

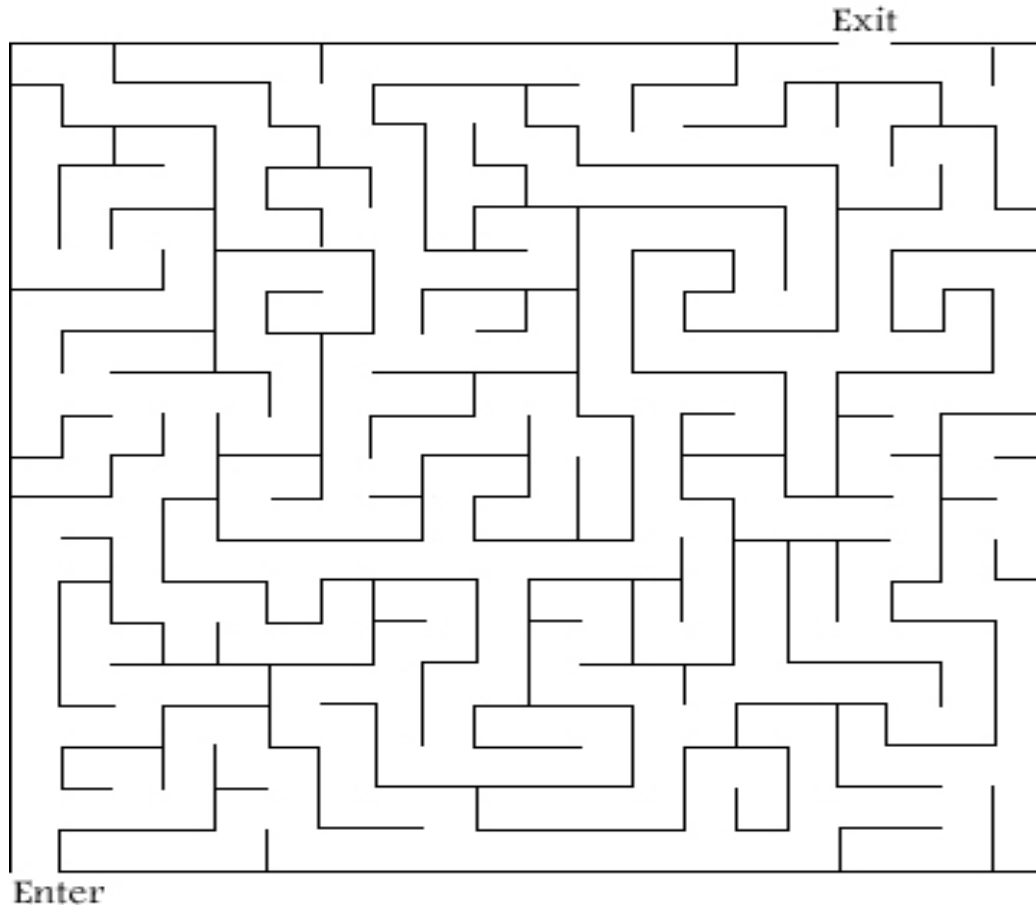
**Welcome
back!**

Problem Solving

Day 2, Session 1

The purpose of computing
is *insight*, not numbers.

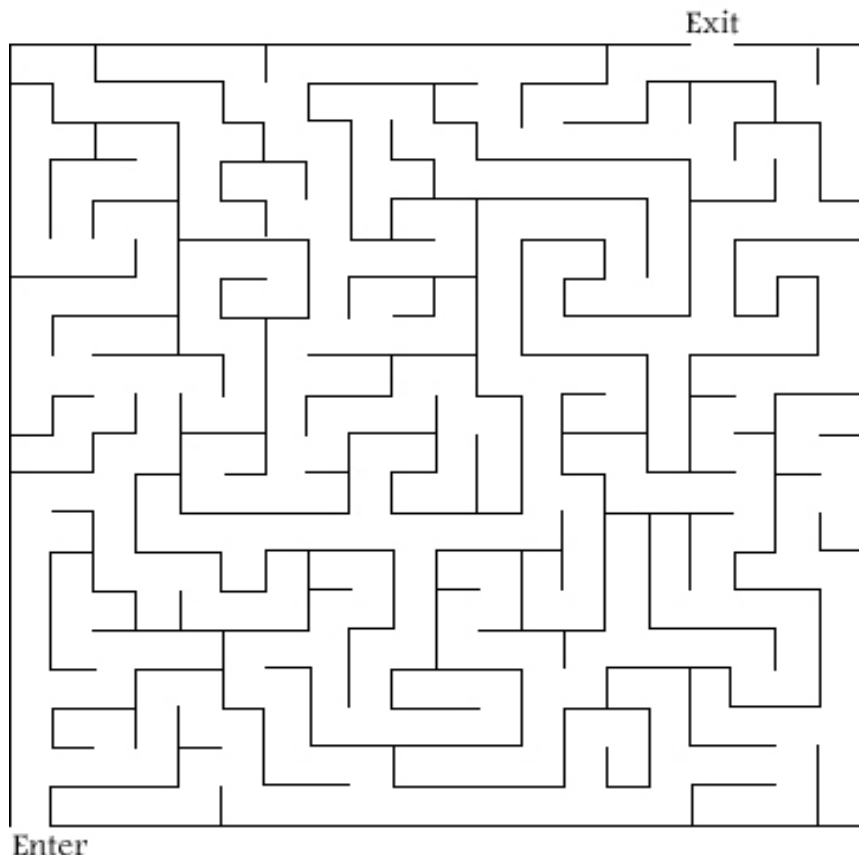
A problem-solving *process*



Here's a familiar problem.

