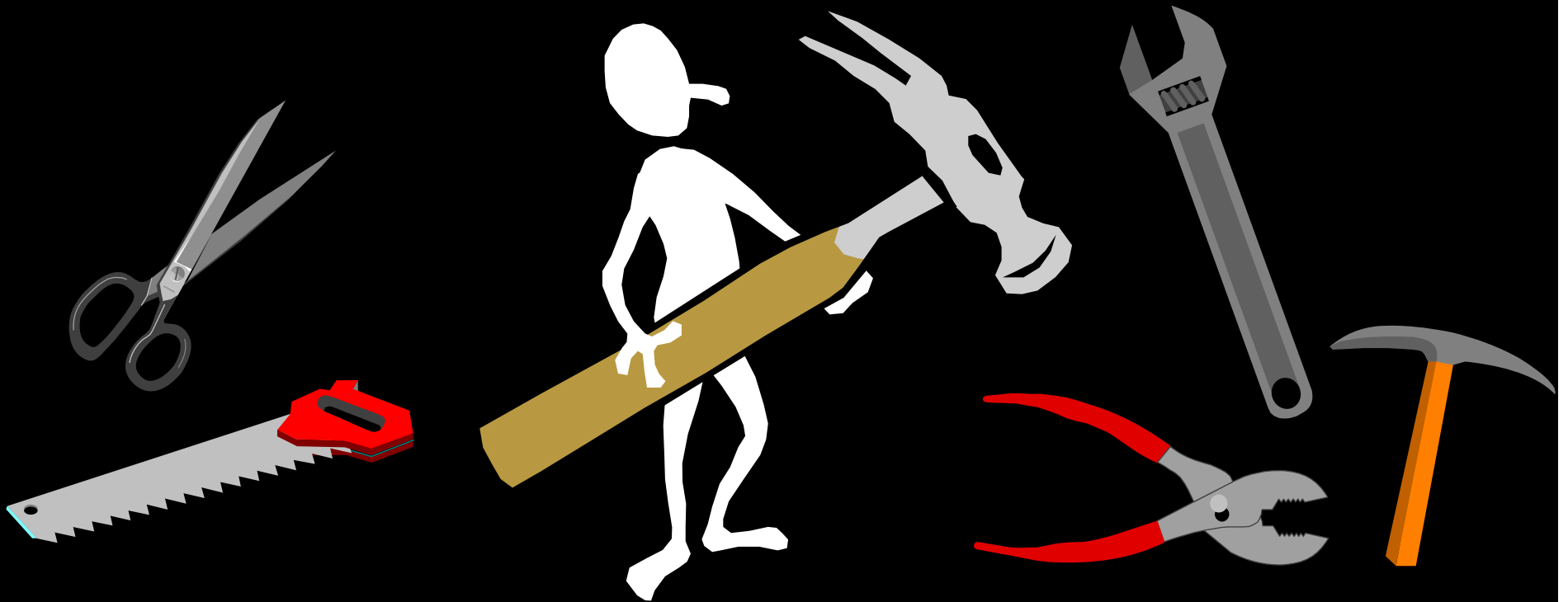


=



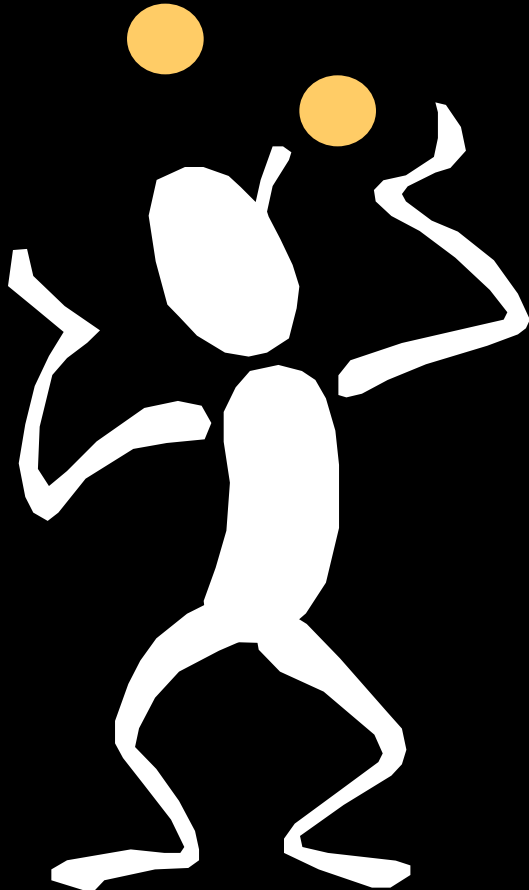
## Toolkit:

