Great Theoretical Ideas In Computer Science

Steven Rudich CS 15-251 Spring 2004

Lecture 13 Feb 24, 2004 Carnegie Mellon University

# Problem Solving: Where does the "aha!" come from?

A volunteer, please.

# Relax .....

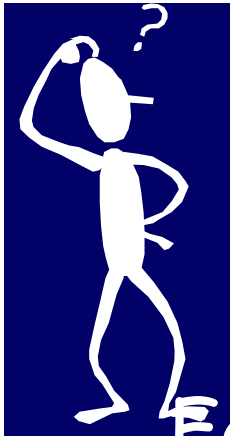
Relax, I am just going to ask you a Microsoft interview question.

# Do You Understand The Question?

Four guys want to cross a bridge that can only hold two people ay 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?

## You have one minute to solve this problem

Four guys want to cross a bridge that can only hold two people ay 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?



## So what is the answer?

Four guys want to cross a bridge that can only hold two people ay 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 + 5 + 2 + 1 = 18, 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 + 5 + 2 + 1 > 17 minutes"

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

"10 + 5 + 2 + 1 = 18, 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."

# Tagging Strategy

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

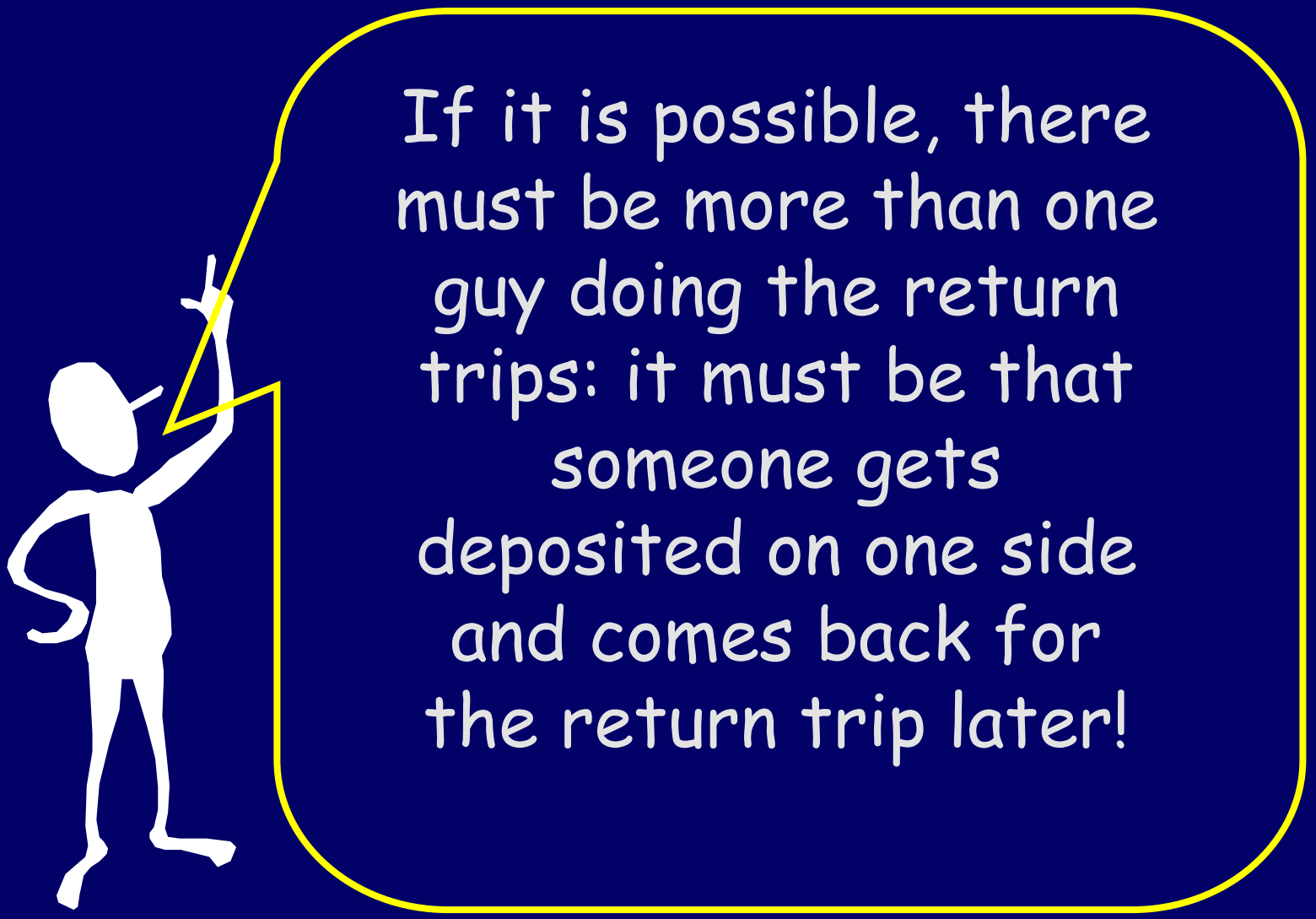
# Phrase Hygiene

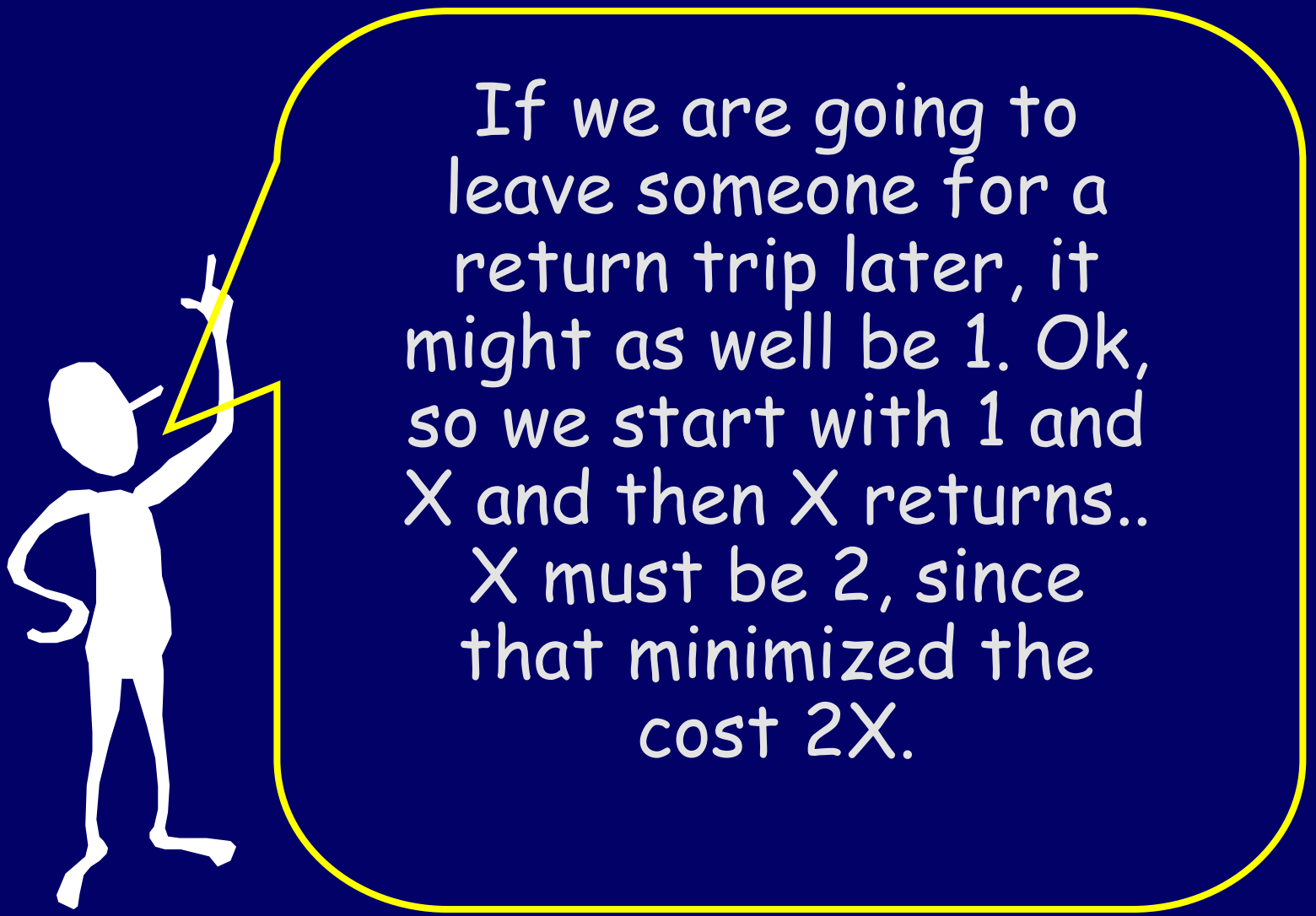
Develop stock phrases to classify statements. Learn from experience and eliminate ambiguous or misleading phrases like “Even if”.