How would you solve it?

A problem-solving *process*



The *process* is much more useful than one solution!

solving one vs. solving many

(1) experiment to understand

-
-
-
-

(2) describe a plan

-
-
-
-

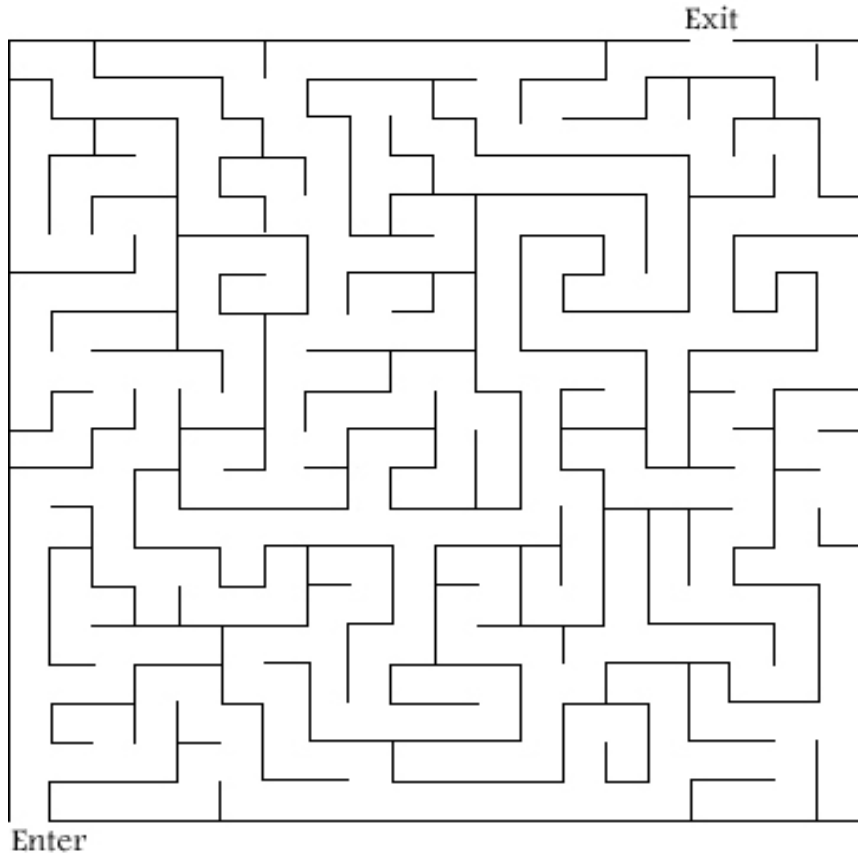
(3) test the plan (*execute* it)

-

(4) reflect and evaluate

-
-
-
-

A problem-solving *process*



(1) experiment to understand

- how to start?
- **what's the goal?**
- try a strategy... and/or another
- try smaller pieces

(2) describe a plan

- first step
- next step...
- **are there choices?**
- if so, how do you handle them

(3) test the plan (*execute* it)

- **go through the steps**

(4) reflect / evaluate

- does it work in this case?
- would it always work?
- **what are hard or easy cases?**
- **what are other applications?**

A problem-solving *process*

3	4	1	3	1
3	3	3	G	2
3	1	2	2	3
4	2	3	3	3
4	1	4	3	2

"rook-jumping maze"

What is the *path* from the ring to the G?

A problem-solving *process*



3	4	1	3	1
3	3	3	G	2
3	1	2	2	3
4	2	3	3	3
4	1	4	3	2

What is the *path*?

What is the *process*?

(1) experiment to understand

- how to start?
- **what's the goal?**
- try a strategy... and/or another
- try smaller pieces

(2) describe a plan

- first step
- next step...
- **are there choices?**
- if so, how do you handle them

(3) test the plan (execute it)

- **go through the steps**

(4) reflect / evaluate

- does it work in this case?
- would it always work?
- **what are hard or easy cases?**
- **what are other applications?**

Another example...

Brick-breaking of chocolate...!



Q: How many breaks are needed to fully separate these pieces?

Break it up...



<i># of breaks</i>	<i># of pieces</i>
1	
2	
	12
<i>N</i>	
	<i>X</i>

Examples from ECS/MyCS



There are 10 people (including you) at a party.

Q: How many handshakes do **you** need in order to greet everyone?

<i># in group</i>	<i># of h.shakes needed</i>
2	
3	
10	

Q: How many handshakes are needed so **everyone** greets everyone?