Name abstract objects and ideas,  
and put them in your toolkit. Know  
their advantages and limitations.

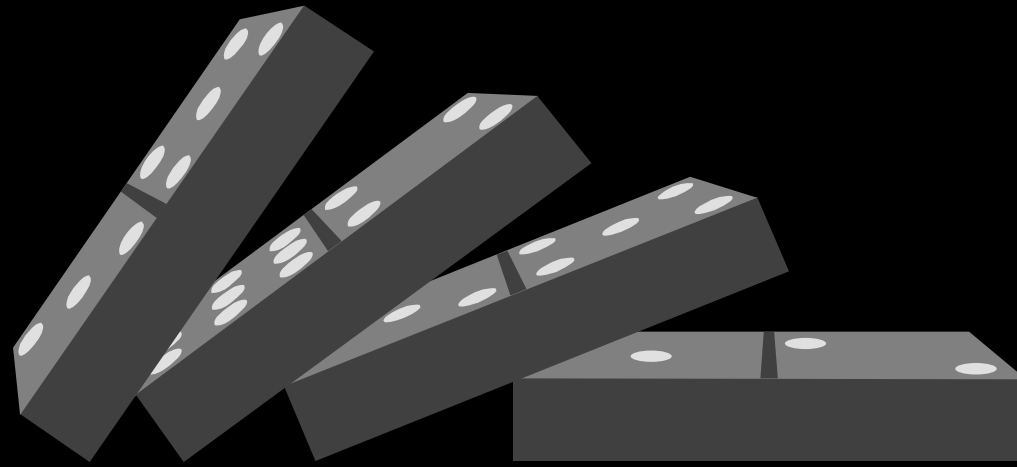


## Exemplification:

Try out a problem or solution on small examples. Look for the patterns.







