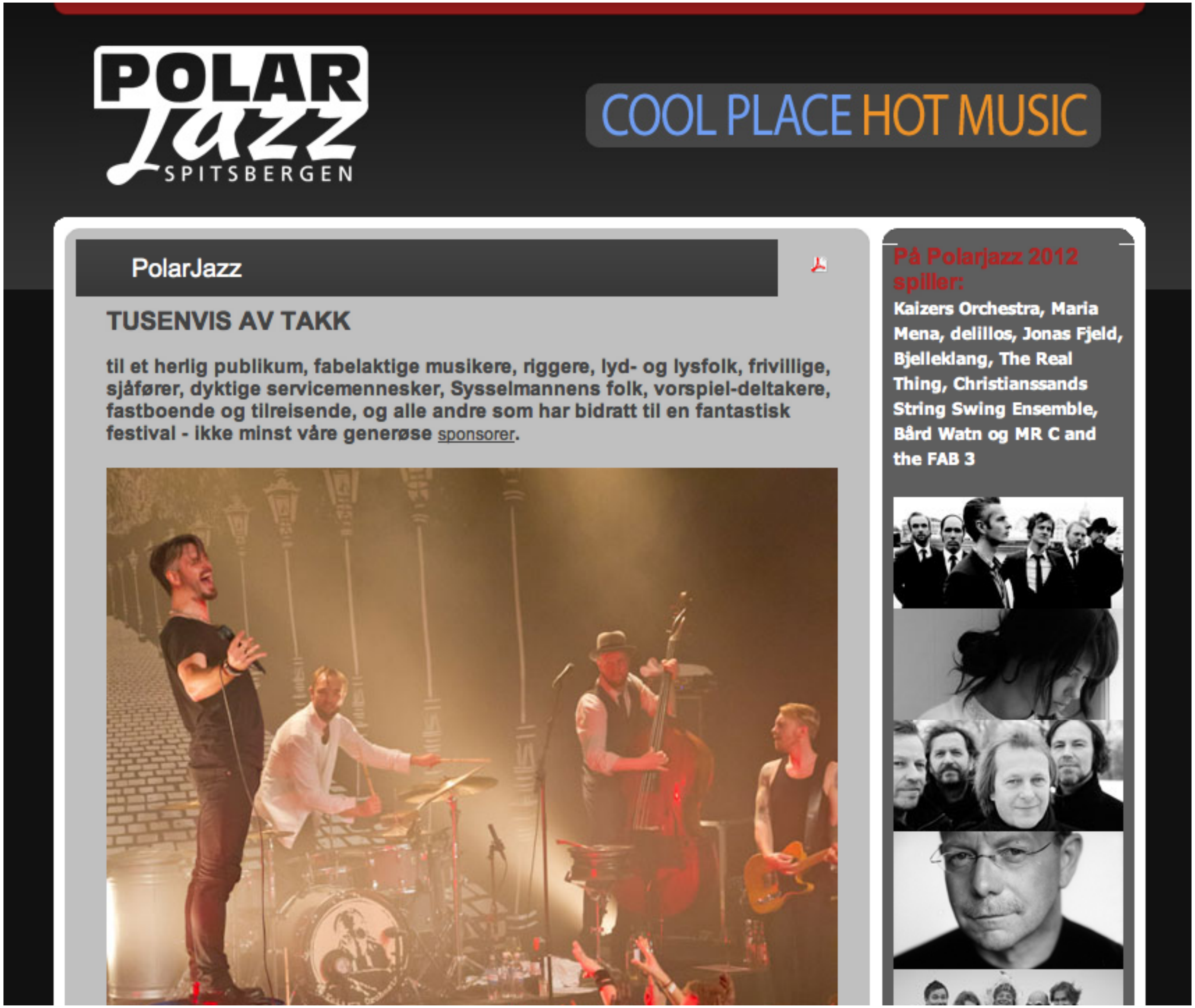
Good morning!

Sorting and Searching

Day 3, Session 1

too hot for me this summer...



PolarJazz

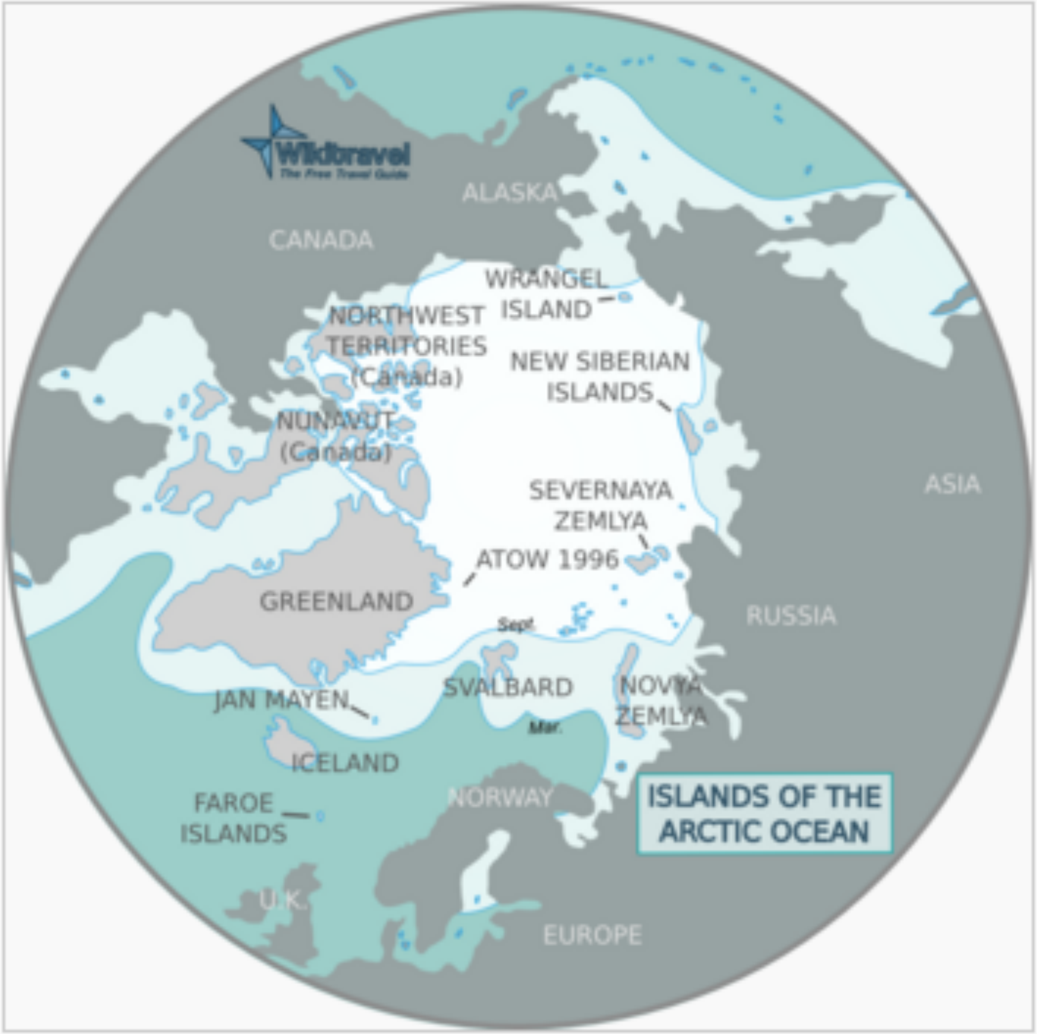
TUSENVIS AV TAKK

til et herlig publikum, fabelaktige musikere, riggere, lyd- og lysfolk, frivillige, sjåfører, dyktige servicemennesker, Sysselmannens folk, vorspiel-deltakere, fastboende og tilreisende, og alle andre som har bidratt til en fantastisk festival - ikke minst våre generøse sponsorer.

På Polarjazz 2012 spiller:

Kaizers Orchestra, Maria Mena, delillos, Jonas Fjeld, Bjelleklang, The Real Thing, Christianssands String Swing Ensemble, Bård Watn og MR C and the FAB 3