<i># in group</i>	<i># of h.shakes needed</i>
2	
3	
10	

And another...



Recognize this game show?

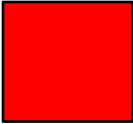
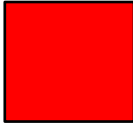
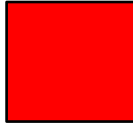
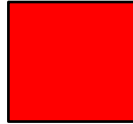
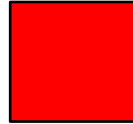

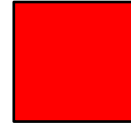
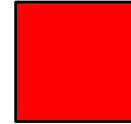
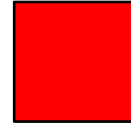
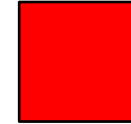


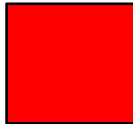
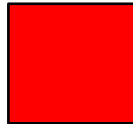



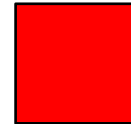




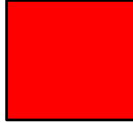
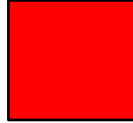




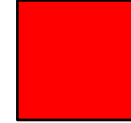
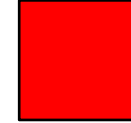


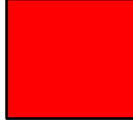
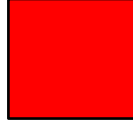


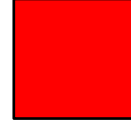
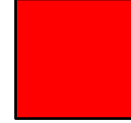
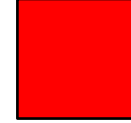
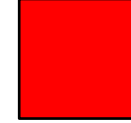







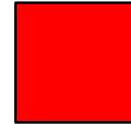
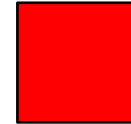
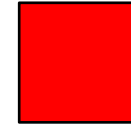


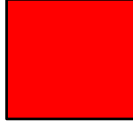





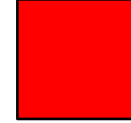
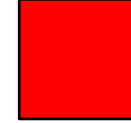


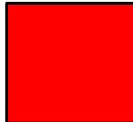
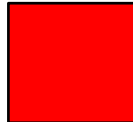



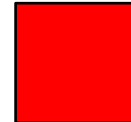


<http://www.youtube.com/watch?v=WKR6dNDvHYQ>

Let's Make a Deal...

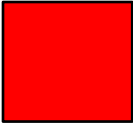
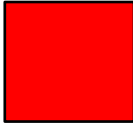
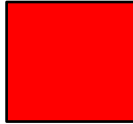
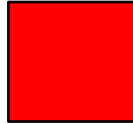
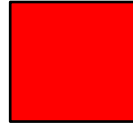

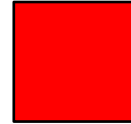
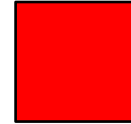
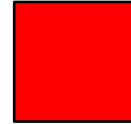
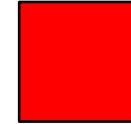


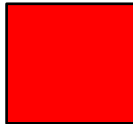
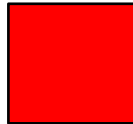



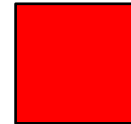

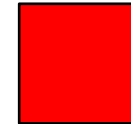


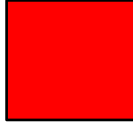
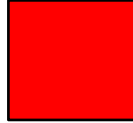




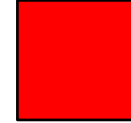
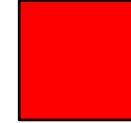


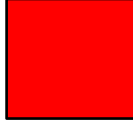
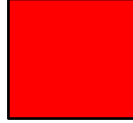


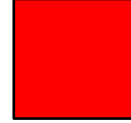
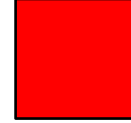
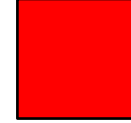
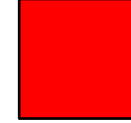


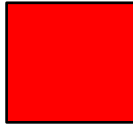




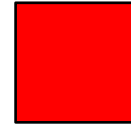
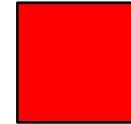
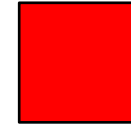


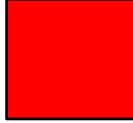





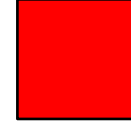
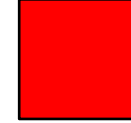


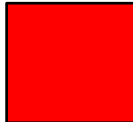
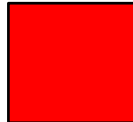



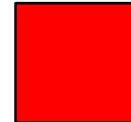




Choose... Reveal... Switch or stay?