**Induction has many guises.**  
**Master their interrelationship.**

- Formal Arguments
- Invariants
- Recursion
- Recurrences

# Modularity:

Decompose a complex problem  
into simpler sub-problems



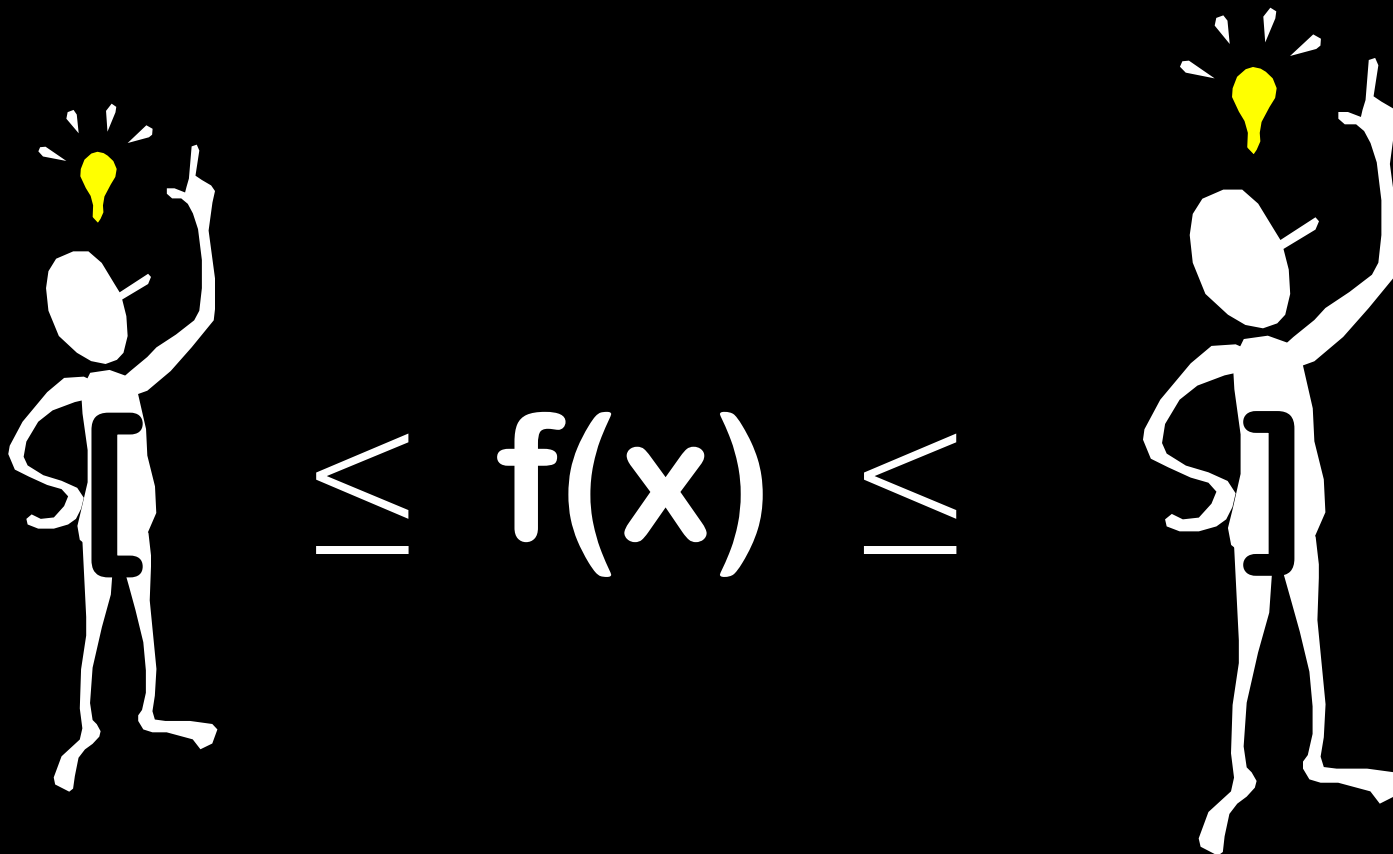
## Improvement:

The best solution comes from a process of repeatedly refining and improving solutions and proofs.



## Bracketing:

What are the best lower and upper bounds that I can prove?



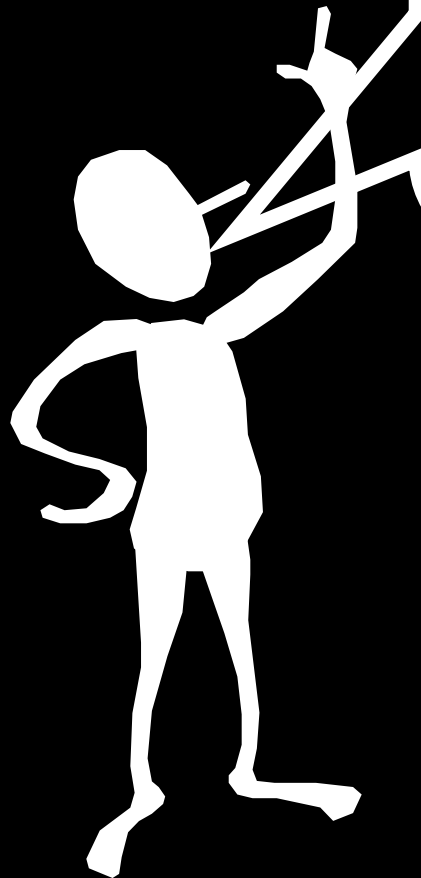
**In this course you will have  
to write a lot of proofs!**