... in *Svalbard:*



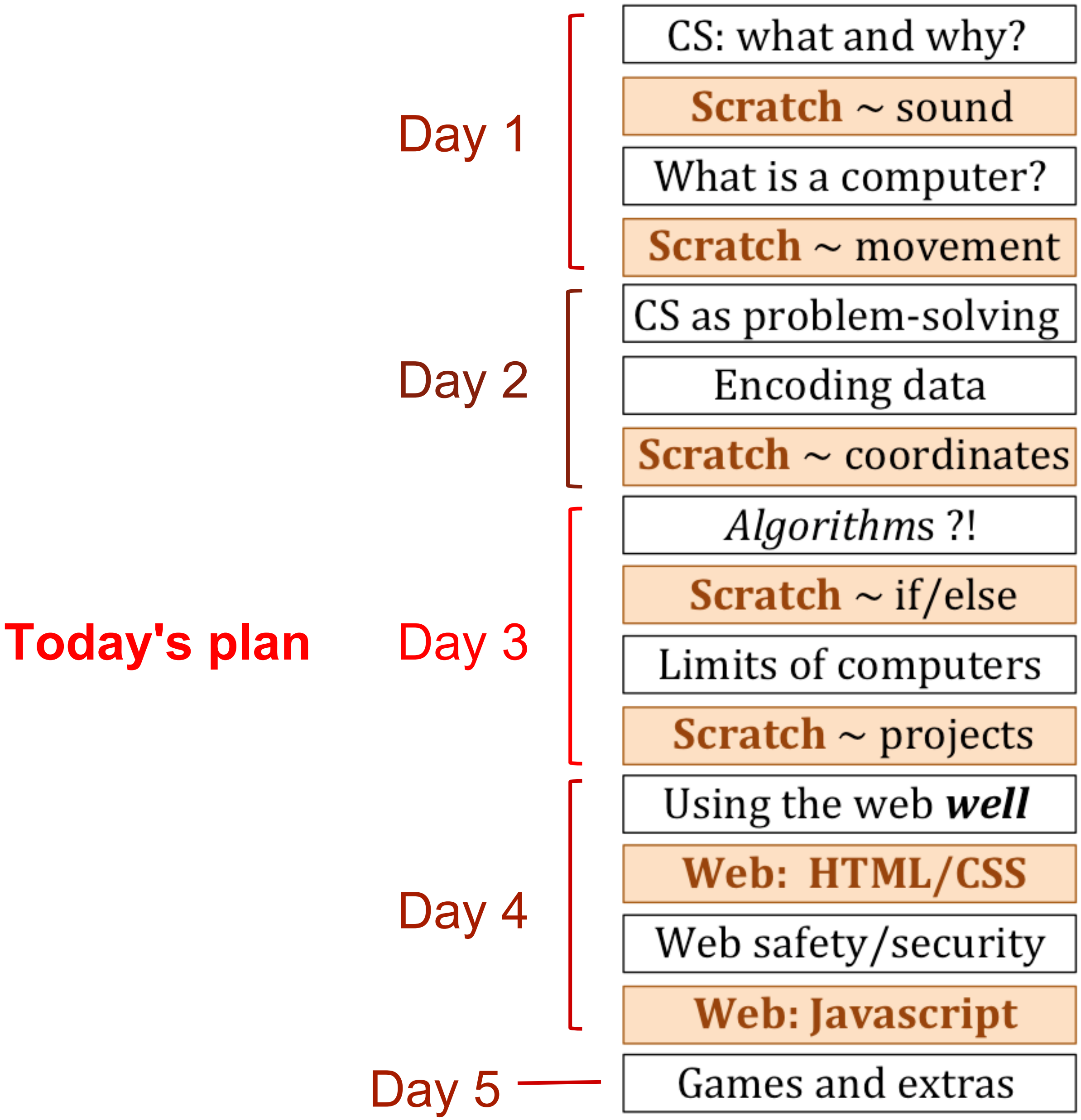
Svalbard in the Arctic Sea

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

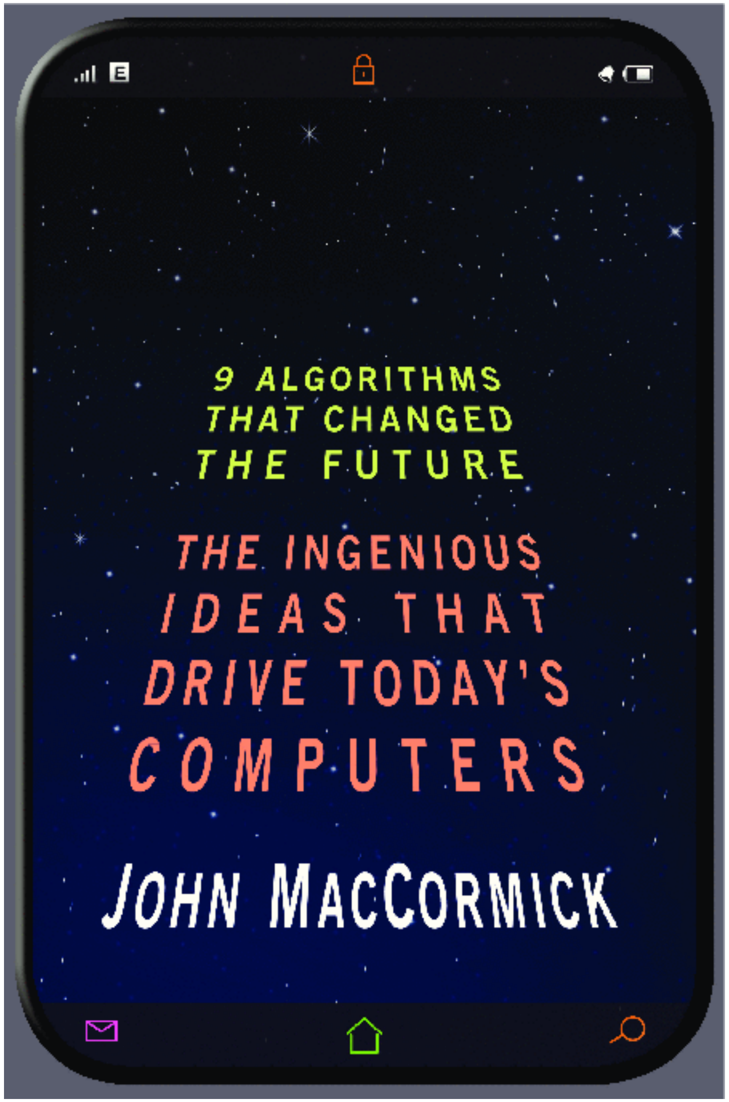
Day 2
- CS as problem-solving
- Encoding data
- **Scratch** ~ coordinates

Day 3
- *Algorithms* ?!
- **Scratch** ~ if/else
- Limits of computers
- **Scratch** ~ projects

Day 4
- Using the web ***well***
- **Web: HTML/CSS**
- Web safety/security
- **Web: Javascript**

Day 5
- Games and extras

"Algorithm"

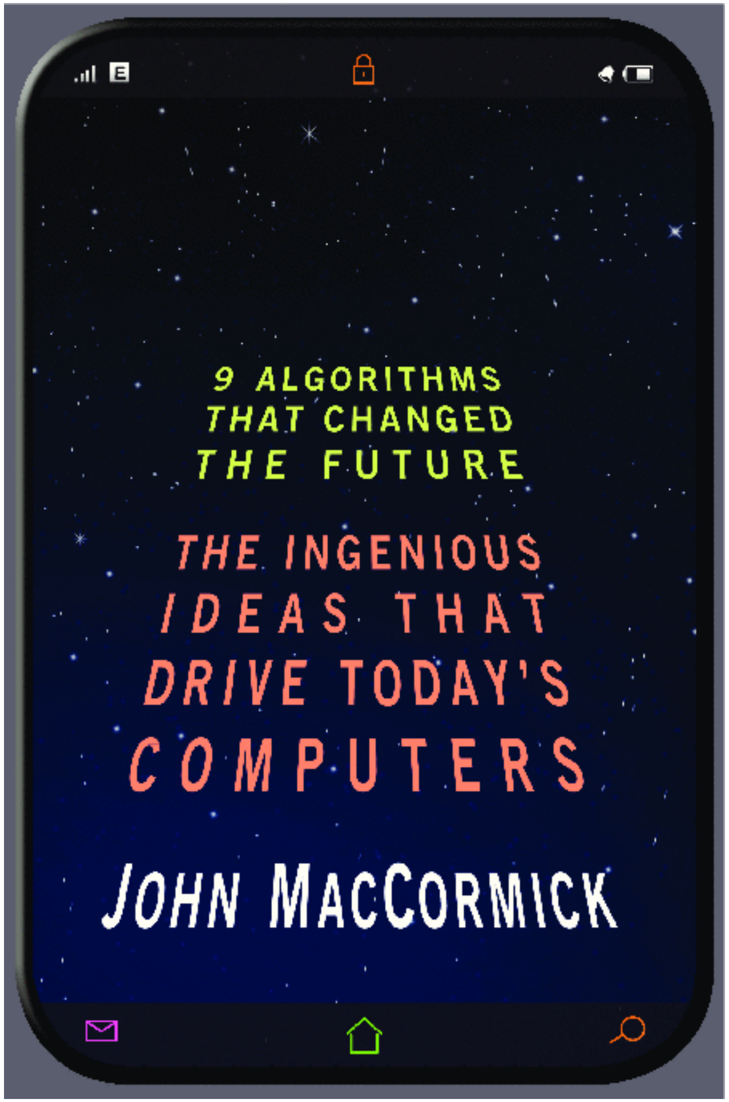

9 ALGORITHMS THAT CHANGED THE FUTURE
THE INGENIOUS IDEAS THAT DRIVE TODAY'S COMPUTERS
JOHN MACCORMICK

An algorithm is a ***process*** to solve a problem.

Simply put, algorithms are what CS is about.

"Algorithm"

An algorithm is a ***process*** to solve a problem.


9 ALGORITHMS THAT CHANGED THE FUTURE
THE INGENIOUS IDEAS THAT DRIVE TODAY'S COMPUTERS
JOHN MACCORMICK

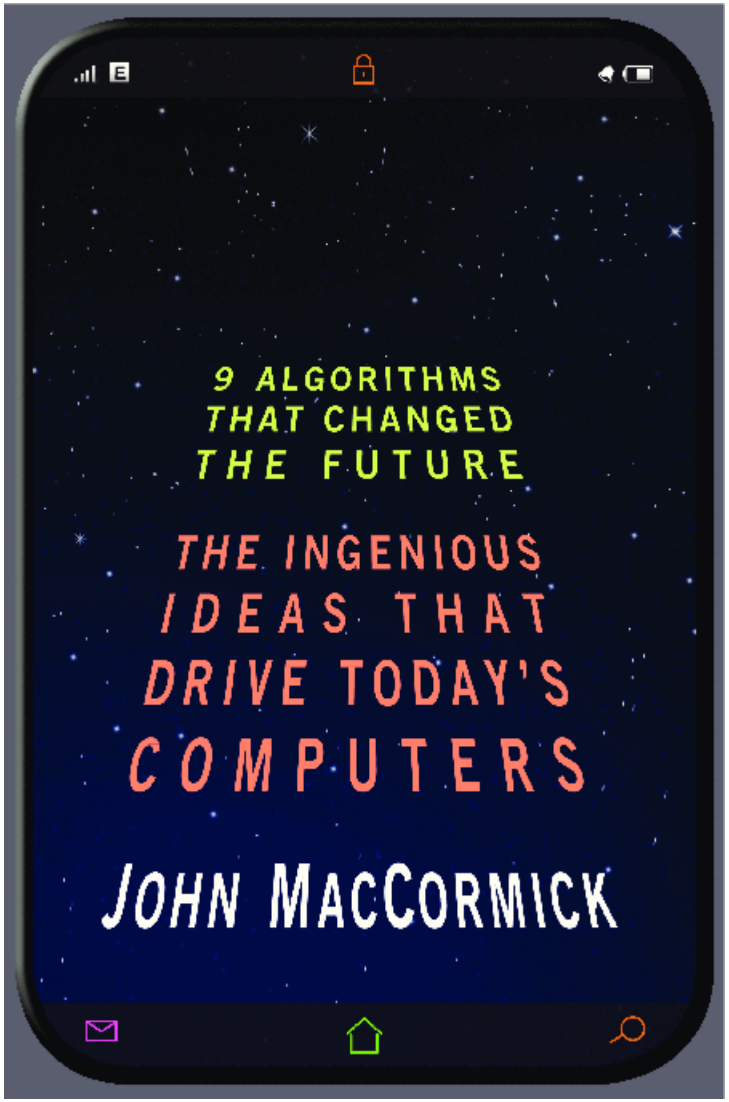
The Nine Algorithms/Chapters:

(1) Ideas Computers Use Every Day
(2) Search Engine Indexing: Finding Needles in the World's Biggest Haystack
(3) PageRank: The algorithm that made Google
(4) Public Key Cryptography: Secrets on a Postcard
(5) Error-Correcting Codes: Mistakes That Fix Themselves
(6) Pattern Recognition: Learning from Experience
(7) Data Compression: Something for Nothing
(8) Databases: The Quest for Consistency
(9) Digital Signatures: Who *Really* Wrote This Software?