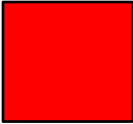
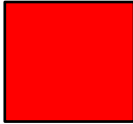
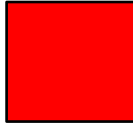
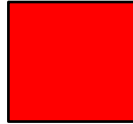
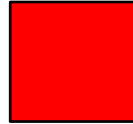

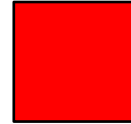
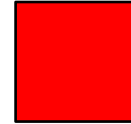
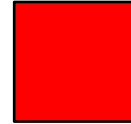
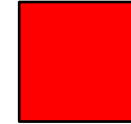


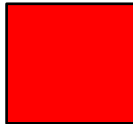
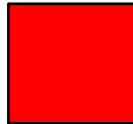



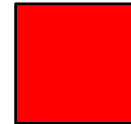




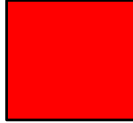
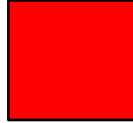




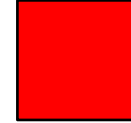
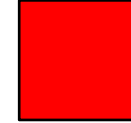


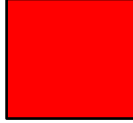
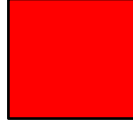


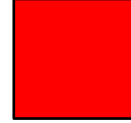
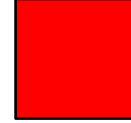
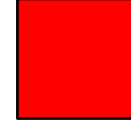
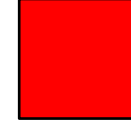


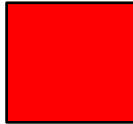




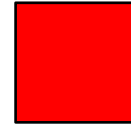
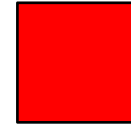
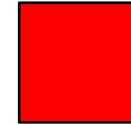


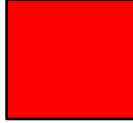





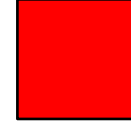
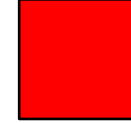


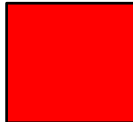
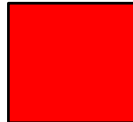



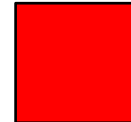


Let's Make a *bigger* Deal...

	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										

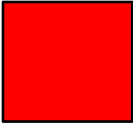
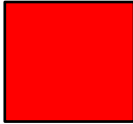
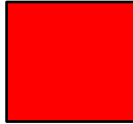
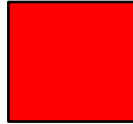
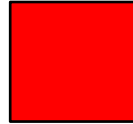

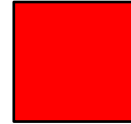
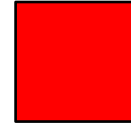
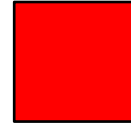
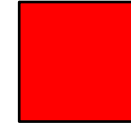


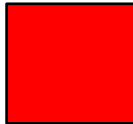
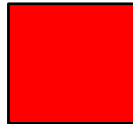



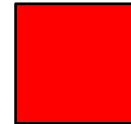




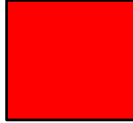
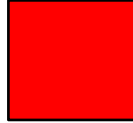




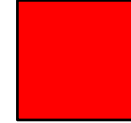
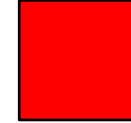


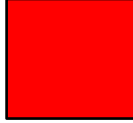
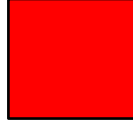


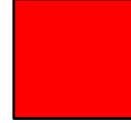
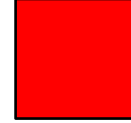
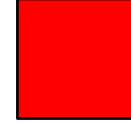
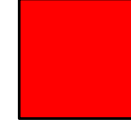


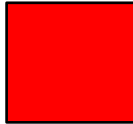




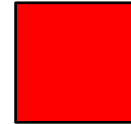
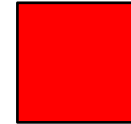
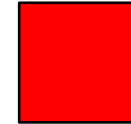


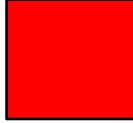





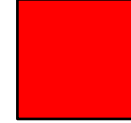
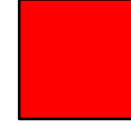


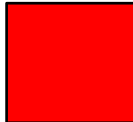
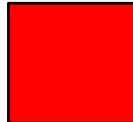



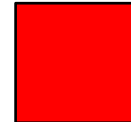


Let's Make a *bigger* Deal...

	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										

Let's Make a *bigger* Deal...

	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										

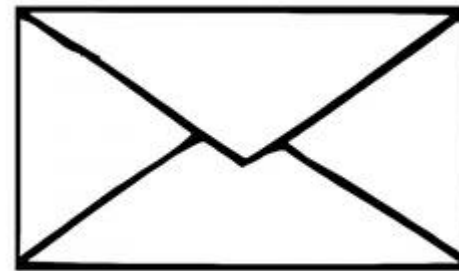
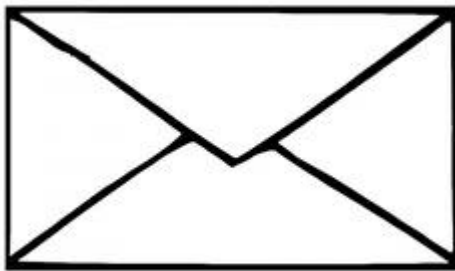
Let's Make a *bigger* Deal...