# Think of Yourself as a (Logical) Lawyer



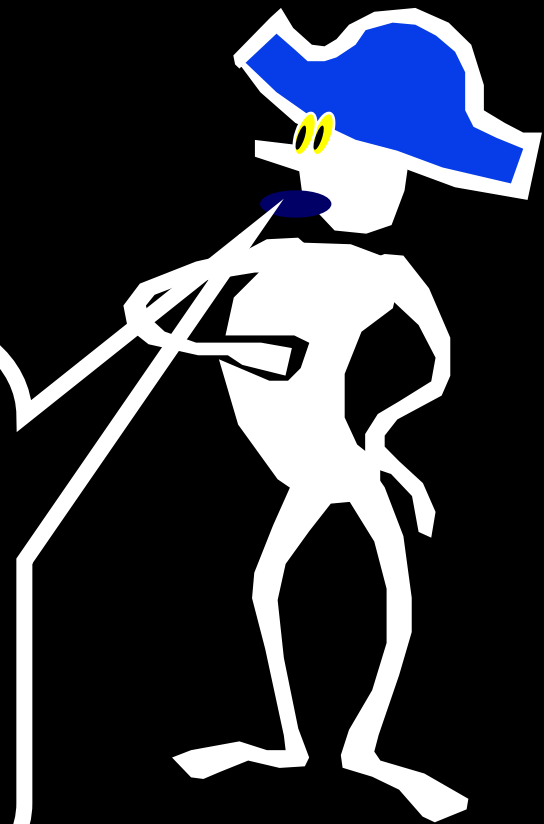
**Your arguments should have no holes, because the opposing lawyer will expose them**

Statement<sub>1</sub>  
Statement<sub>2</sub>  
⋮  
Statement<sub>n</sub>



**Prover**

There is no  
sound reason  
to go from  
Statement<sub>1</sub> to  
Statement<sub>2</sub>

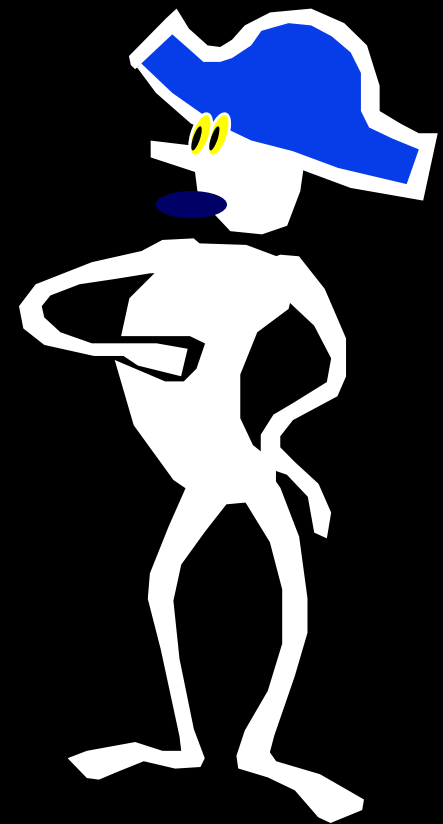


**Verifier**

The verifier is very thorough,  
(he can catch all your mistakes),  
but he will not supply missing  
details of a proof

A valid complaint on his part  
is: **I don't understand**

The verifier is similar to a  
computer running a program  
that you wrote!



Verifier



# Writing Proofs Is A Lot Like Writing Programs

You have to write the correct sequence of statements to satisfy the verifier

Errors than can occur with a program and with a proof!