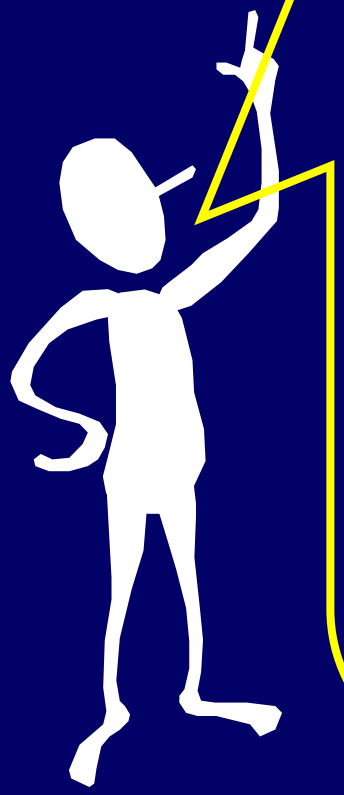
Keep track of what you actually know – remember what you merely suspect.

"10 + 5 + 2 + 1 = 18, 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!


If we are going to leave someone for a return trip later, it might as well be 1. Ok, so we start with 1 and X and then X returns.. X must be 2, since that minimized the cost 2X.



1 2 5 10

5 10

2 5 10

2

1 2

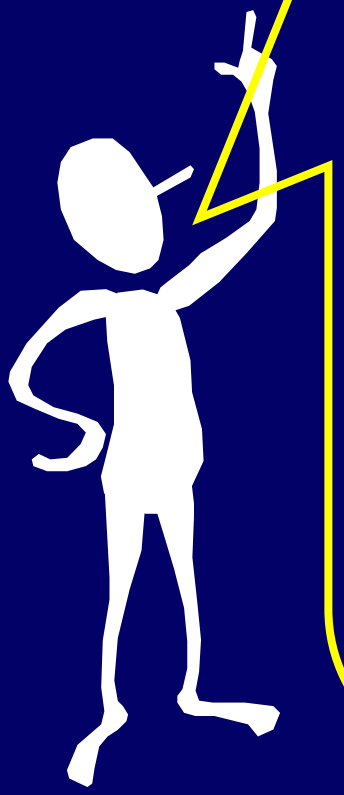
2 1

1

1 5 10

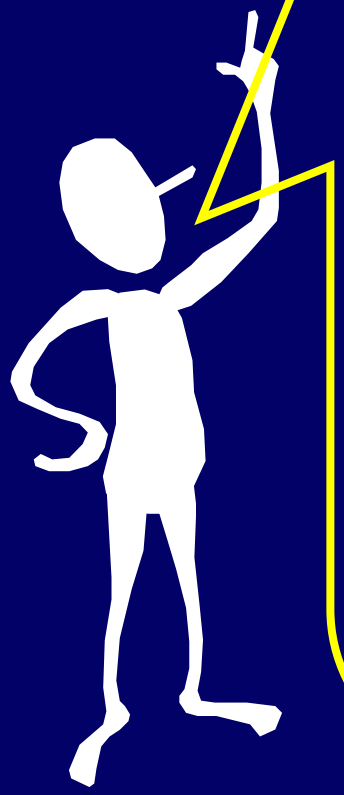
5 10

1 2 5 10



5 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

That really was a Microsoft question.

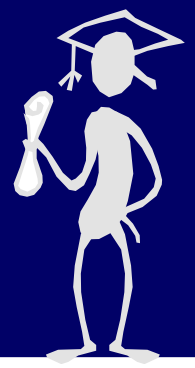
Why do you think that they ask such questions, as opposed to asking for a piece of code to do binary search?

# 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

# www.discretemath.com

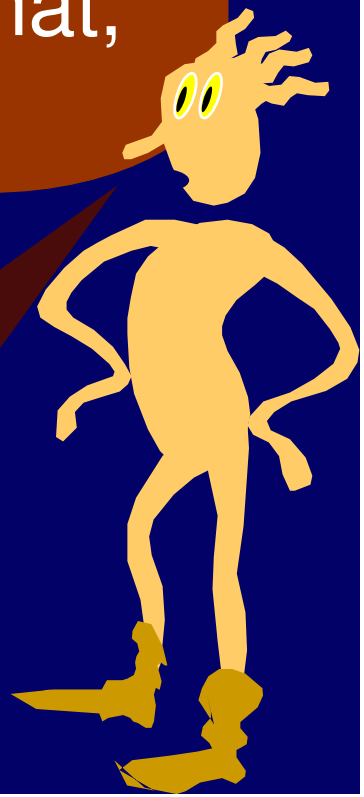
- Content: An up to date grasp of fundamental problems and solutions
- Method: Principles and techniques to solve the vast array of unfamiliar problems that arise in a rapidly changing field



Content, i.e., definitions, formulas, recipes, standard manipulations, I can handle!

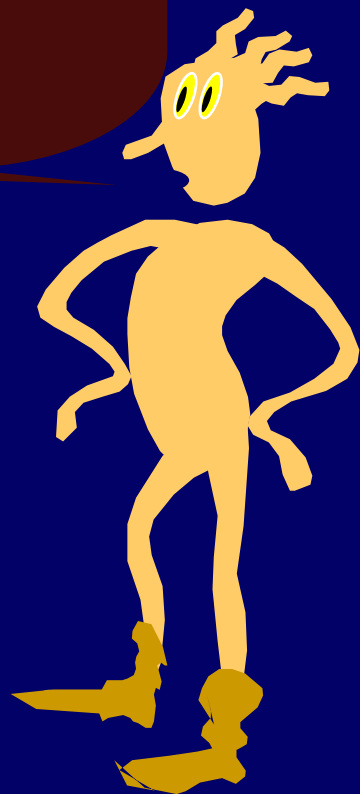
"Method" is intimidating. What if, the plain and simple truth is that, I am not that smart?

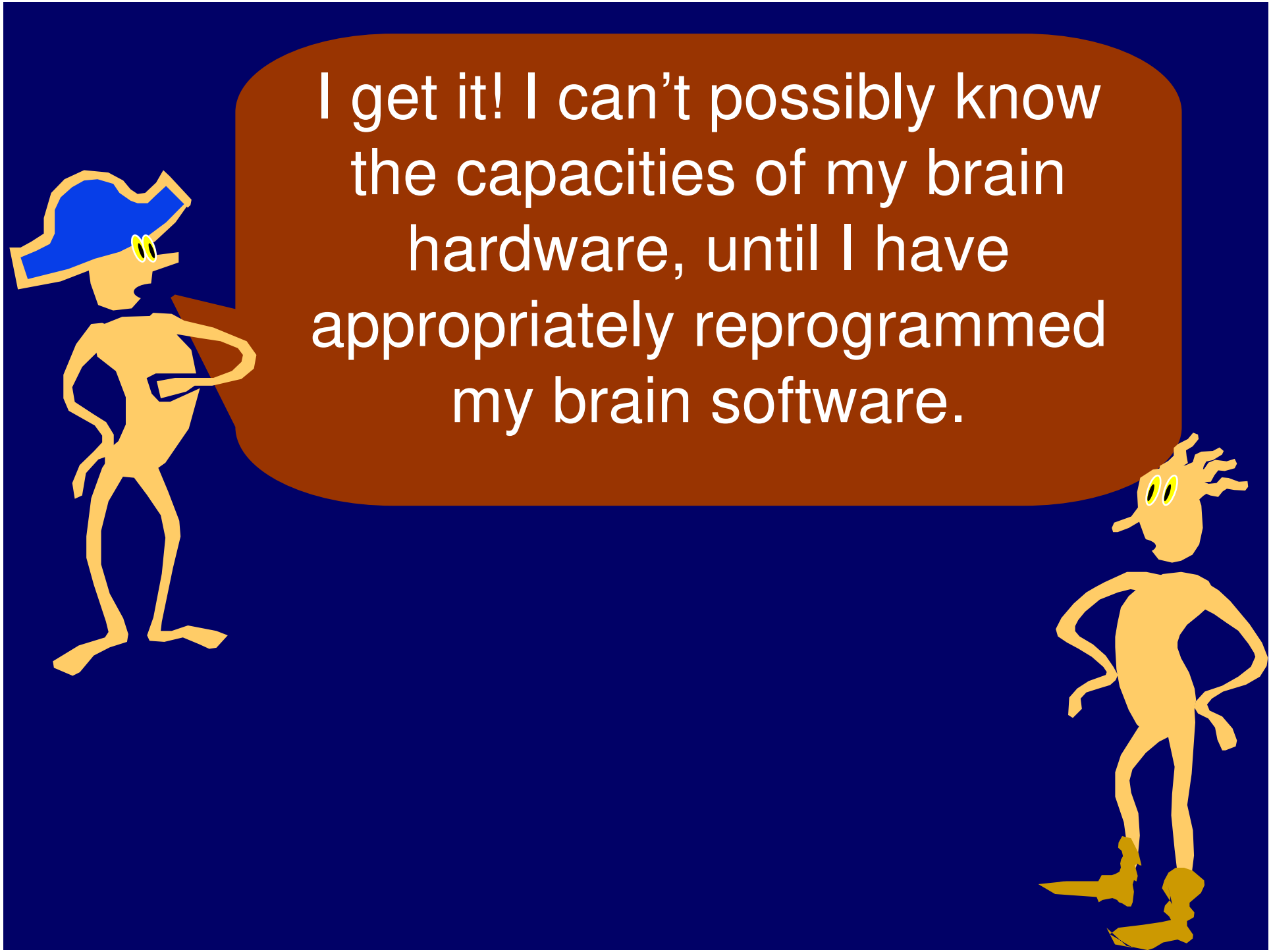
Don't jump to unwarranted conclusions! Clever and resourceful problem solving is a skill that can be taught and refined.