(10) What Is Computable?
(11) Conclusion: More Genius at Your Fingertips?

"Algorithm"

An algorithm is a ***process***
to solve a problem.



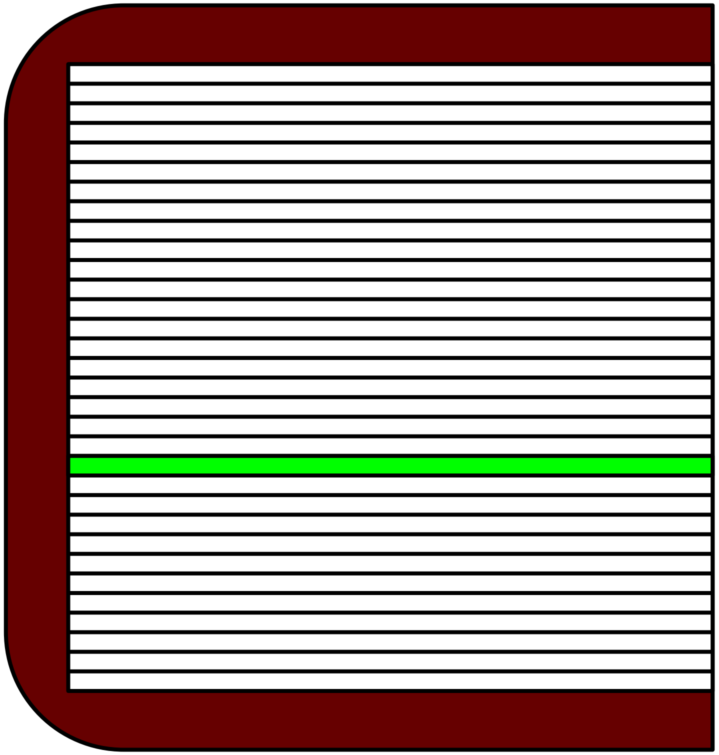
The Nine Algorithms/Chapters:

(1) Ideas Computers Use Every Day
(2) Search Engine Indexing: Finding Needles in the World's Biggest Haystack
(3) PageRank: The algorithm that made Google
(4) Public Key Cryptography: Secrets on a Postcard
(5) Error-Correcting Codes: Mistakes That Fix Themselves
(6) Pattern Recognition: Learning from Experience
(7) Data Compression: Something for Nothing
(8) Databases: The Quest for Consistency
(9) Digital Signatures: Who *Really* Wrote This Software?

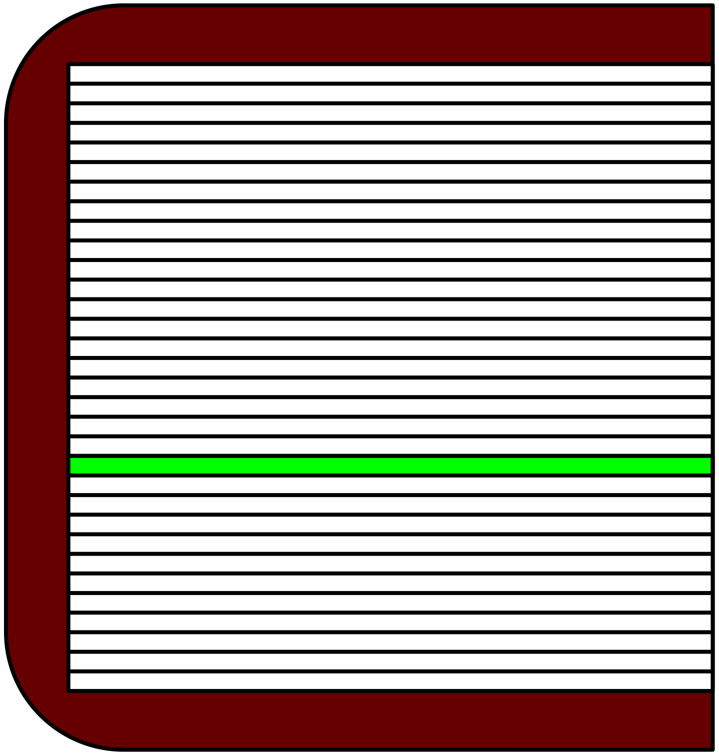
(10) What Is Computable?
(11) Conclusion: More Genius at Your Fingertips?

Searching!

"spam"



Wubster's -
not in order!

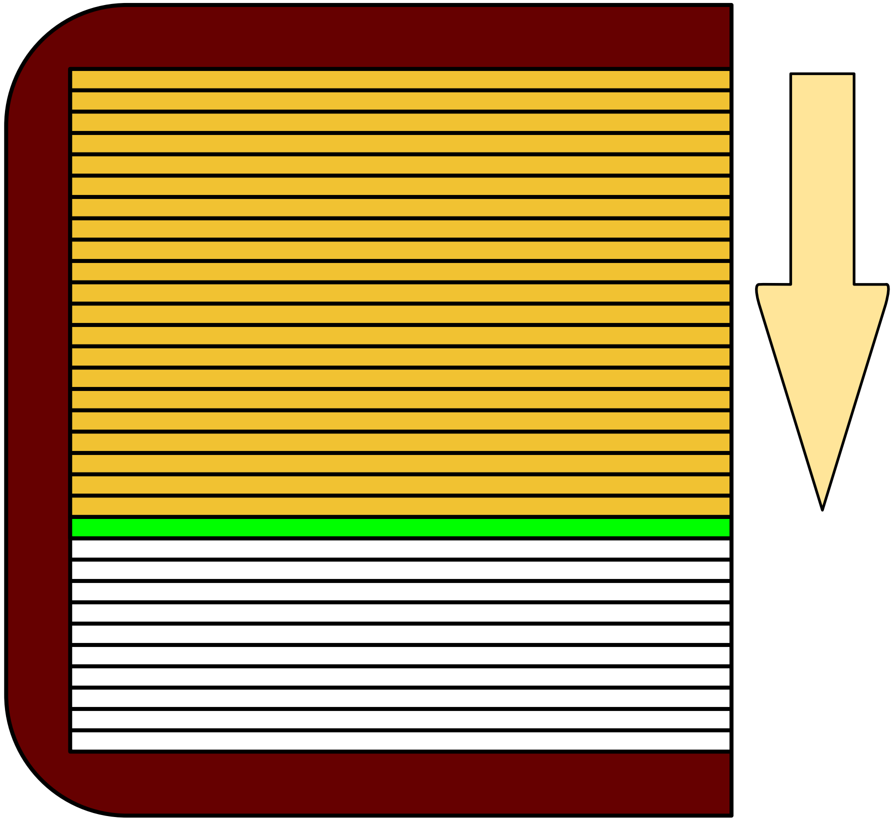


Webster's -
in order

What ***algorithm*** do we use to find "spam"?

Linear Search (unsorted book)

- Start at the first word.
- "Is this the word we want?"
- If it's not, go to the next word.



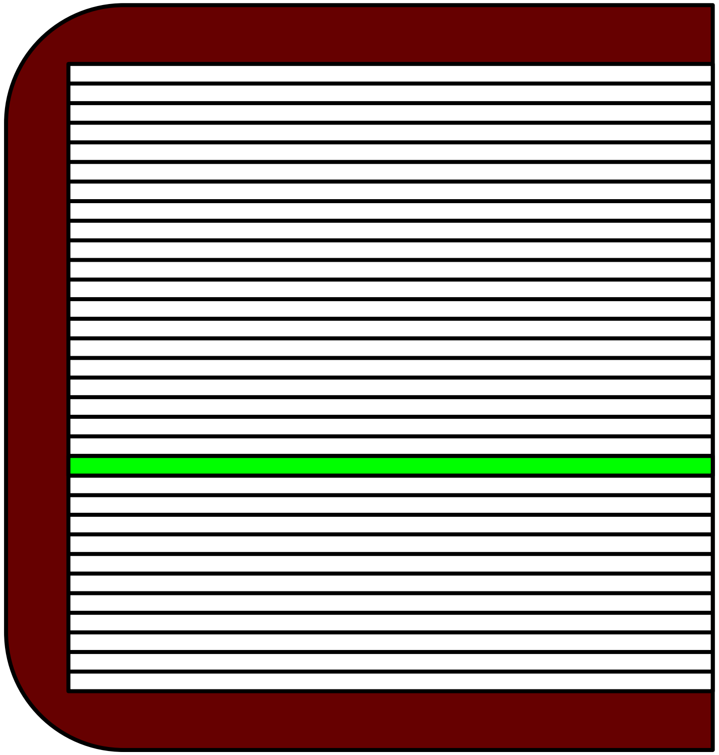
How many words do we have to look at before we get to the right one?

... if the words are not in order

Binary Search (*sorted* dictionary)

Can we find words faster if they're sorted?