Syntax error

Undefined term

Infinite Loop

Output is not quite what was needed

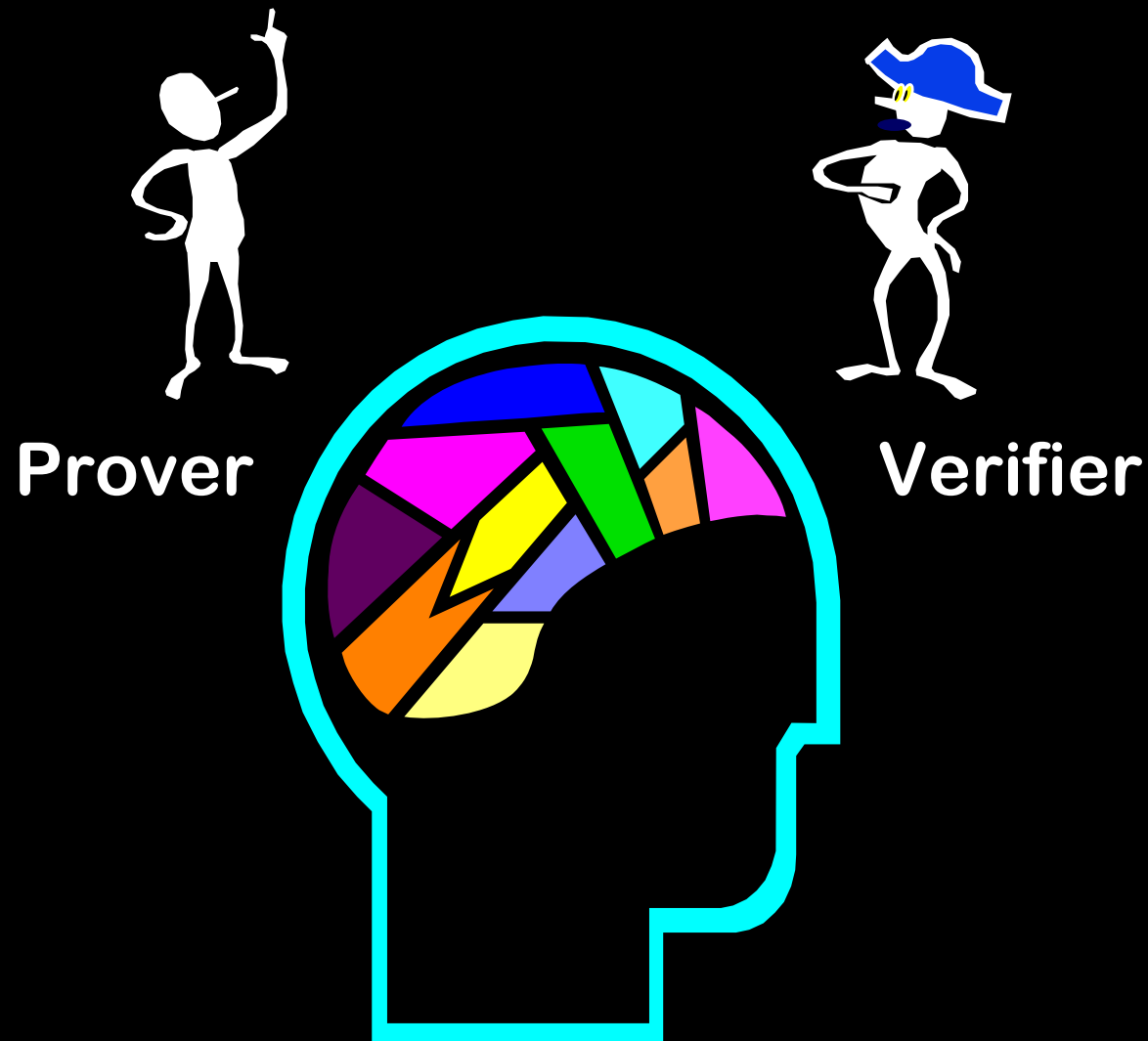
Good code is well-commented and written in a way that is easy for other humans (and yourself) to understand

Similarly, good proofs should be easy to understand. Although the formal proof does not require certain explanatory sentences (e.g., “the idea of this proof is basically X”), good proofs usually do

# Writing Proofs is Even Harder than Writing Programs

The proof verifier will not accept a proof unless every step is justified!