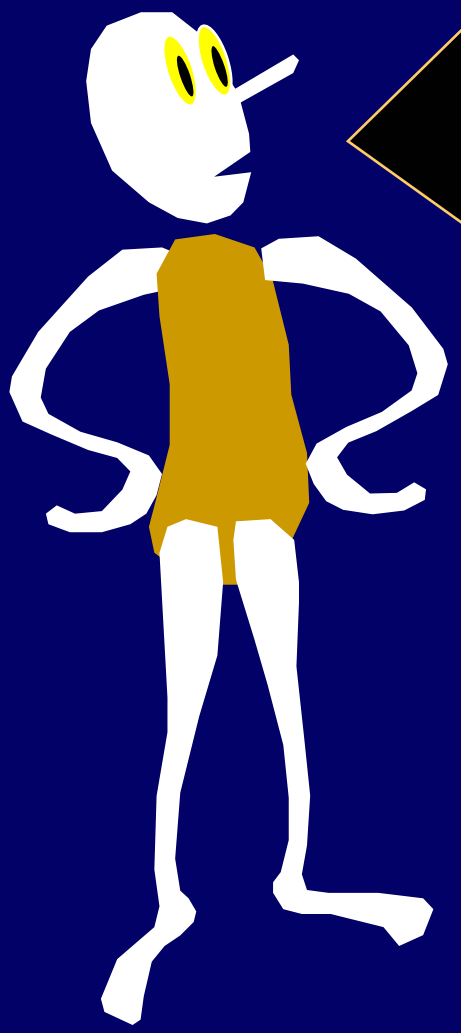
Yeah, but I knows lots of people who don’t need a lecture on problem solving methods.
Brilliance just comes to them.
So you are not a natural?
– What of it? –
Some world class tennis players did not start as natural backhanders. They had to be coached and developed.

Bonzo, I don’t pretend to know the nature of your potential, but I am sure that if you study, practice, and refine your problem solving skills, you will become massively better at it than you are now.




I get it! I can’t possibly know the capacities of my brain hardware, until I have appropriately reprogrammed my brain software.

Aha! I know where the
"aha!" comes from!



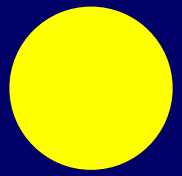
- Representation
- Induction
- Modularity
- Exemplification
- Refinement
- Abstraction
- Bracketing

# Toolkit:

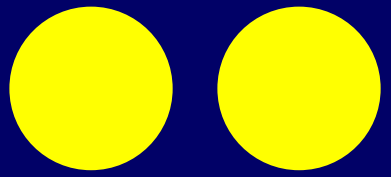
Name abstract objects and ideas, and put them in your toolkit. Know their advantages and limitations.