	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										

Two envelopes...



Both envelopes have some money in them. You know that one envelope has double the amount in the other envelope.



Suppose you choose an envelope and you find **M** money in it. Say, \$20.

Now you have the opportunity to trade.

Should you?

Take-home message

Understanding problems is the key to solving them -- and it's *much more important* than solving a particular one!

Computer Science is the science of analyzing (and solving) *information-based* problems, i.e., those based on **data**.

or puzzles

could these puzzles work with middle-schoolers?

Looking forward...

4-7 days
spiraling

MyCS

Middle-years
Computer
Science

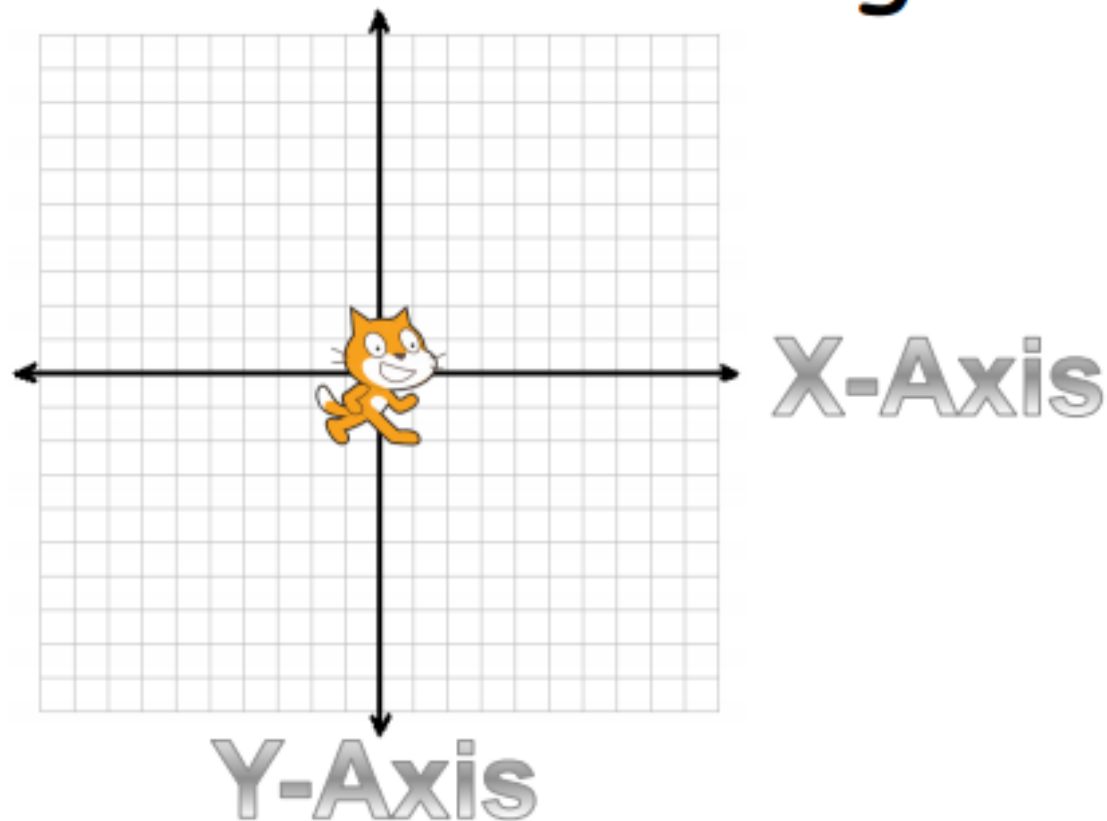
Today's plan

Day 1	CS: what and why?
	Scratch ~ sound
	What is a computer?
	Scratch ~ movement
Day 2	CS as problem-solving
	Encoding data
	Scratch ~ coordinates
Day 3	<i>Algorithms ?!</i>
	Scratch ~ if/else
	Limits of computers
	Scratch ~ projects
Day 4	Using the web <i>well</i>
	Web: HTML/CSS
	Web safety/security
	Web: Javascript
Day 5	Games and extras