It's as if a compiler required your programs to have every line commented (using a special syntax) as to why you wrote that line



**A successful mathematician plays both roles  
in their head when writing a proof**

# Gratuitous induction proof.

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

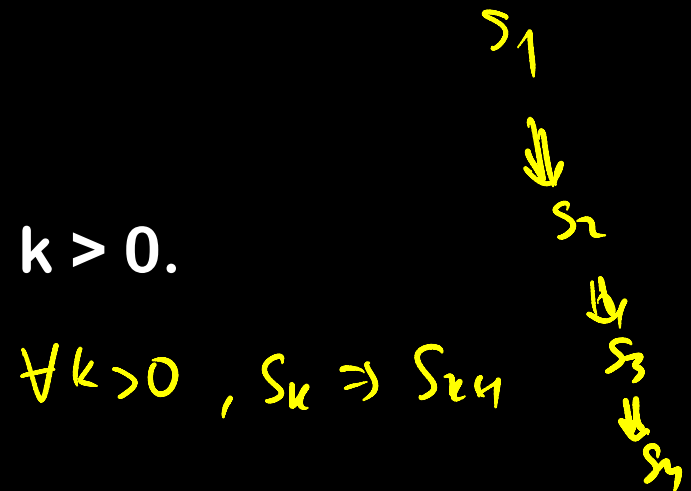
Base case:  $S_1 = “1 = 1(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for some  $k > 0$ .

Induction step:

$$\begin{aligned} 1 + 2 + 3 + \dots + k + (k+1) &= k(k+1)/2 + (k+1) \text{ (by I.H.)} \\ &= (k+1)(k+2)/2. \end{aligned}$$

Thus  $S_{k+1}$ .



# Gratuitous induction proof.

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

Base case:  $S_1 = “1 = 1(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for some  $k > 0$ .

Induction step:

what’s  $n$ ??

$$\begin{aligned} 1 + 2 + 3 + \dots + n + (n+1) &= n(n+1)/2 + (n+1) \text{ (by I.H.)} \\ &= (n+1)(n+2)/2. \end{aligned}$$

Thus  $S_{k+1}$ .

# Gratuitous induction proof.

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

Base case:  $S_1 = “1 = 1(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for some  $k > 0$ .

Induction step:

$$\begin{aligned} 1 + 2 + 3 + \dots + k + (k+1) &= k(k+1)/2 + (k+1) \text{ (by I.H.)} \\ &= (k+1)(k+2)/2. \end{aligned}$$

Thus  $S_{k+1}$ .





**10**

## **Proof by Throwing in the Kitchen Sink**

**The author writes down every theorem or result known to mankind and then adds a few more just for good measure**

**When questioned later, the author correctly observes that the proof contains all the key facts needed to actually prove the result**

**Very popular strategy on 251 exams**

**Believed to result in extra credit with sufficient whining**

**10**

## Proof by Throwing in the Kitchen Sink

The author writes down every theorem

Like writing a program with functions that do most everything you'd ever want to do (e.g. sorting integers, calculating derivatives), which in the end simply prints "hello world"

sufficient whining

# 9

## Proof by Example

The author gives only the case  $n = 2$  and suggests that it contains most of the ideas of the general proof.

**Like writing a program that only works for a few inputs**

# 8

## Proof by Cumbersome Notation

Best done with access to at least four alphabets and special symbols.

Helps to speak several foreign languages.

Like writing a program  
that's really hard to read  
because the variable  
names are screwy

# 7

## Proof by Lengthiness

An issue or two of a journal devoted to your proof is useful. Works well in combination with Proof strategy #10 (throwing in the kitchen sink) and Proof strategy #8 (cumbersome notation).

Like writing 10,000 lines  
of code to simply print  
“hello world”

# 6

## Proof by Switcharoo

Concluding that  $p$  is true when both  $p \Rightarrow q$  and  $q$  are true

Makes as much sense as:

```
If (PRINT "X is prime") {  
    PRIME(X);  
}
```

## Switcharoo Example

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

Base case:  $S_1 = “1 = 1(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for some  $k > 0$ .