# Representation:
Understand the relationship between different representations of the same information or idea

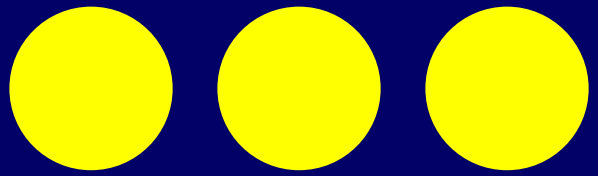
1



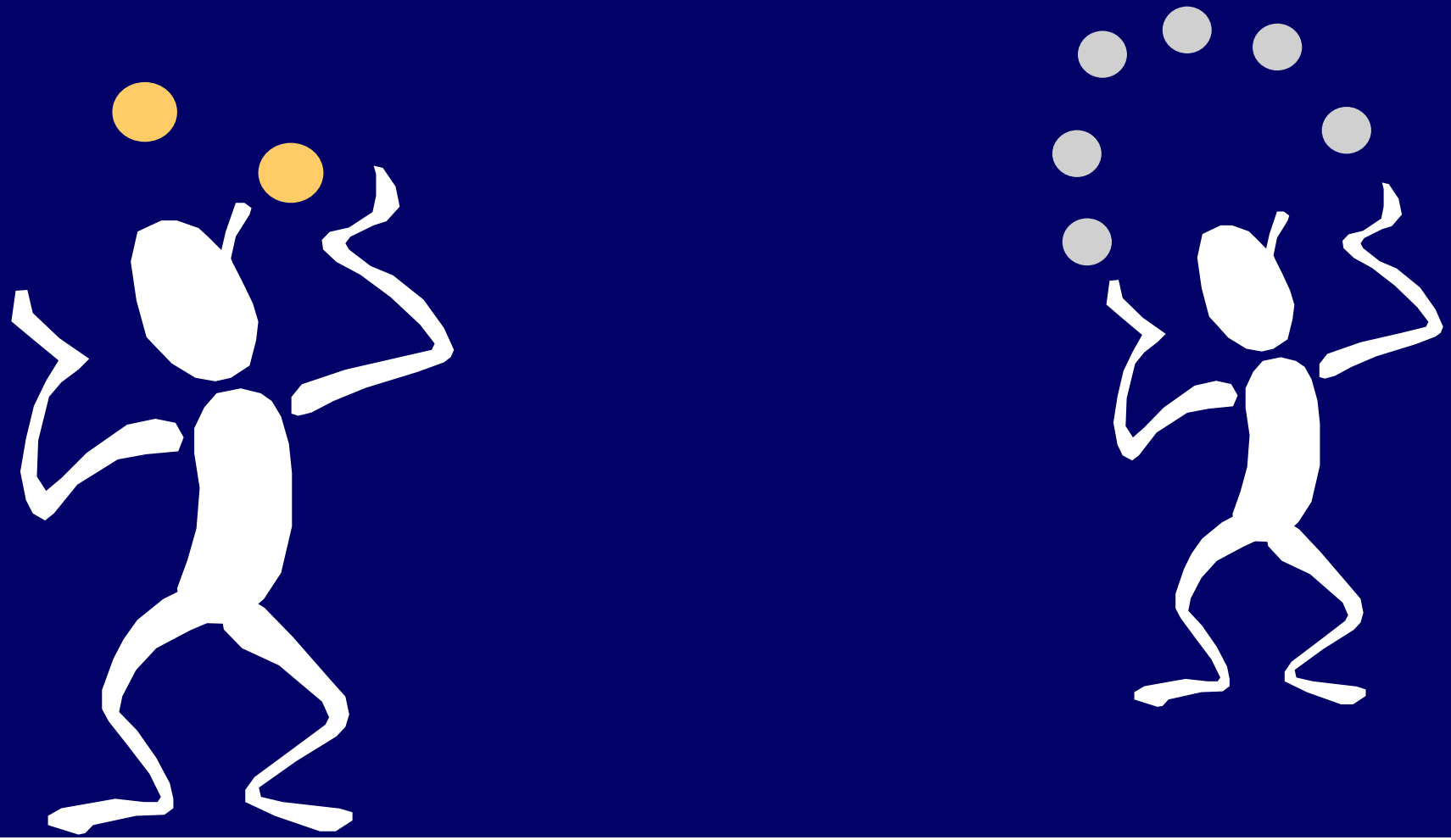
2



3

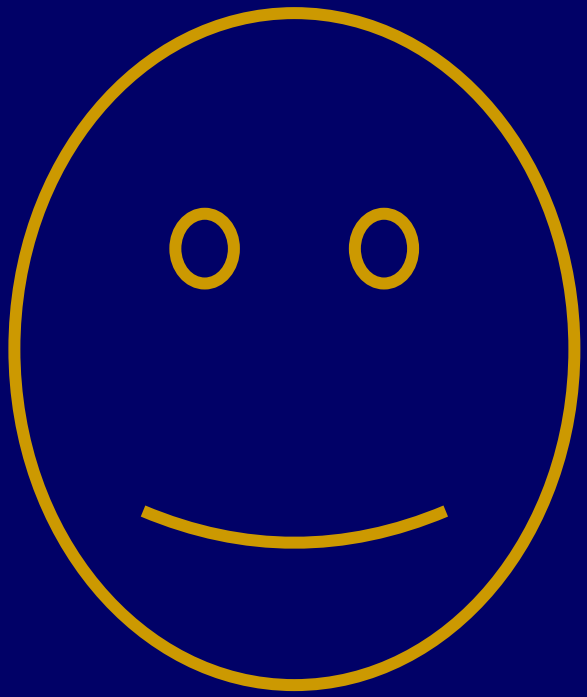


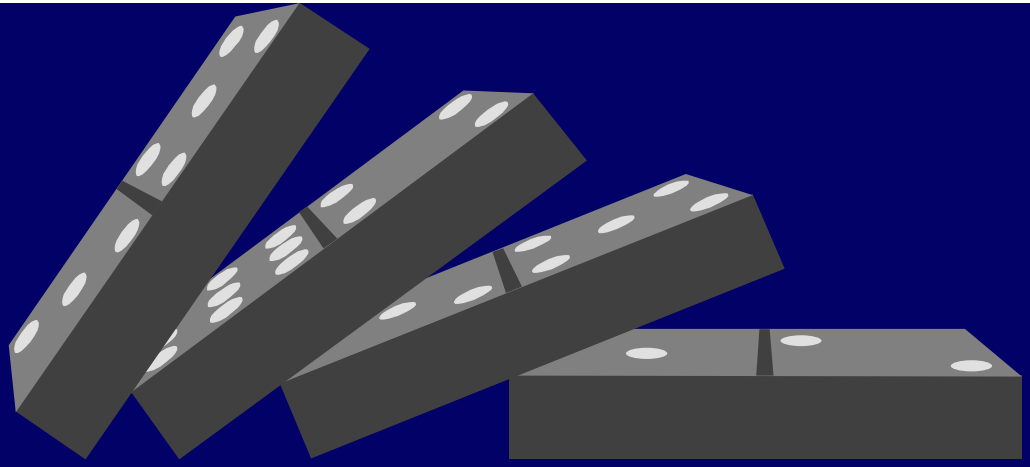
# Exemplification:
Try out a problem or solution on small examples.



# Abstraction:
# Abstract away the inessential features of a problem

=





Induction has many guises.
Master their interrelationship.

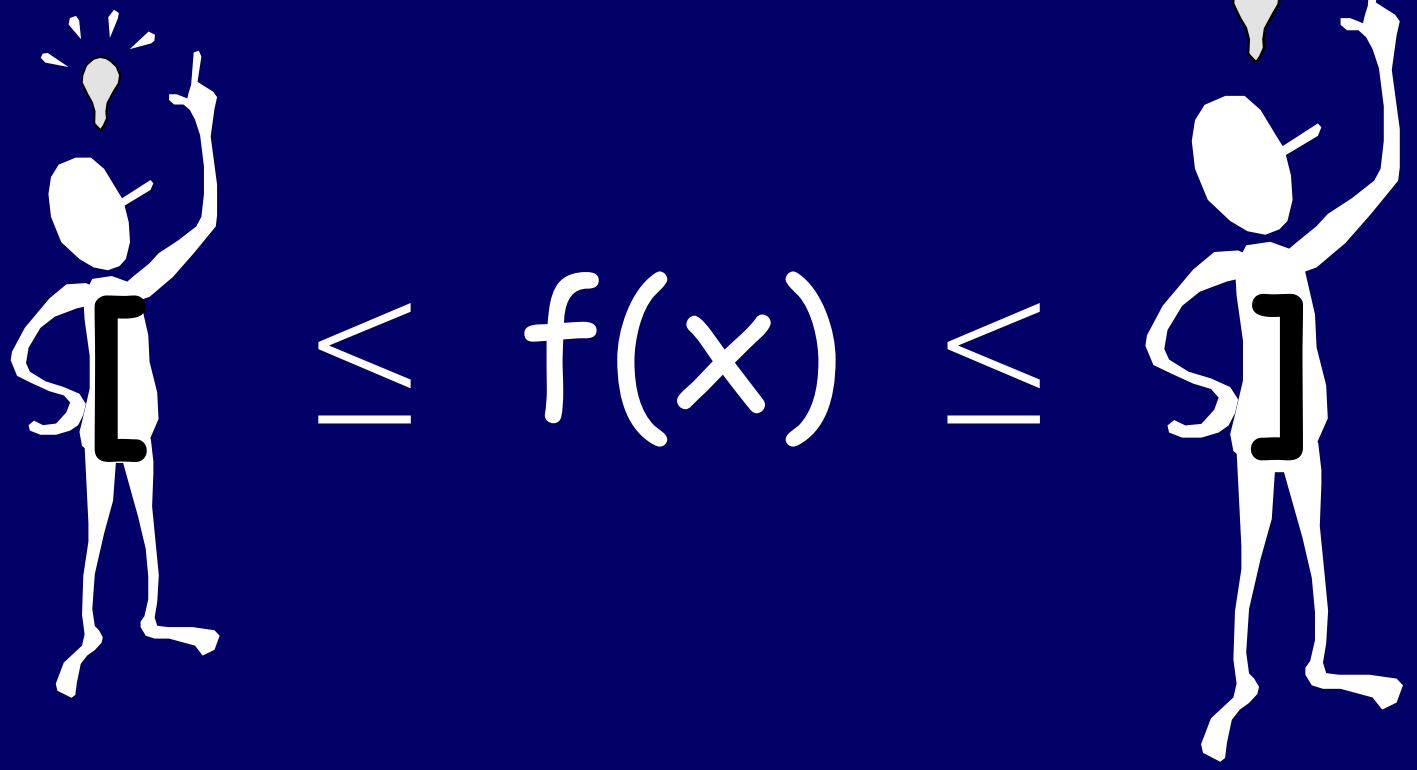
- **Formal Arguments**
- **Loop Invariants**
- **Recursion**
- **Algorithm Design**
- **Recurrences**

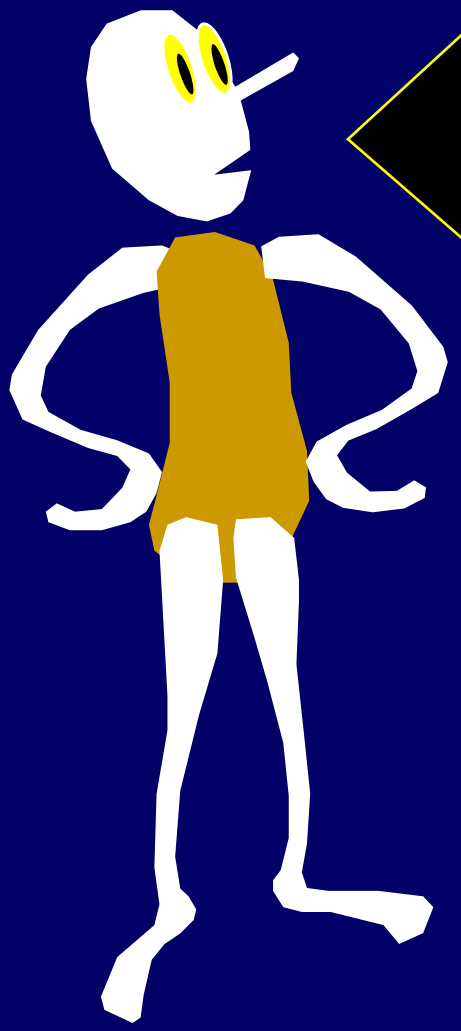
# Modularity:
# Decompose a complex problem into simpler subproblems

# Improvement:

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

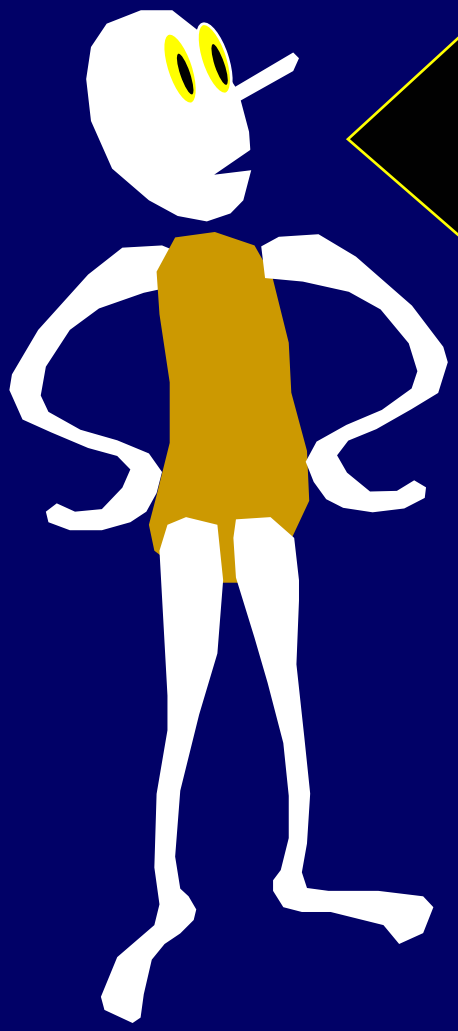
# Bracketing:
# What are the best lower and upper bounds that I can prove?