Scratch: Position

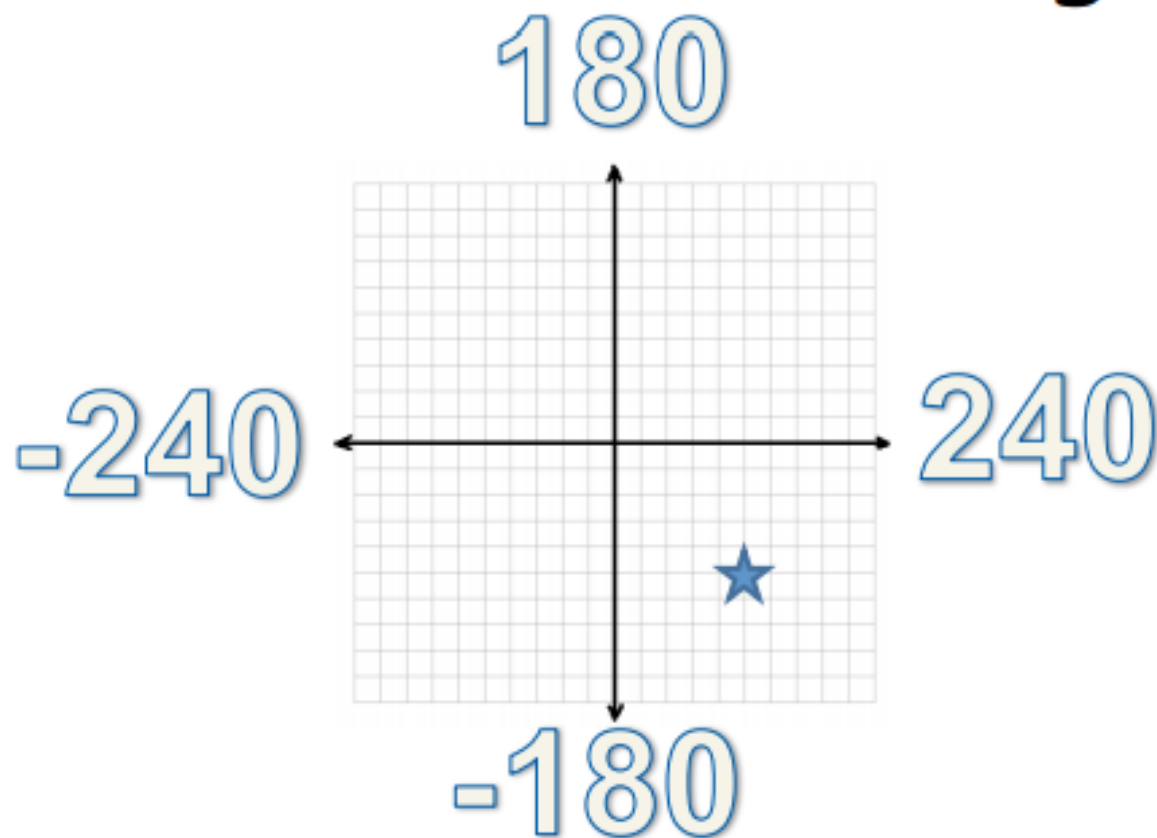
Day 2, Session 2

Position on the Stage!



In Scratch, the stage is actually a big X/Y coordinate plane. Like a graph!

Position on the Stage!



The x-axis (side to side) goes from -240 to 240 .
The y-axis (up and down) goes from -180 to 180 .

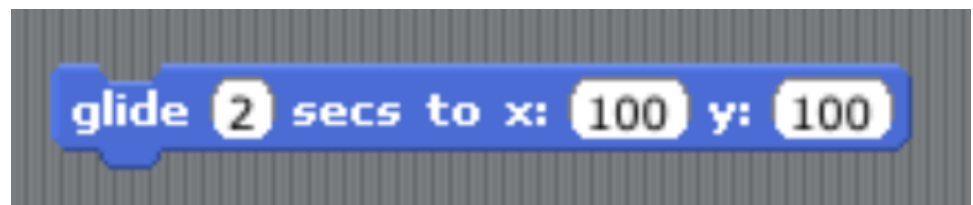
Go To X Y



The go to x y block makes the sprite move to the x and y coordinates you provide. It has 2 variables:

1. You can change the position on the x axis.
2. You can change the position on the y axis.

Want smooth motion?



Coordinate Practice



Review...



Coordinates in Scratch Worksheet Name: _____

Answer the following questions WITHOUT using Scratch.

For each question, draw the shape on the stage, then label each point, then draw the script you would use to draw that picture.

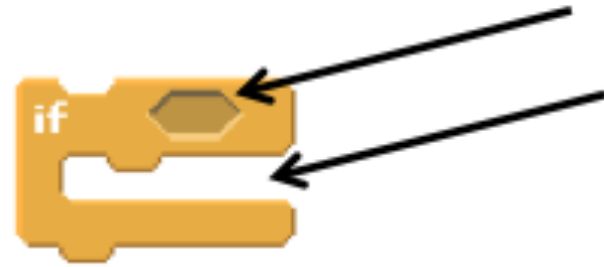
New...

Coordinates in Scratch Worksheet Name: _____

Answer the following questions using Scratch and the  block or the  block.

For each question, draw the picture on the stage write a script to draw the picture ACCURATELY, then draw your final script.

If Block



The "if" block can be found in the control tab.

The if block means, "**IF** this happens, **THEN** do this."

You can put blocks **ON** the if block and **IN** the if block.

Sensing Block



Most of the blocks that fit ON the if block are **sensing** blocks, which can be found in the sensing tab.

Different sensing blocks can detect if a **key** is pressed, if the sprite is **touching** something, or if the sprite moves to a certain **position**.

More Debugging



Review...

Debugging 1 Worksheet

Name: _____

For each question, explain why the script does not work (where is the bug?) and draw a debugged script (one that works correctly).