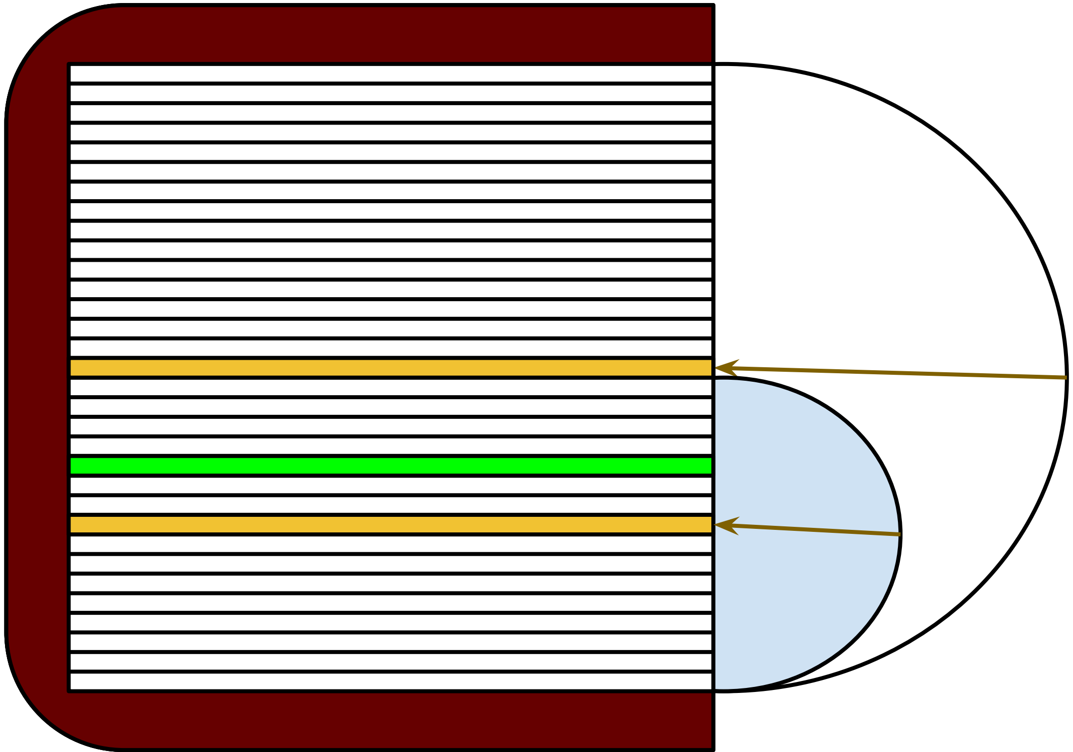
If so, how?!?



Webster's
Dictionary

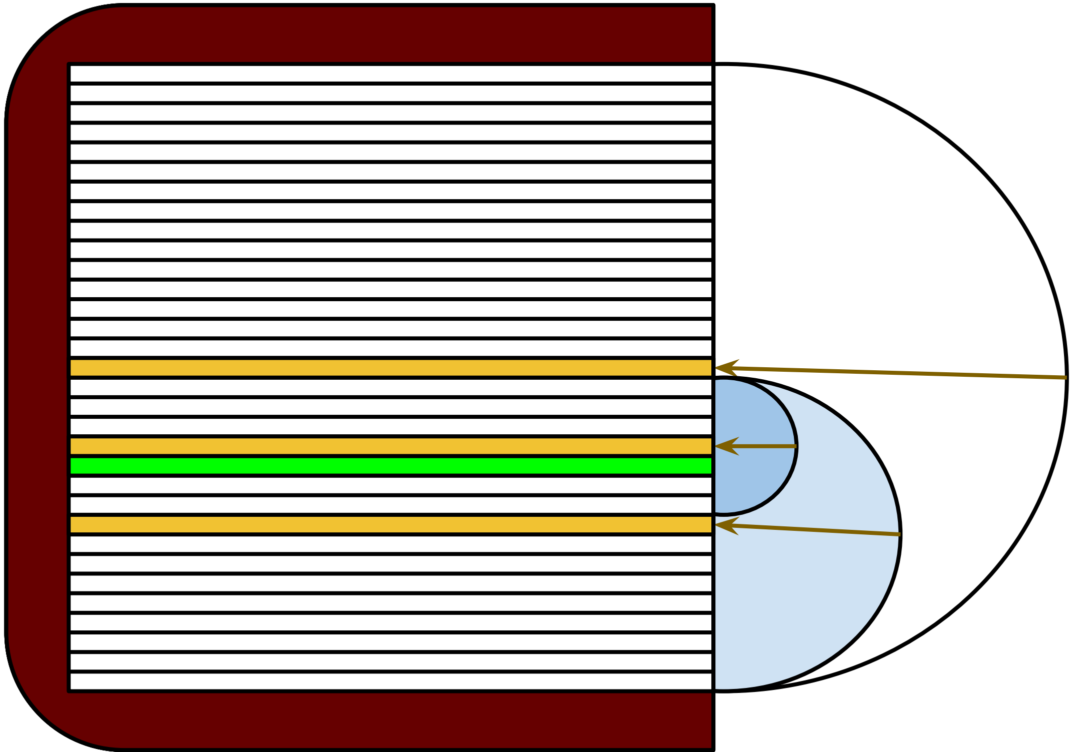
Binary Search (*sorted* dictionary)

- Start in the middle.
- "Is this the word we want?"
    - If it's before the word we want, go to the middle of the second half.
    - If it's after the word we want, go to the middle of the first half.



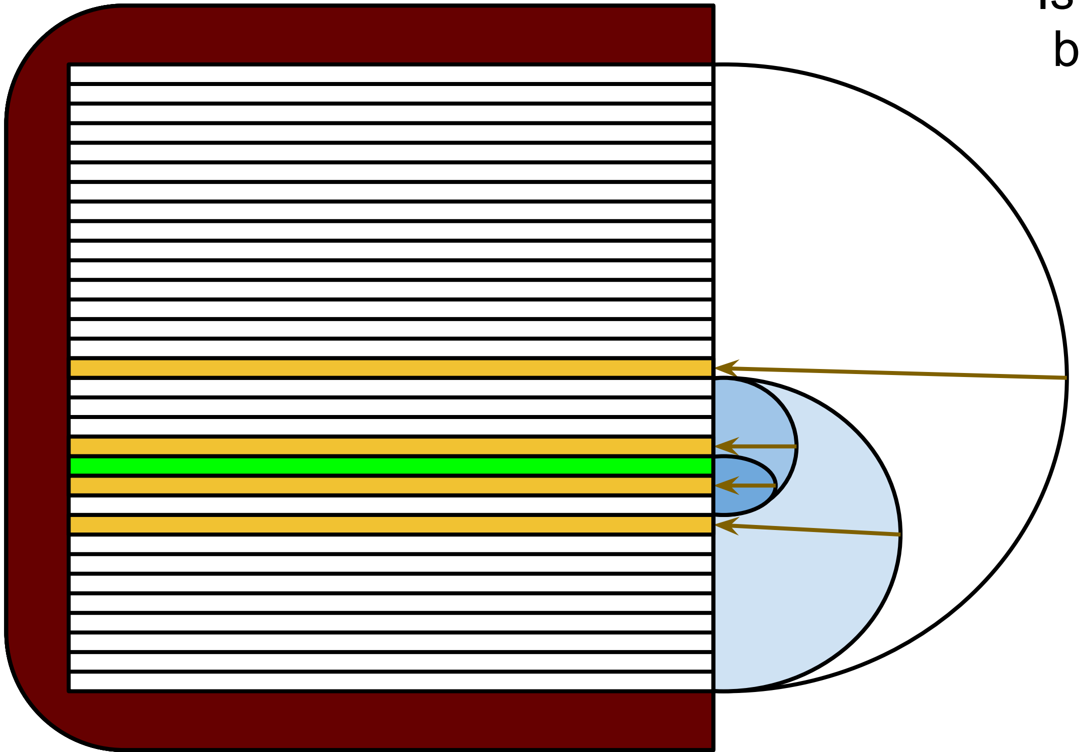
Binary Search (*sorted* dictionary)

- Start in the middle.
- "Is this the word we want?"
  - If it's before the word we want, go to the middle of the second half.
  - If it's after the word we want, go to the middle of the first half.



Binary Search (*sorted* dictionary)

- Start in the middle.
- "Is this the word we want?"
  - If it's before the word we want, go to the middle of the second half.
  - If it's after the word we want, go to the middle of the first half.



Is this **binary searching** going to be better than linear searching?

If our book has 100,000 words in it, how many do we have to look at before we find the right one?

- in the BEST case?
- in the WORST case?
- on average?

Linear vs. Binary Search

With a large amount of ***sorted*** data, binary search is almost always more efficient.

But ***how*** do we get our data sorted?

Let's sort!

ACTIVITY

Work with your partner to sort your set of weights -- they are all different!

Challenge: come up with a sorting ***algorithm*** that can be easily described and copied.

Notes: Compare only two weights at a time! Computers can compare only two at a time, and they need a concrete set of conditions that lets them know that a list is sorted.

Extra: Count the number of comparisons you make between any two weights.

Comparisons

The basic building-block of sorting is *comparing two items*.