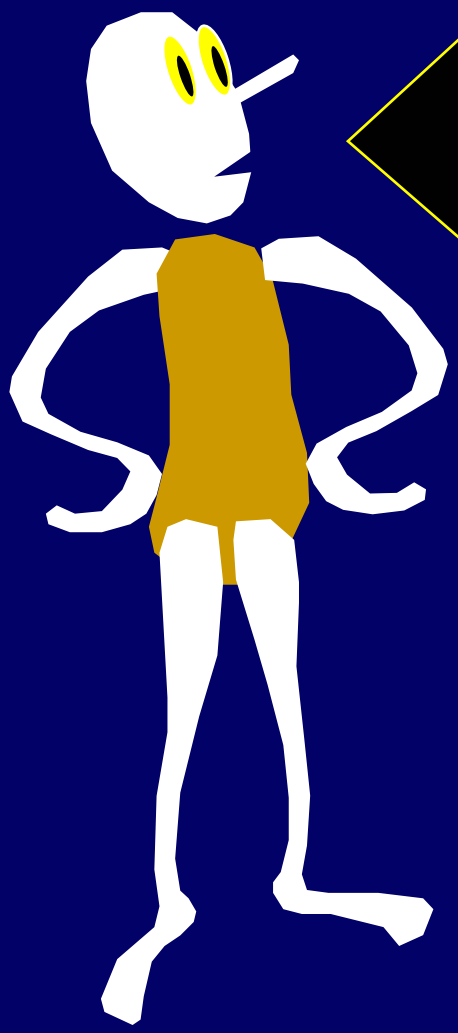
Representation:

This whole course is designed to show you the power of different abstract representations and their relationships.



Don't passively let the representation choose you!

Actively choose the representation!



# Named representations:

Unary & Binary & beyond

Continued Fractions

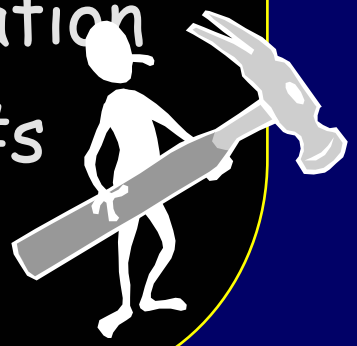
Inductive representation

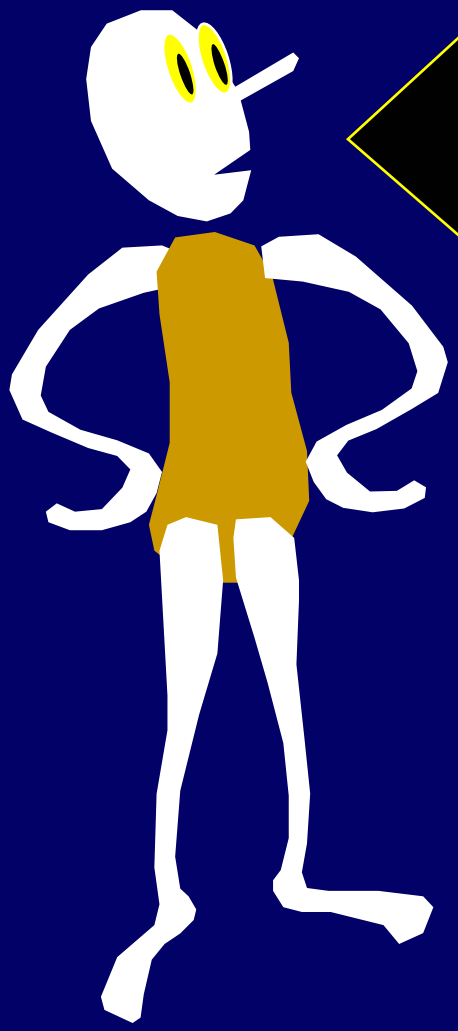
Choice tree representation

Polynomial representation

Block walking Representation

Tiling Rep of Fibonacci #s

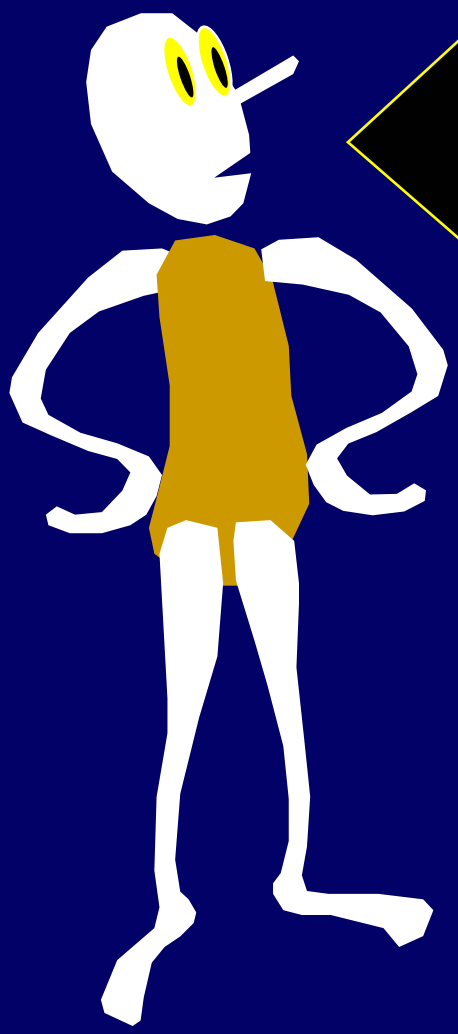




Exemplification:

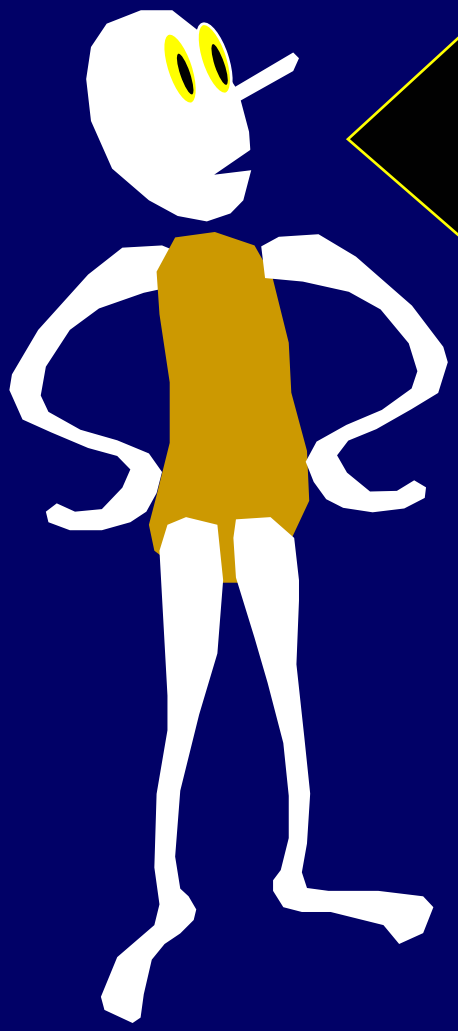
Party handshaking problem.