1. What is a "bug"?

Debugging 2 Worksheet

Name: _____

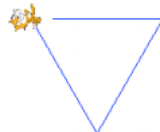
For each question, explain why the script does not work (where is the bug?) and draw a debugged script (one that works correctly).

1. We want the sprite to draw a triangle. Draw a circle around the bug(s), then explain and fix the problem.

Buggy Script

```
pen down
move 100 steps
turn 120 degrees
move 120 steps
turn 120 degrees
move 120 steps
```

Result



Debugged Script

Explanation of bug(s):

New...

Binary, Bits, and Encoding

Day 2, Session 3

How can we **represent** data in a single, consistent way so *that machines can process it* ?

How do numbers really work?

Decimal is based on the number **10**.



Why 10 and not 12?

How do numbers really work?

Binary is based on the number **2**.

Why 2 instead of 10, or 1, or 3?



Binary bubbles...?!



Try these!

Convert from binary to decimal:

Binary

Decimal

1101

1001

101010

1100010

Convert from decimal to binary:

12

25

33

111

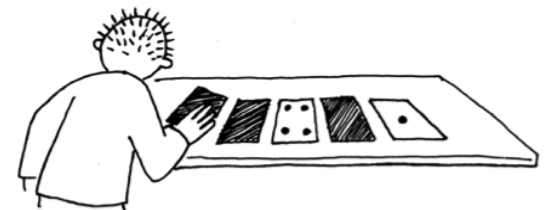


Encoding/Decoding *Text*

1	2	3	4	5	6	7	8	9	10	11	12	13
a	b	c	d	e	f	g	h	i	j	k	l	m
14	15	16	17	18	19	20	21	22	23	24	25	26
n	o	p	q	r	s	t	u	v	w	x	y	z

beepin' ...

boopin' ...

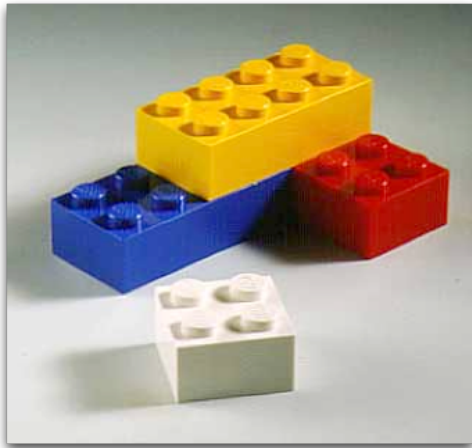


Encoding/Decoding Text

1	2	3	4	5	6	7	8	9	10	11	12	13
a	b	c	d	e	f	g	h	i	j	k	l	m
14	15	16	17	18	19	20	21	22	23	24	25	26
n	o	p	q	r	s	t	u	v	w	x	y	z

Room to work

Key insight



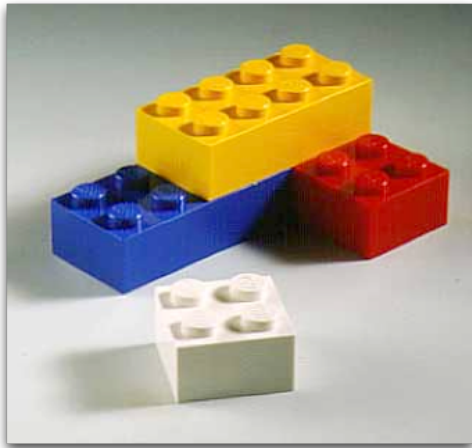
Could you represent a
lego tower in binary?

All information can be represented as a combination of 1's and 0's.

If you know **how it's encoded**, you can then **decode it** to get back the original information!

Binary encodings are just *one* way to do it.

Lego Encoding



Challenge:

First, **encode** a lego structure with binary numbers.

Next, **swap** with another pair.

Then, **decode** their numbers to build the structure -- they *should* be the same...

Lego example!



Here is our encoding:

color	binary
red	0
yellow	1

brick type	binary
2X3	0
2X2	1

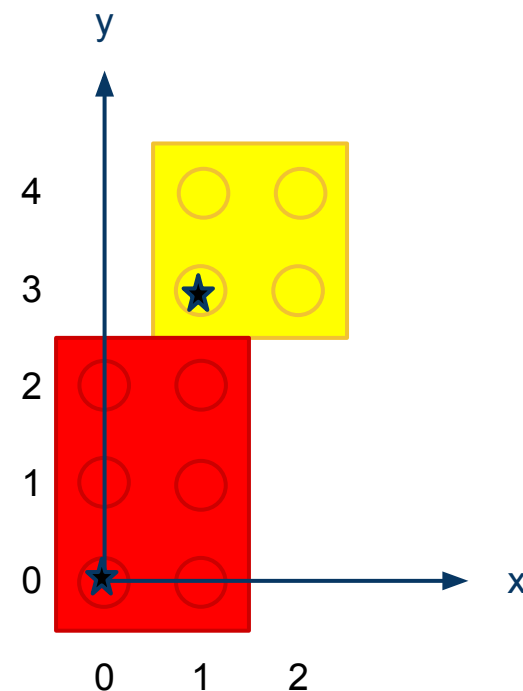
orientation	binary
horizontal	0
vertical	1

Using it, we can encode these bricks:

x- coordinate	y- coordinate	color	brick type	orientation
0	0	red	2X3	vertical
1	3	yellow	2X2	horizontal



x- coordinate	y- coordinate	color	brick type	orientation
00	00	0	0	1



Lego example!