Consider these two numbers:

110111001100100100010100111110101

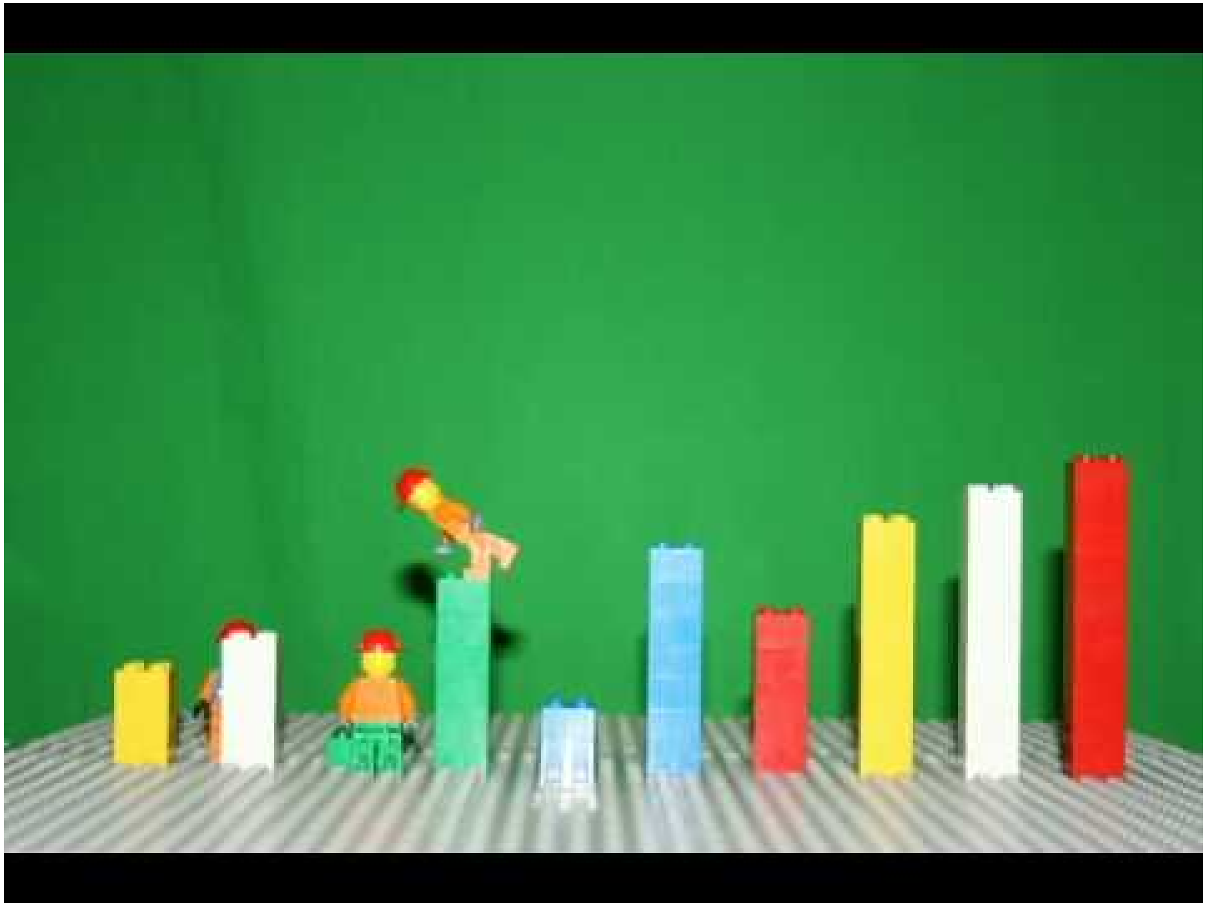
and

110111001100100100010101111110101

Which number is larger?

Our sorting algorithms:

Lego Bubble Sort



http://www.youtube.com/watch?v=MtcrEhrt_K0

Watch this sorting algorithm... can you describe how it works?

Linear vs. Binary Sorting!

How do they compare?

1) Min Sort

2) Merge Sort

How does this relate to dictionary-searching?

Min Sort

DEMO

Linear Sorting

Search for the first element, then the second, and so on...

Compare elements pairwise to find the minimum element. Then, remove it from the unsorted list and place it at the end of the sorted list. Repeat until all elements are in the sorted list.

Merge Sort

Binary Sorting

DEMO

We can merge two sorted lists by comparing the two first elements and placing the lesser element at the end of the sorted list until all elements are in the sorted list.

So, we can split our unsorted list into several one-element "sorted" lists, merge them into sorted two-element lists, and keep merging until we have one big sorted list.

Sorting out sorting

ACTIVITY

Let's race *the algorithms*

Divide into groups of 8...

1) Act out minsort - count your comparisons: ______

2) Act out mergesort - count comparisons: ______

3) Try the parallel sorting network - count: ______

Which one wins?

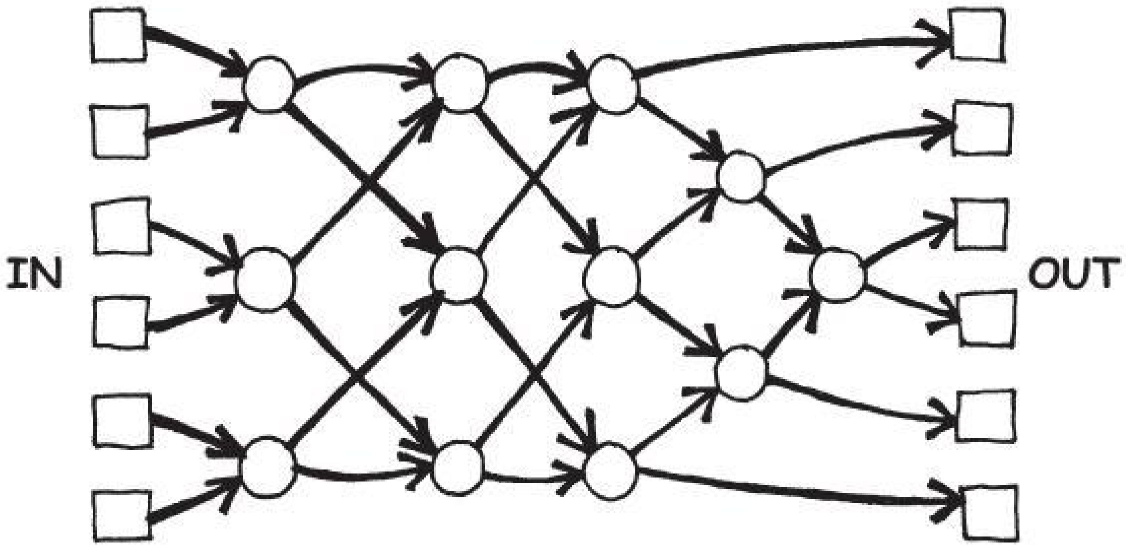
Parallel Sorting

ACTIVITY

Follow the arrows:

higher # goes left

lower # goes right



Who can sort themselves faster using a parallel sorting network? Teachers or students?

Is MergeSort for real? *Let's see...*

10 piles of 16

5 piles of 32

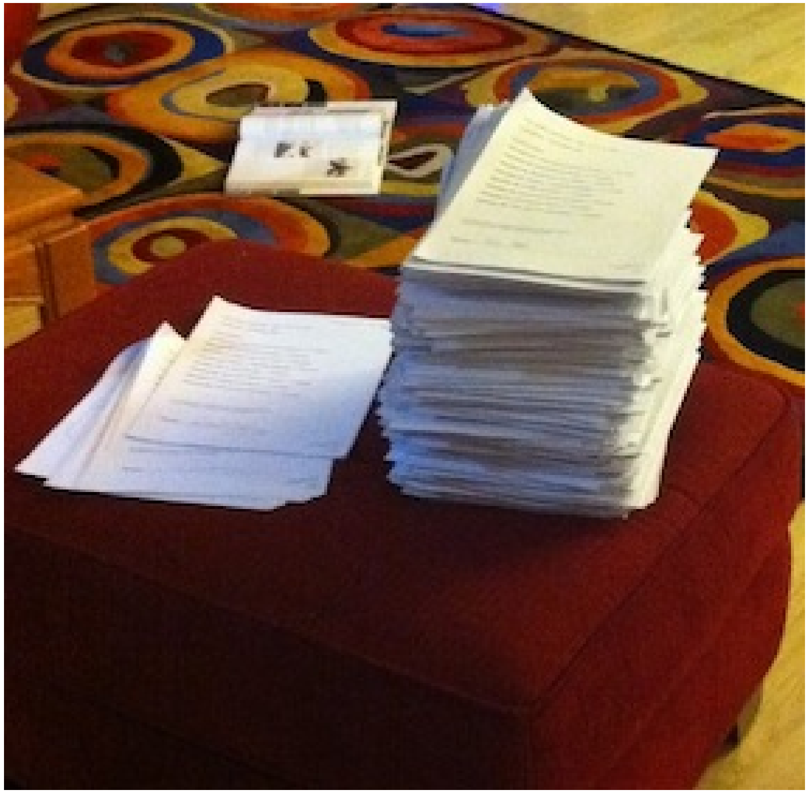
Is MergeSort for real? *Let's see...*

5 piles of 32

2 piles of 80

Is MergeSort for real? *Let's see...*

2 piles of 80

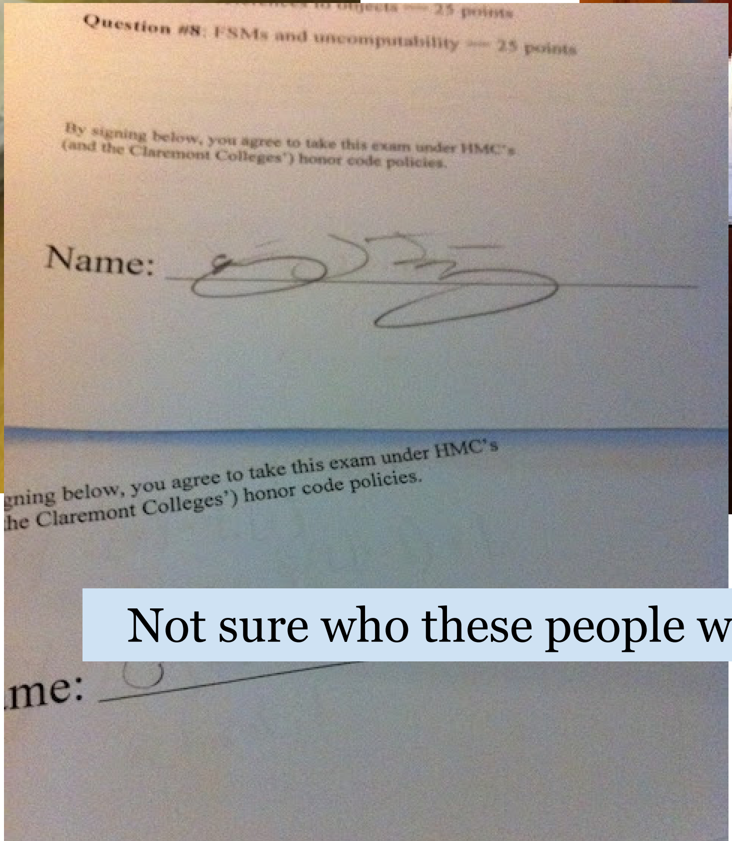


1 pile of 158

158?