Induction step:

By  $S_{k+1}$ ,  $1 + 2 + 3 + \dots + k + (k+1) = (k+1)(k+2)/2$

Hence, blah blah,  $S_k$  is true.

**5**

## **Proof by “It is Clear That...”**

**“It is clear that that the worst case is this:”**

**Like a program that calls a  
function that you never wrote**



# 4

## Proof by Assuming The Result

Assume X is true

⋮

Therefore, X is true!

Like a program with this code:

```
RECURSIVE(X) {  
    :  
    :  
    return RECURSIVE(X);  
}
```

## “Assuming the Result” Example

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

Base case:  $S_1 = “1 = 1(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for all  $k > 0$ .

Induction step:

$$\underbrace{1 + 2 + 3 + \dots + n}_n + (n+1) = n(n+1)/2 + (n+1) \text{ (by I.H.)}$$
$$= (n+1)(n+2)/2.$$

Thus  $S_{k+1}$ .

# 3

## Not Covering All Cases

Usual mistake in inductive proofs: A proof is given for  $N = 1$  (base case), and another proof is given that, for any  $N > 2$ , if it is true for  $N$ , then it is true for  $N+1$

Like a program with this function:

```
RECURSIVE(X) {  
    if (X > 2) { return 2*RECURSIVE(X-1); }  
    if (X = 1) { return 1; }  
}
```

## “Not Covering all Cases” Example

$S_n$  = “sum of first  $n$  integers’s =  $n(n+1)/2$ ”.

Want to prove:  $S_n$  is true for all  $n > 0$ .

Base case:  $S_0 = “0 = 0(1+1)/2”$ .

I.H. Suppose  $S_k$  is true for some  $k > 0$ .

Induction step:

$$\begin{aligned} 1 + 2 + 3 + \dots + \cancel{n} + \cancel{n} + 1 &= \cancel{n(n+1)/2} + \cancel{n+1} \text{ (by I.H.)} \\ &= \cancel{(n+1)(n+2)/2}. \end{aligned}$$

Thus  $S_{k+1}$ .

# 2

## Incorrectly Using “By Definition”

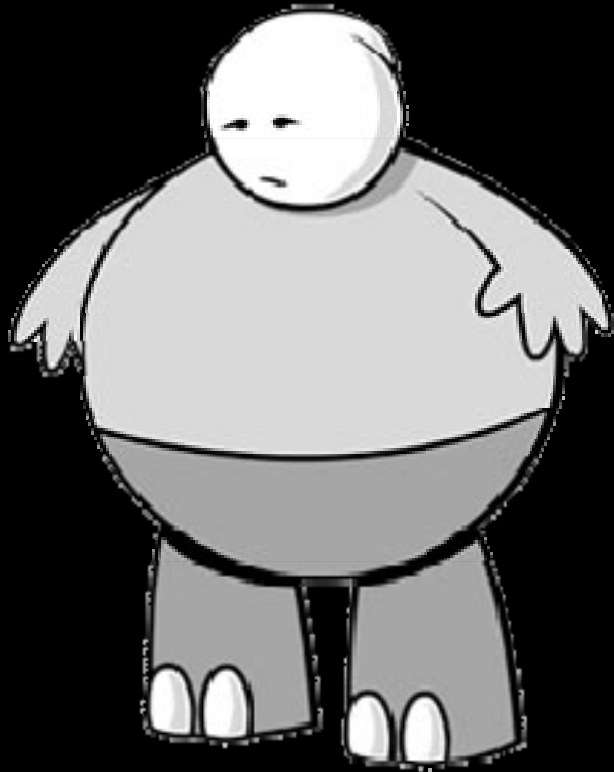
“By definition,  $\{ a^n b^n \mid n > 0 \}$  is not a regular language”

Like a program that assumes a procedure does something other than what it actually does

1

# Proof by OMGWTFBBQ





Here's What  
You Need to  
Know...

## Solving Problems

- Always try small examples!
- Use enough paper

## Writing Proofs

- Writing proofs is sort of like writing programs, except every step in a proof has to be justified
- Be careful; search for your own errors