Perspicacious Pirates problem



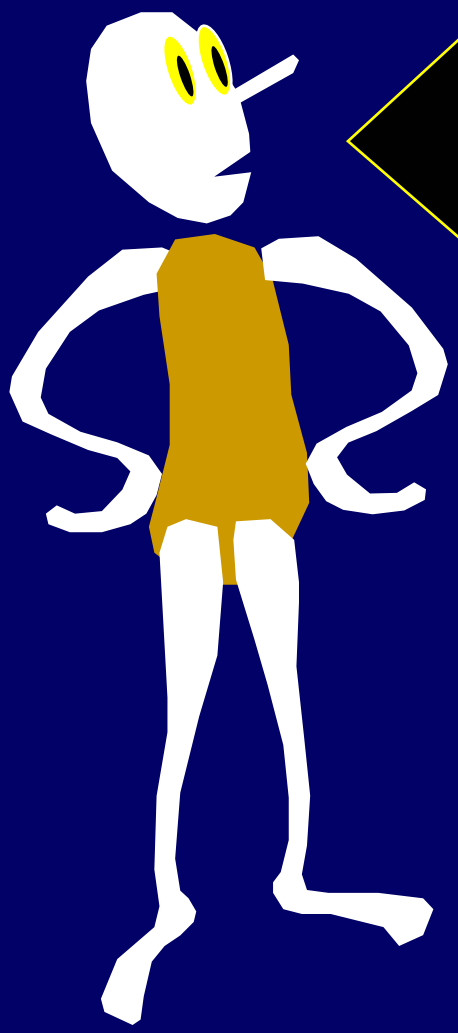
Abstraction:

Raising to power versus addition chains.



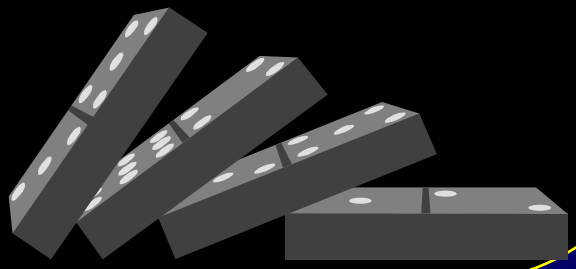
Abstraction:

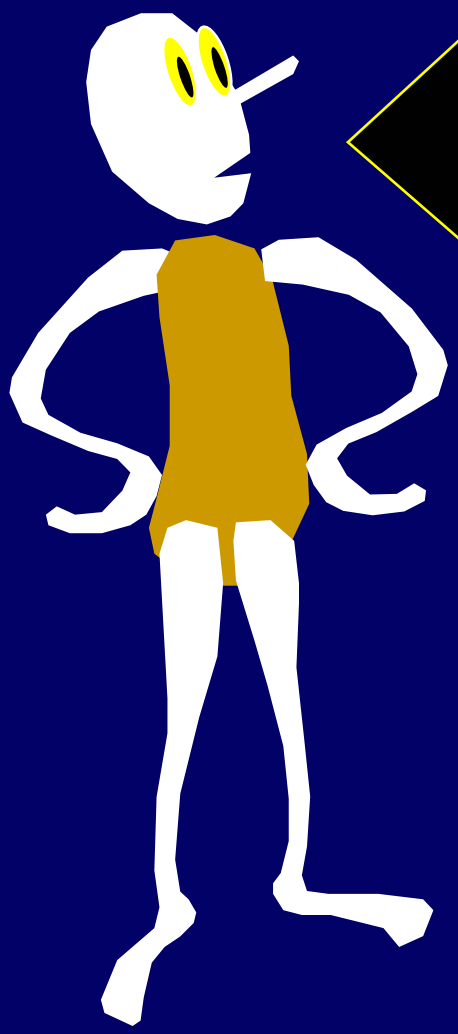
Raising to power works for any binary, associative operation.



Induction!

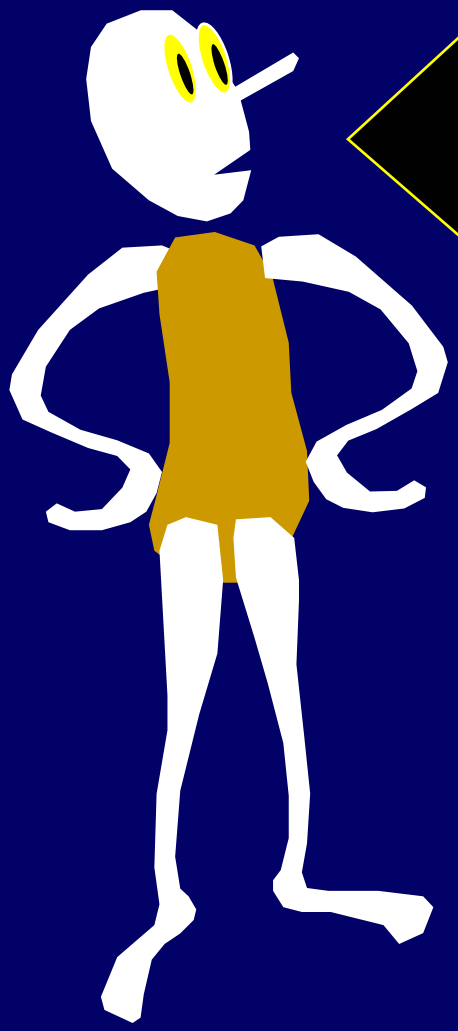
Inductive
Representation/Proof
Modularity
Recursion





Modularity:

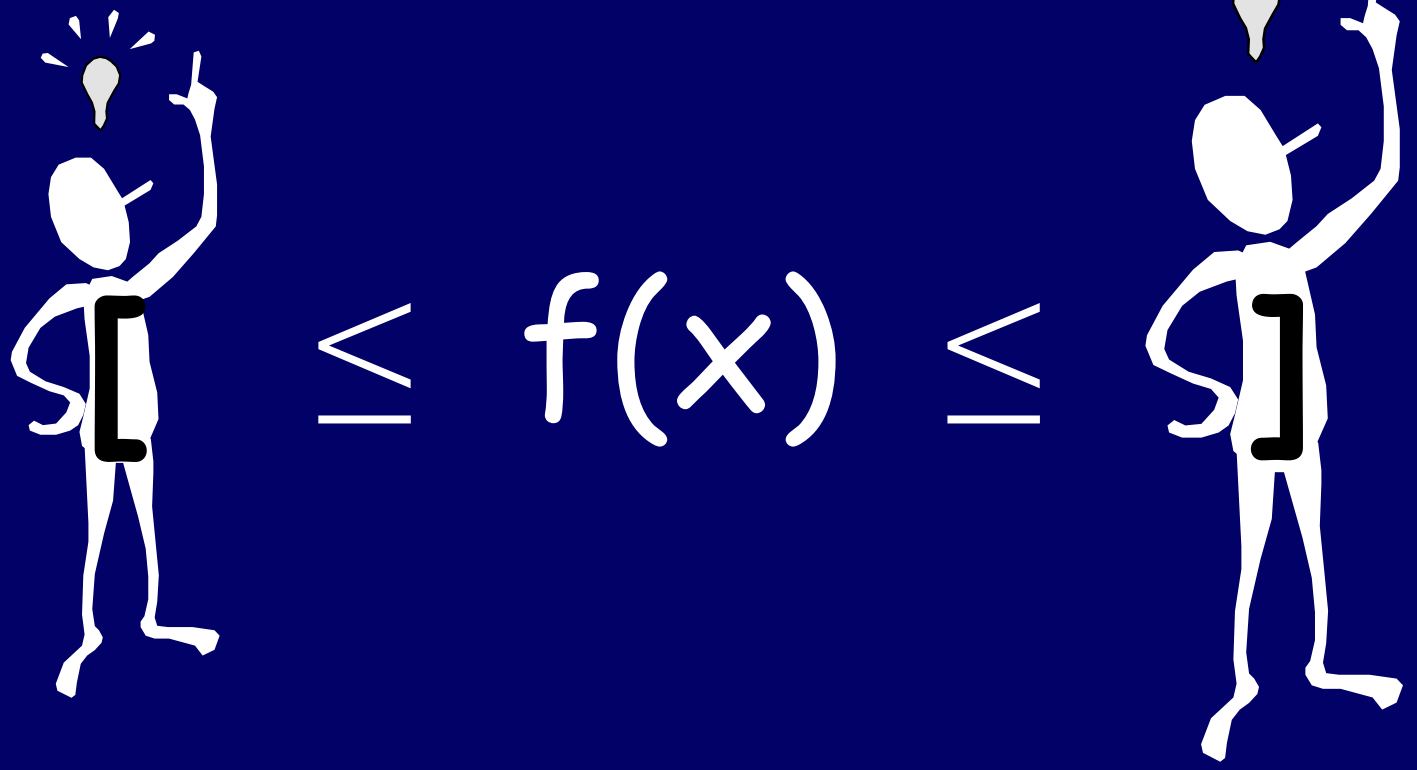
Computer from NAND gates.