Is MergeSort for real? *Yes!*

2 piles



...ences to objects — 25 points

Question #8: FSMs and uncomputability — 25 points

By signing below, you agree to take this exam under HMC's (and the Claremont Colleges') honor code policies.

Name:

gning below, you agree to take this exam under HMC's the Claremont Colleges') honor code policies.

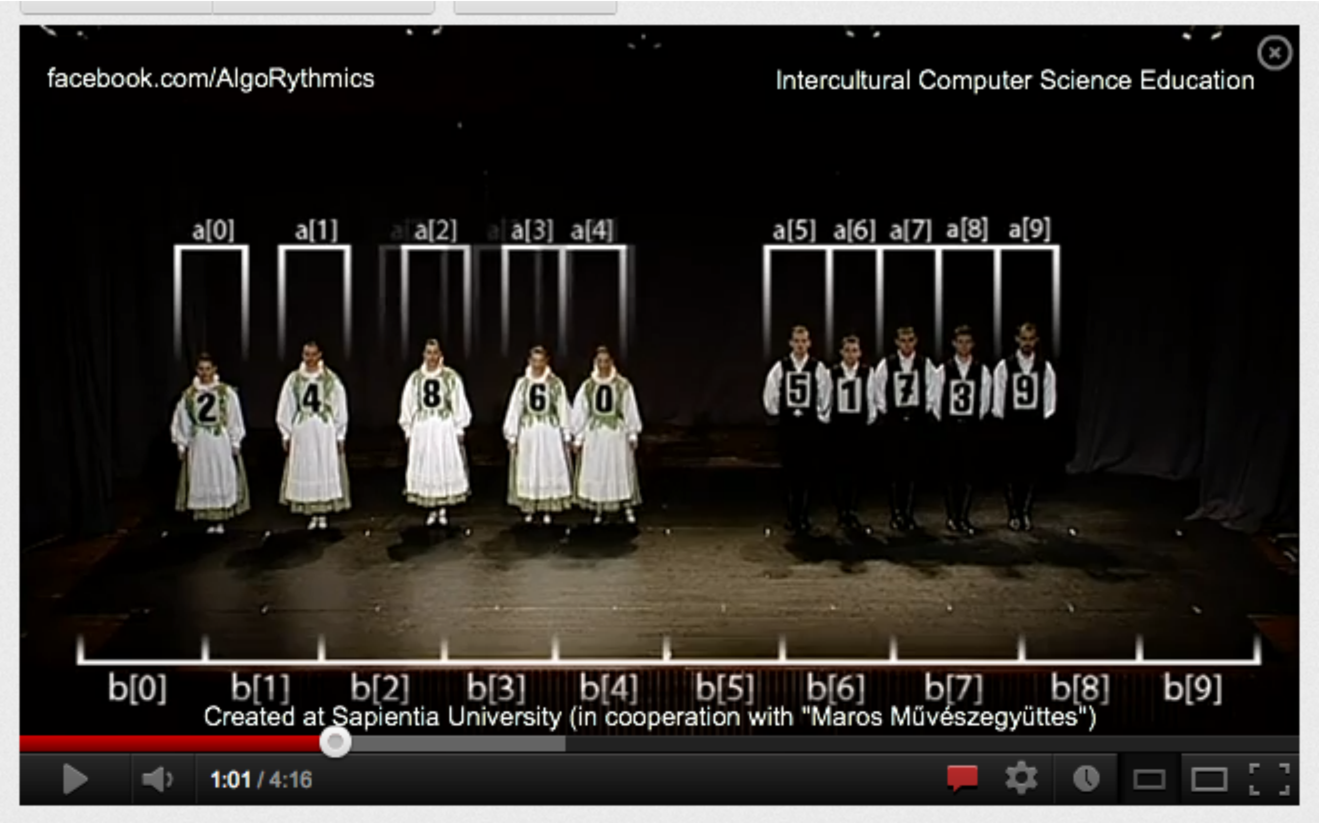
me:

Not sure who these people were....

1 pile of 158

158?

Mergesort, in dance

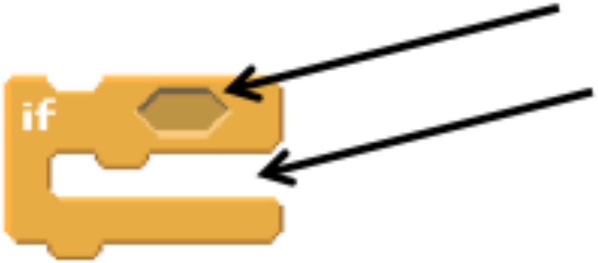

facebook.com/AlgoRythmics
Intercultural Computer Science Education
a[0] a[1] a[2] a[3] a[4] a[5] a[6] a[7] a[8] a[9]
2 4 8 6 0 5 1 7 3 9
b[0] b[1] b[2] b[3] b[4] b[5] b[6] b[7] b[8] b[9]
Created at Sapientia University (in cooperation with "Maros Művészegyüttes")
1:01 / 4:16

look this up to see the many options...

Scratch: Ifs and Elses

Day 3, Session 2

If Block

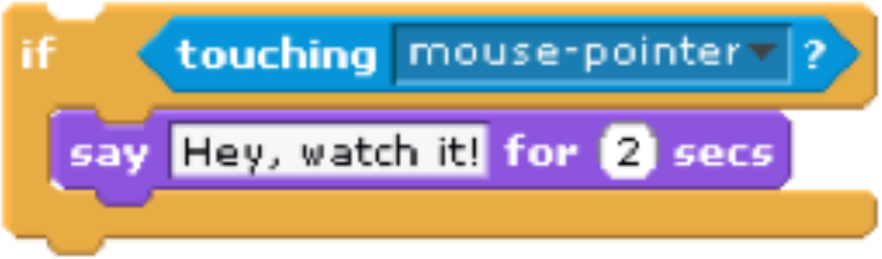


The "if" block can be found in the **control** tab.

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

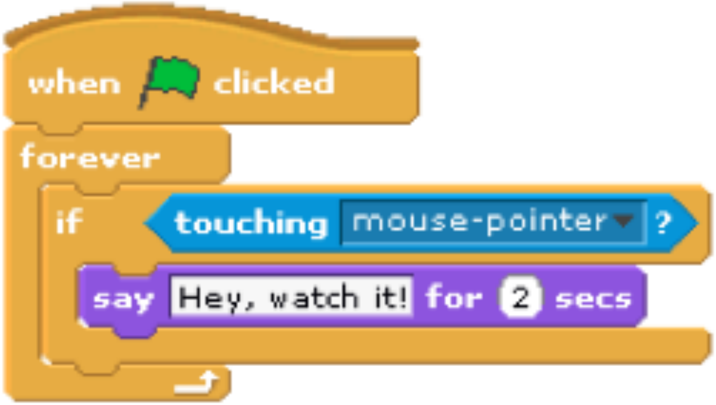


```
if touching mouse-pointer ?
  say Hey, watch it! for 2 secs
```

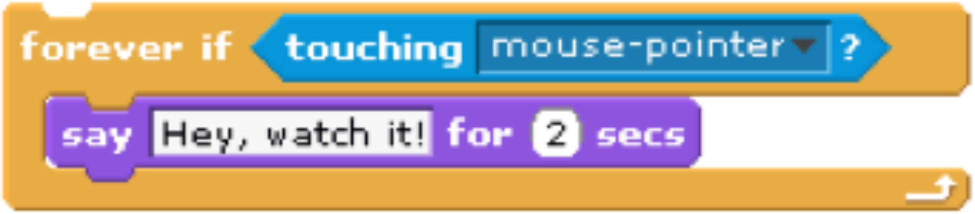
Most of the blocks that fit ON the if block are **sensing** blocks, which can be found in the sensing tab.

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**.

Forever If



```
when [flag] clicked
forever
    if <touching [mouse-pointer] ?>
        say [Hey, watch it!] for (2) secs
```



```
forever if <touching [mouse-pointer] ?>
    say [Hey, watch it!] for (2) secs
```

Usually, you don't want to just run the if block **once**, you want it to **always** be running.

To make it run forever you can either put it in a **forever block** or use the **forever if** block.

Helicopter Game

ACTIVITY

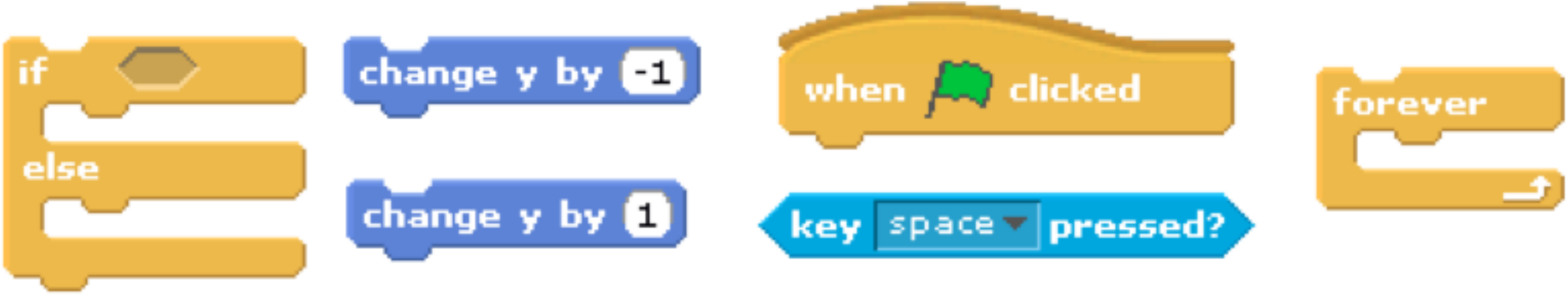
Open Scratch and add the **helicopter** sprite.

It can be found in the "transportation" folder of sprites.