Here is our encoding:

color	binary
red	0
yellow	1

brick type	binary
2X3	0
2X2	1

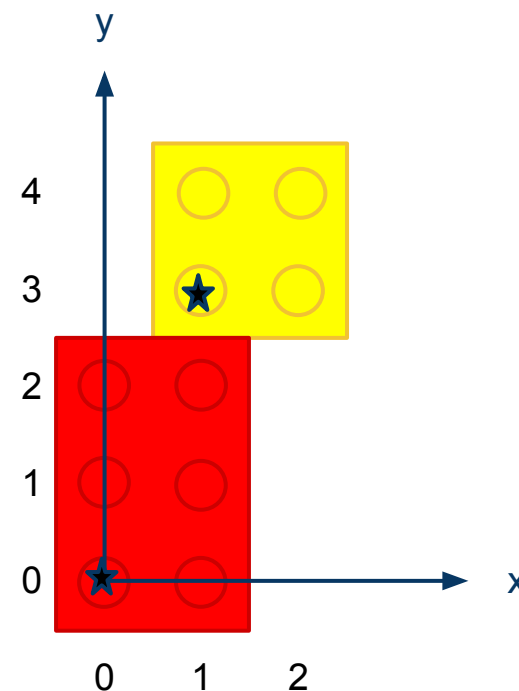
orientation	binary
horizontal	0
vertical	1

Using it, we can encode these bricks:

x- coordinate	y- coordinate	color	brick type	orientation
0	0	red	2X3	vertical
1	3	yellow	2X2	horizontal



x- coordinate	y- coordinate	color	brick type	orientation
00	00	0	0	1
01	11	1	1	0



Decode this one!



x- coordinate	y-coordinate	color	brick type	orientation
00	00	1	1	0
11	00	0	1	1
01	00	0	0	0
01	00	1	0	0

Legend:

color	binary
red	0
yellow	1

brick type	binary
2X3	0
2X2	1

orientation	binary
horizontal	0
vertical	1

Now, your own encodings!



Use the worksheet to create your own encoding in **ABSOLUTE SECRECY!**

Then, swap encodings with another group: they will try to decode yours and you will try to decode theirs!

The decoded Legos *should* look just like the originals -- *keep them hidden...*

Scratch with Images

Day 2, Session 4

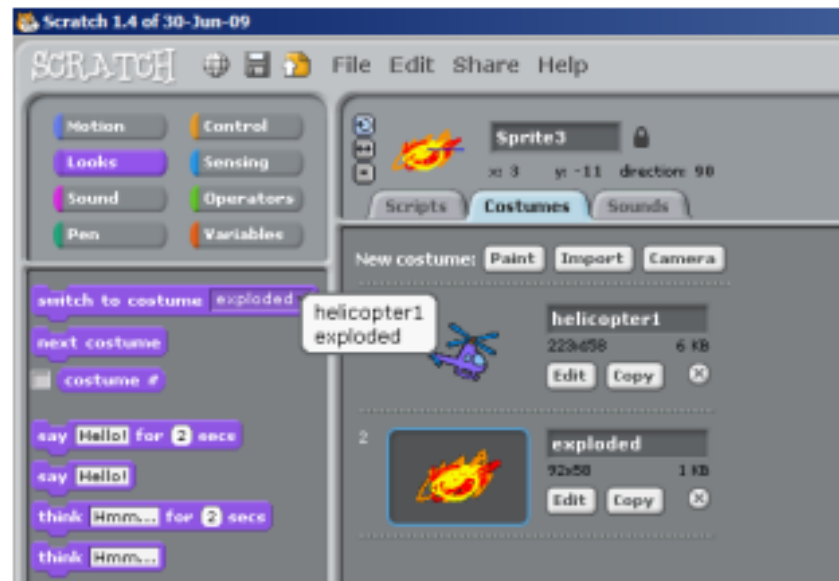
Costumes



A costume changes the way a sprite looks.

By using costumes, you can continue to control a sprite using the same scripts but you can also change its appearance.

Changing Costumes



Blocks to change a sprite's costume can be found in the **Looks** tab.

The **switch to costume** block will make the sprite change to a specific costume you choose.

The **next costume** block will make the sprite change to whatever costume is next in its list of costumes.

Simple Stories



Find a Scratch sprite with multiple costumes, and give it some dialogue that involves a costume change.

But I want to be creative!

Costumes can be **drawn** in Scratch, or **imported** in.



Last Story of the Day



Using costumes you draw yourself or import from outside Scratch, create a short story or extend your old one so that it includes several costume changes.