Improvement:

Chocolate breaking problem

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



$? \leq M(30) \leq 6$

$? \leq M(n) \leq ?$

**M(n) = numbers of additions required to make n.**

$? \leq M(30) \leq 6$

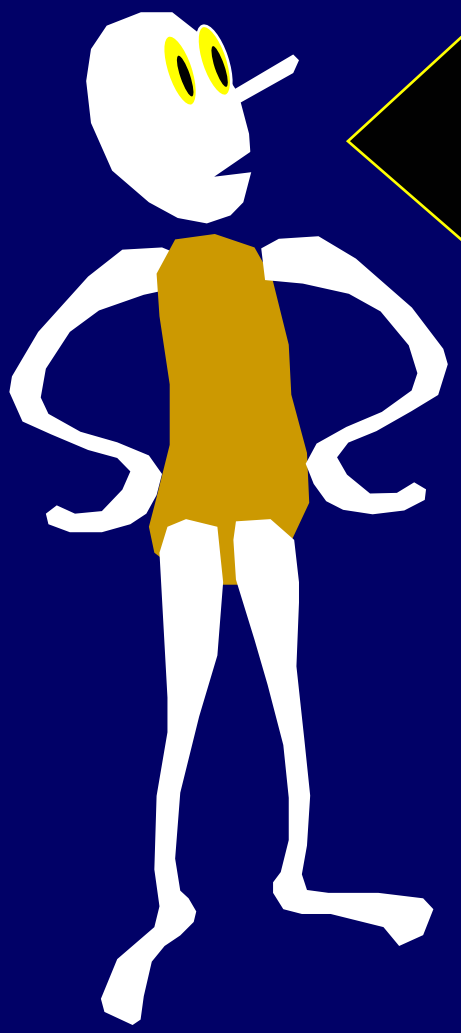
$? \leq M(n) \leq ?$

**FIRST CUT: What brackets can you do off the top of your head?**

$? \leq M(30) \leq 6$

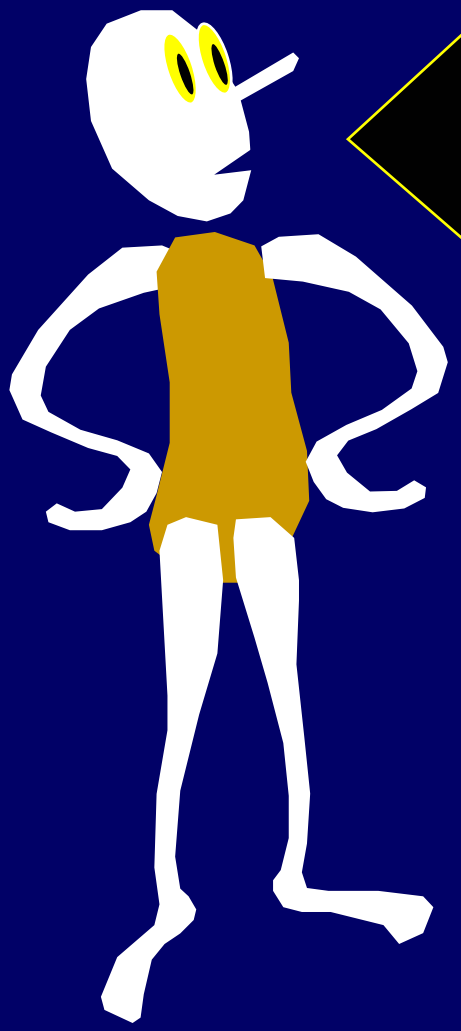
$? \leq M(n) \leq ?$

**SECOND CUT: What brackets can you do in five minutes?**



AHA! Method:

New problem encountered. Possible representation? Try it out on small examples. Any familiar tools come to mind? Get rid of inessential details. Do some quick bracketing. Inductive viewpoint? Avoid mistagging statements.



The first step is to get the problem right!

# 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.

# Try and remove inessential details.

Does the person with the problem still recognize it?

# Conjecture Versus Fact

Keep your thoughts tagged with their certainty status.

YOU ARE YOUR OWN WORST ENEMY!

# Getting To Know The Problem

Try it out on small examples

Articulate a quantity to bracket

What brackets can you do in five minutes?

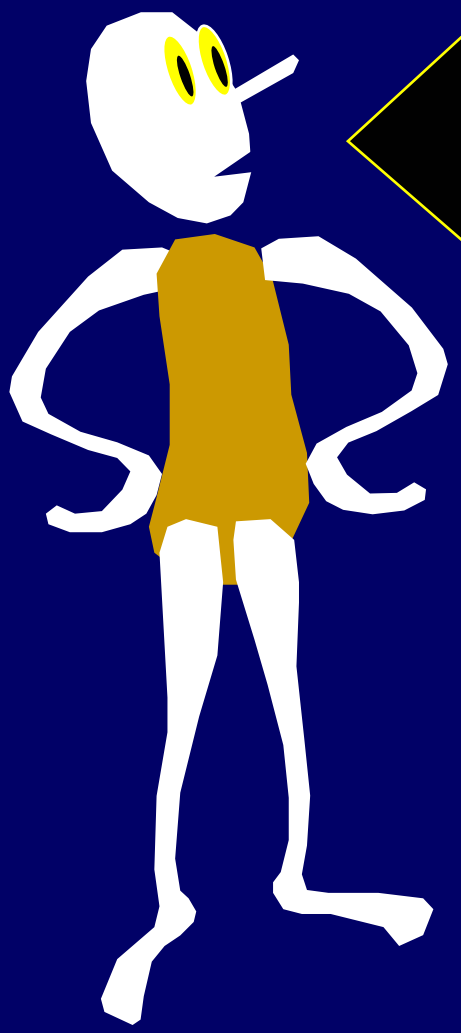
# Choose Your Representation

Think of your toolkit of representations, do any of them seem like the better choice than the original?

Change representations and get to know the problem again (as before)

# Choose Your Representation

What is your representation for your intermediate results?



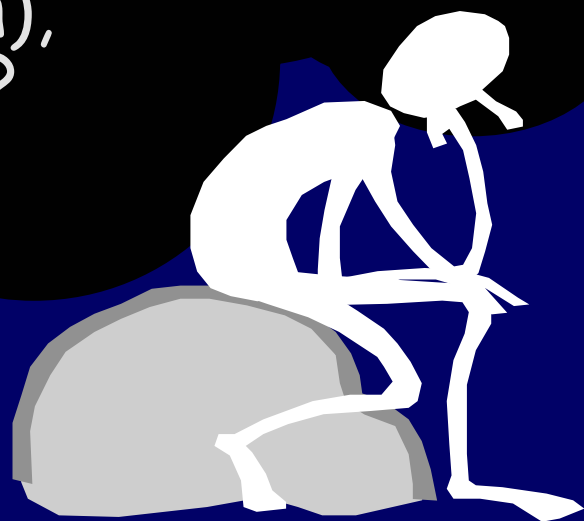
AHA! Method:

New problem encountered. Possible representation? Try it out on small examples. Any familiar tools come to mind? Get rid of inessential details. Do some quick bracketing. Inductive viewpoint? Avoid mistagging statements.

Let $a_0, a_1, a_2, \ldots a_n, X$ be input numbers. Goal is to compute :
$a_n X^n + a_{n-1} X^{n-1} + \ldots + a_1 X + a_0$

At any stage we are allowed to ADD or MULTIPLY any two numbers created so far.

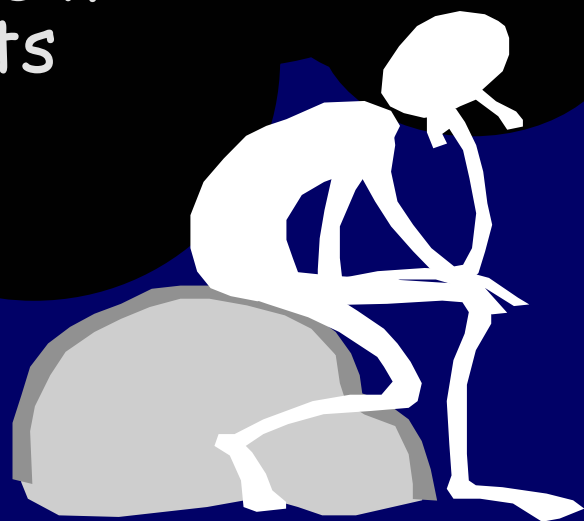
How do we MINIMIZE, f(n), the number of operations?



Let $a_0, a_1, a_2, \ldots a_n, X$ be input numbers. Goal is to compute :
$a_n X^n + a_{n-1} X^{n-1} + \ldots + a_1 X + a_0$

To make any a single expression that is a function of all n+2 inputs will take at least n+1 operations.