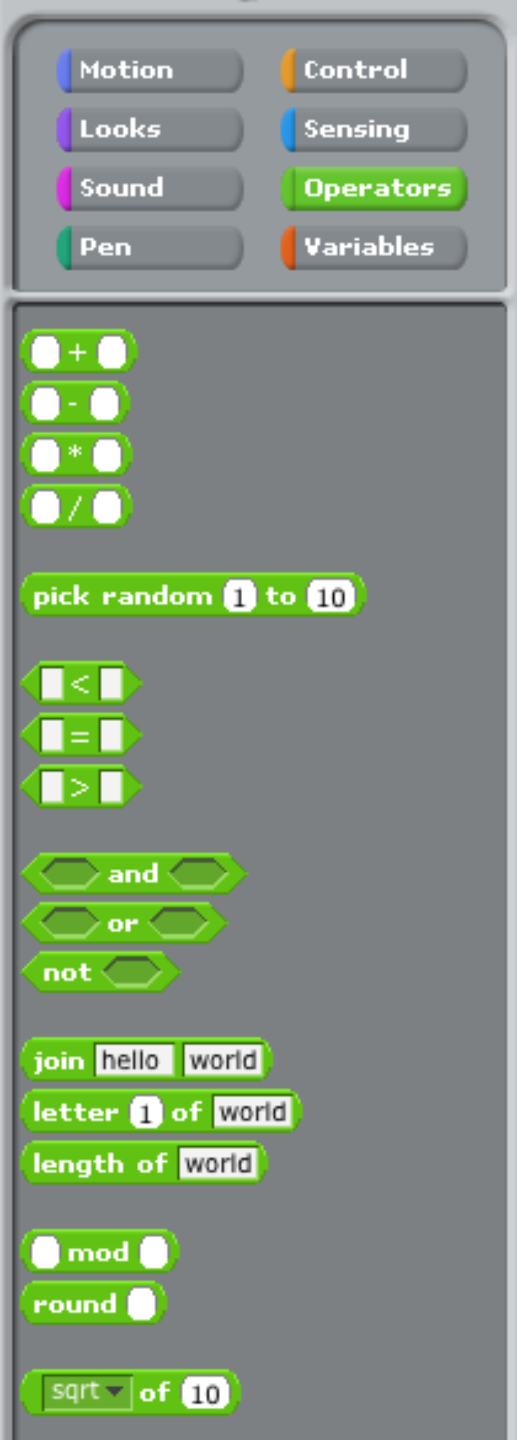
Helicopter Game

ACTIVITY

Use these blocks to write a script that makes it so that if you press the UP arrow, the helicopter goes up, but if you DON'T press the up arrow, the helicopter goes down (like gravity!)



Make sure to save when done!



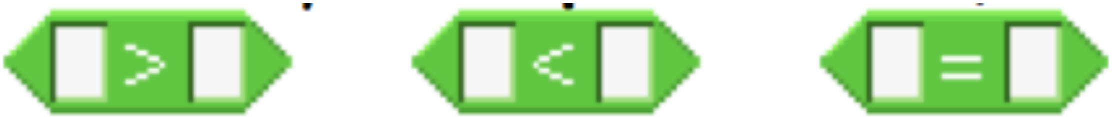
Operators

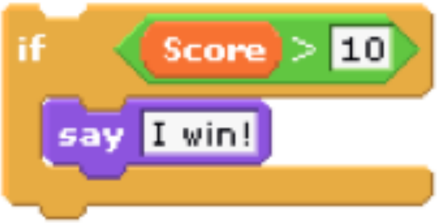
In Scratch, **operators** let you **connect** other ideas together.

Many of these operators do things you have seen in **math**.

Common Operators

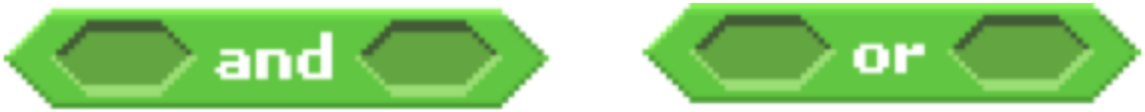
Some operators let you **compare** values (like numbers).

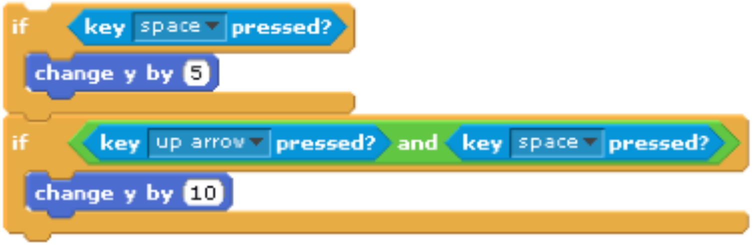




```
if Score > 10
    say I win!
```

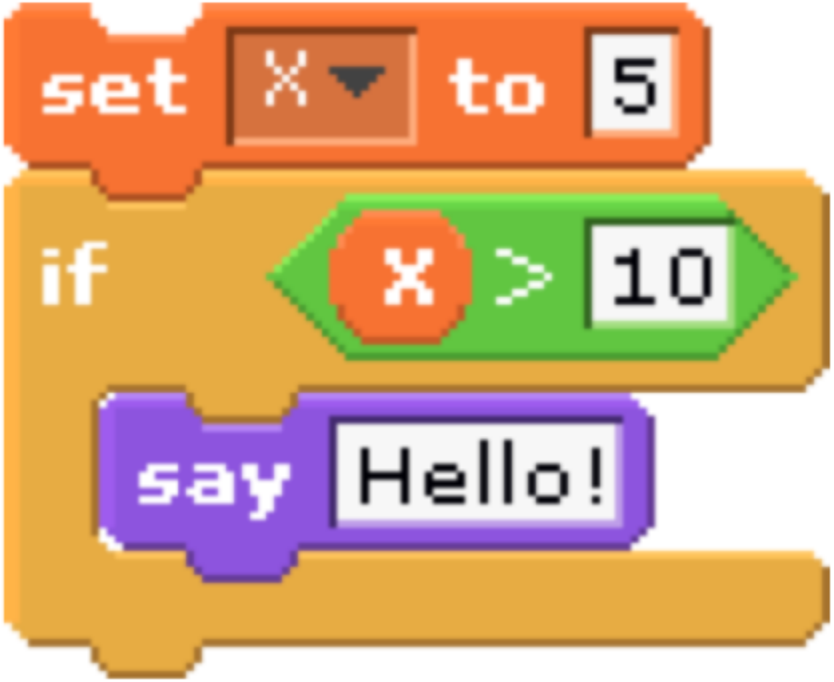
Other operators let you **combine** other blocks.





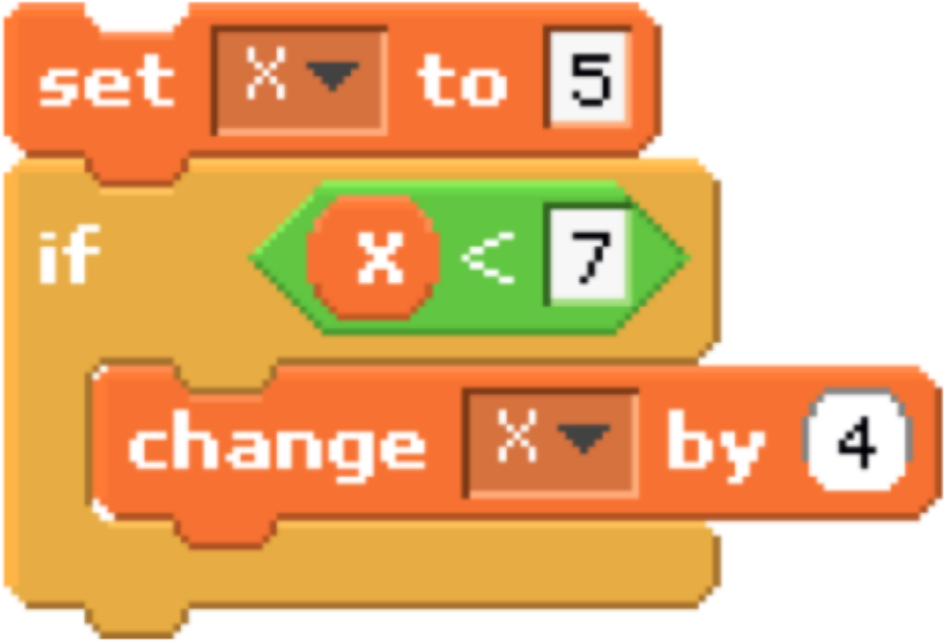
```
if key space pressed?
    change y by 5
if key up arrow pressed? and key space pressed?
    change y by 10
```

Will the sprite say "Hello!" when you run this script?



```
set X to 5
if X > 10
    say Hello!
```

What will the variable "X" be after you run this script?



```
set X to 5
if X < 7
    change X by 4
```

Predict and Test

ACTIVITY

To get comfortable with operators, partner with the person sitting next to you and go through the worksheet.

Storytelling

ACTIVITY

Create a story with two characters. Have their interaction include movement, costume changes, as well as say.

Scratch stories

ACTIVITY

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.

or, you can create a game...

Helicopter Game Again...

ACTIVITY

Open up your helicopter game.

Change the background to have a start, an end, and some obstacles (color is key!).

Then, make sure that if the helicopter hits an obstacle, it resets back to the start.

You now have a playable game!

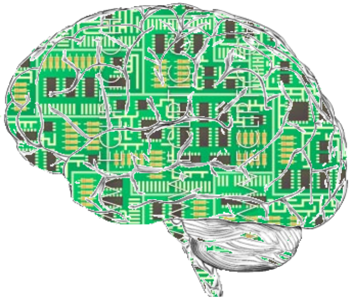
Limitations of Computer Science

Day 3, Session 3

Can computers be intelligent?

IBM's Watson:

Chatbots such as Eliza and Jabberwacky



Turing Test

ACTIVITY

How well a computer can mimic human intelligence?
Alan Turing wanted to know!

The Turing Test determines whether a computer's answers to questions are distinguishable from human answers.

You must separate answers from a human from answers from a computer.

Decide with your partner which question you would ask to decide whether each of three "agents" is human or computer...

Some possible questions...