We can use n-1 to create the powers of X. Then we can use n to multiply each power by its coefficient. We can use n additions to add them up.
$n+1 \leq f(n) \leq 3n - 1$

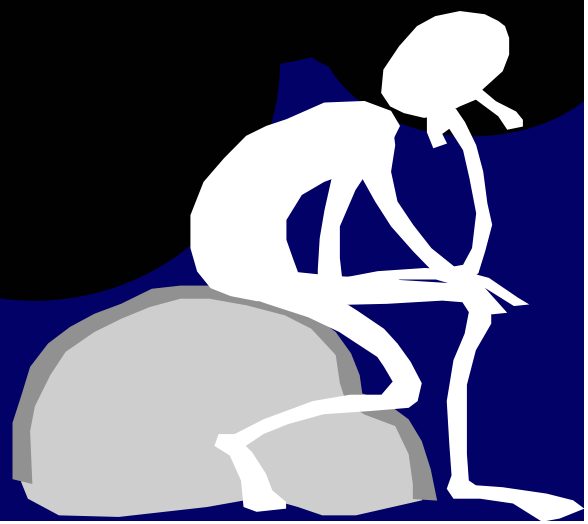


Let $a_0, a_1, a_2, \ldots a_n, X$ be input numbers. Goal is to compute :

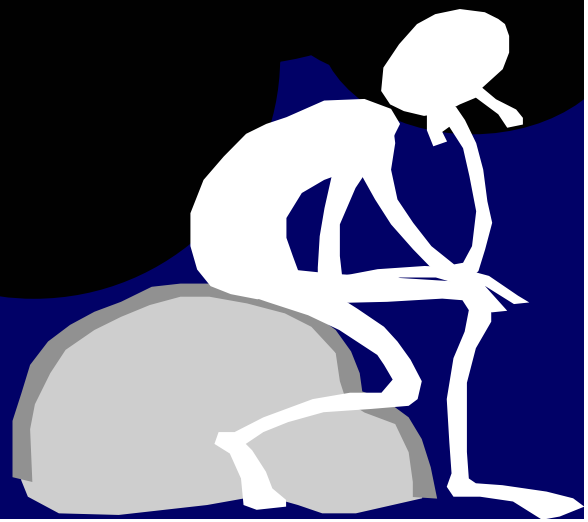
$$a_n X^n + a_{n-1} X^{n-1} + \ldots + a_1 X + a_0$$

Can I get better brackets on f ?

$$n+1 \leq f(n) \leq 3n - 1$$



In polynomial representation I can think of all the intermediate results as polynomials.



$$P_k(X) = a_n X^k + a_{n-1} X^{k-1} + \ldots + a_{n-k+1} X + a_{n-k}$$

Goal: Compute $P_n(X)$

Inductive Goal (Invariant):
Compute $P_0(X), P_1(X), \ldots p_n(X)$

Slick code [Loop invariant $P==P_i(X)$]

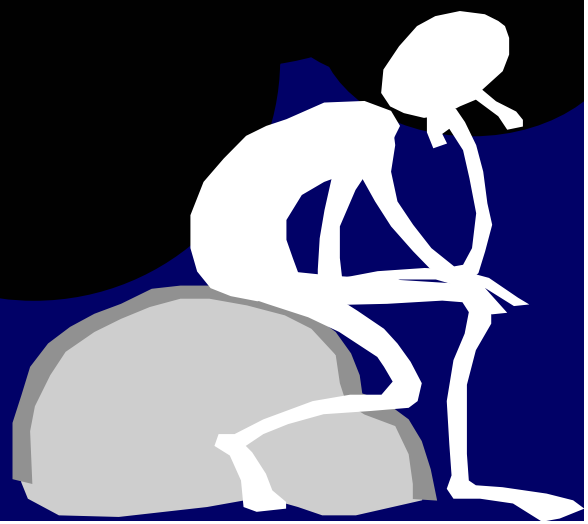
$P := a_n$;
For i = 1 to n do $P := P*X + a_{n-i}$

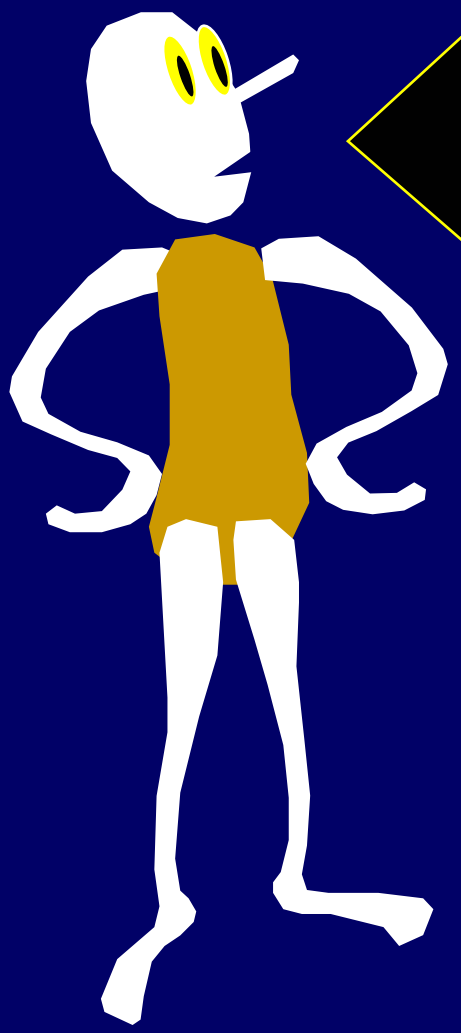
TOTAL: n mults and n additions. $f(n) \leq 2n$

Let $a_0, a_1, a_2, \ldots a_n, X$ be input numbers. Goal is to compute :
$a_n X^n + a_{n-1} X^{n-1} + \ldots + a_1 X + a_0$

At any stage we are allowed to ADD or MULTIPLY any two numbers created so far.

$n+1 \leq f(n) \leq 2n$





AHA! Method:

New problem encountered. Possible representation? Try it out on small examples. Any familiar tools come to mind? Get rid of inessential details. Do some quick bracketing. Inductive viewpoint? Avoid mistagging statements.

A volunteer, please.

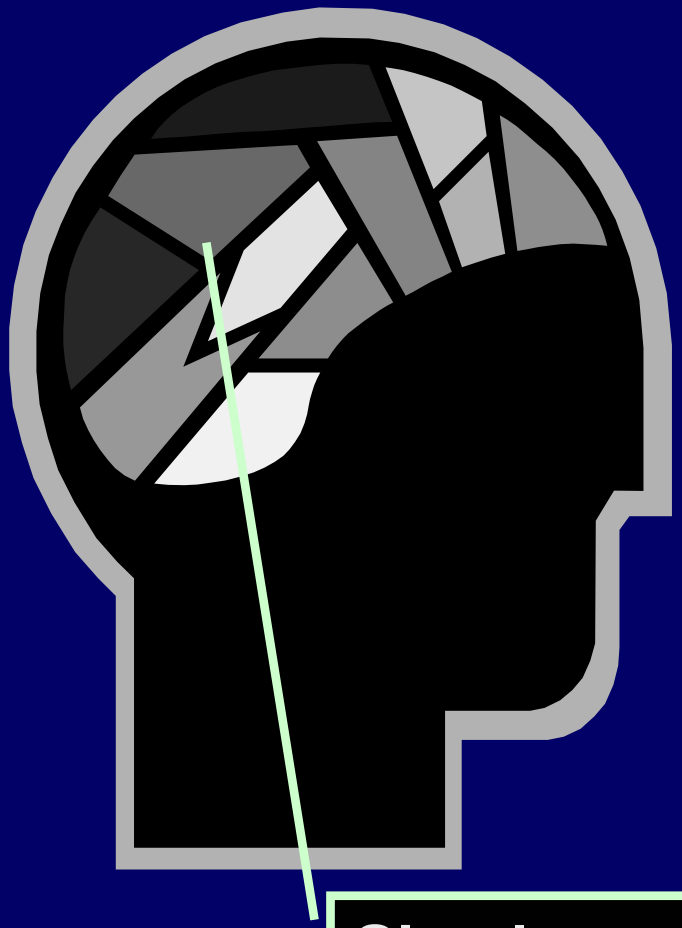
# Martial Arts 101

- The novice makes a **huge** motion
- The black belt makes a small motion
- The master makes a tiny motion

Violin, piano, tennis, magic, programming, singing, . . .

- The novice makes a **huge** motion
- The black belt makes a small motion
- The master makes a tiny motion

# Scanning the brains of master problem solvers



The better the problem solver, the less brain activity is evident. The real masters show almost no brain activity!

**Simple and to the point**

# Scanning the brains of master problem solvers

The expert represents his/her state as simply as possible.

# Value Simplicity

The root of effective thinking, effective science, effective mathematics, effective engineering is to keep the issues simple.