1. What is the name of Bart Simpson's baby sister?
2. What do you think of Roald Dahl?
3. Are you a computer?
4. What is the next number in the sequence 3, 6, 9, 12, 15?
5. What do you think of nuclear weapons?
6. What is 2 X 78?
7. What is the square root of 2?
8. Add 34957 to 70764.
9. Do you like school?
10. Do you like dancing?
11. What day is it today?
12. What time is it?
13. How many days are there in February in a leap year?
14. How many days are there in a week?
15. For which country is the flag a red circle on a white background?
16. Do you like to read books?
17. What food do you like to eat?

How would you use this in class?

- Perhaps try...   Twenty Questions

Simulating humans is difficult!

So, let's try ***simpler*** life ~ *cells*

Real cells

White Blood Cell Chases Bacteria

http://www.youtube.com/watch?v=BNizygcJA9A

Can ***simpler*** cells still be "life"-like?

Simple Cells

	1	2	3	4	5	6
A						▨
B		B2	▨			
C		▨	▨ C3			
D		▨	▨			
E				▨	▨	
F						

▨ = Living cell

□ = Empty space

A grid of cells depending on

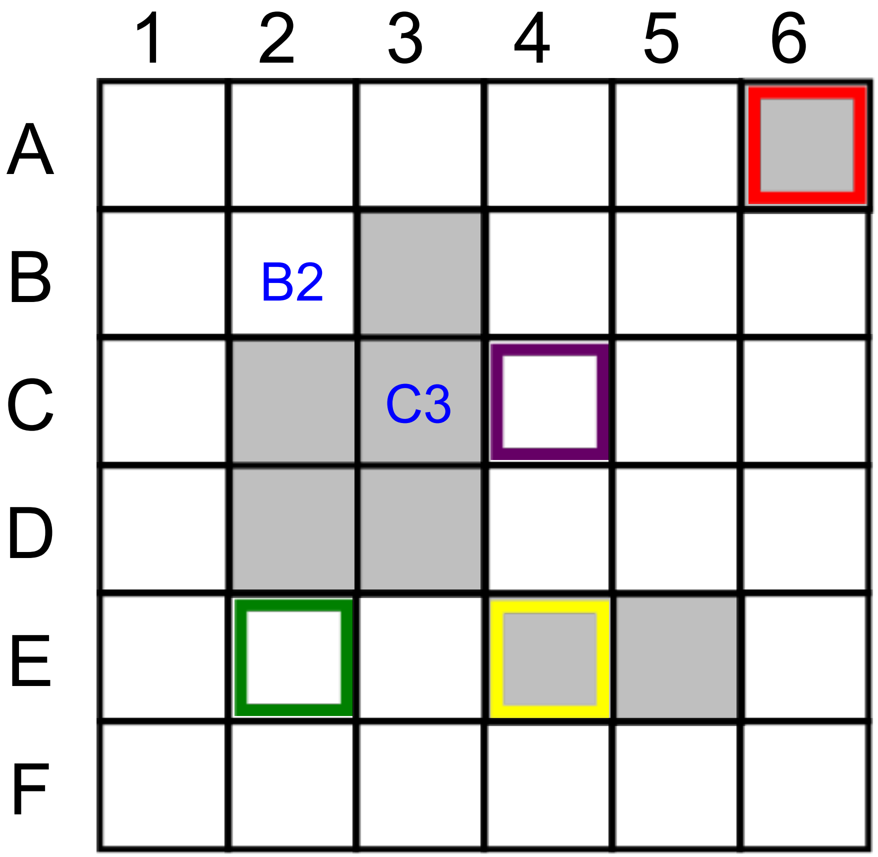
(1) their rules (DNA)

(2) environment (neighbors)

How many live cells are in this grid?

Simple Cells

ACTIVITY



	1	2	3	4	5	6
A	□	□	□	□	□	▨
B	□	B2	▨	□	□	□
C	□	▨	▨ C3	□	□	□
D	□	▨	▨	□	□	□
E	□	□	□	▨	▨	□
F	□	□	□	□	□	□

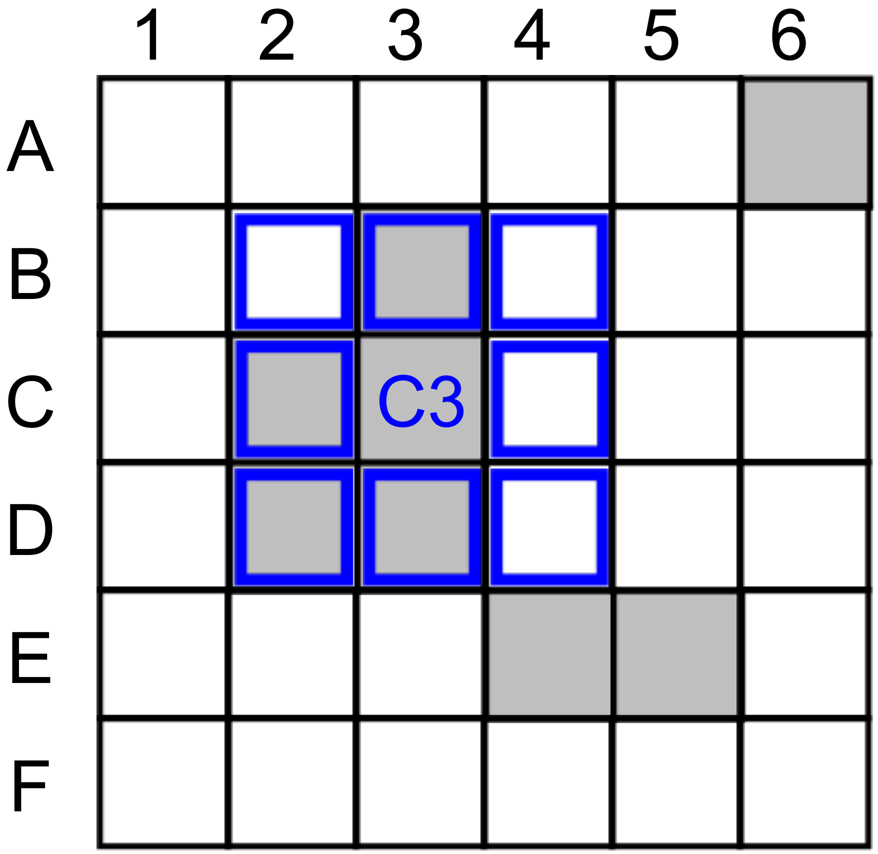
▨ = Living cell

□ = Empty space

	Name	Is it live or empty?
▨		
□		
▨		
□		

Neighbor cells

ACTIVITY



▨ = a living cell

□ = empty space

	Living neighbors	**Empty** neighbors
C3		

Cell C3 has **8** neighbors

How many of each?

Neighbor cells

ACTIVITY

	1	2	3	4	5	6
A						▨ A6
B		B2	▨			
C		▨	▨ C3	C4		
D		▨	▨			
E		E2		▨ E4	▨	
F						

▨ = Living cell

□ = Empty space

	# of living neighbors
▨ A6	
C4	
▨ E4	
E2	

Each cell's future depends on its living neighbors

Two rules

	1	2	3	4	5	6
A						0n
B			2n			
C		4n	4n			
D		3n	4n			
E				2n	1n	
F						

BEFORE

A living cell with ***2 or 3*** living neighbors ***survives***.
Others die.

An empty cell with exactly 3 living neighbors ***comes to life***. No others do.

Two rules

	1	2	3	4	5	6
A						
B		▨	▨			
C				▨		
D		▨				
E			▨	▨		
F						

AFTER

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***. No others do.

Two rules

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***.

	1	2	3	4	5	6
A						▨
B			▨			
C		▨	▨			
D		▨	▨			
E				▨	▨	
F						

BEFORE

	1	2	3	4	5	6
A						
B		▨	▨			
C				▨		
D		▨				
E			▨	▨		
F						

AFTER

Two rules

ACTIVITY

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***.

	1	2	3	4	5	6
A						
B		■	■			
C				■		
D		■				
E			■	■		
F						

BEFORE

	1	2	3	4	5	6
A						
B						
C						
D						
E						
F						

Fill in the next generation here.

Two rules

ACTIVITY

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***.

Only TWO survive, and TWO are born.

A						
B		▨	▨			
C				▨		
D		▨				
E			▨	▨		
F						

BEFORE

	1	2	3	4	5	6
A						
B						
C						
D						
E						
F						

Fill in the next generation here.

Two rules

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***.

	1	2	3	4	5	6
A						
B		▨	▨			
C				▨		
D		▨				
E			▨	▨		
F						

BEFORE

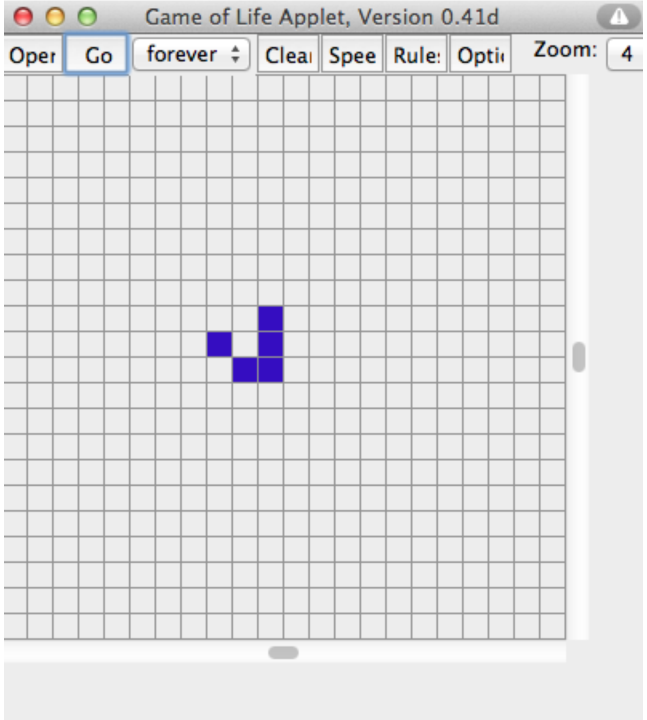
	1	2	3	4	5	6
A						
B			▨			
C		▨				
D				▨		
E			▨			
F						

AFTER

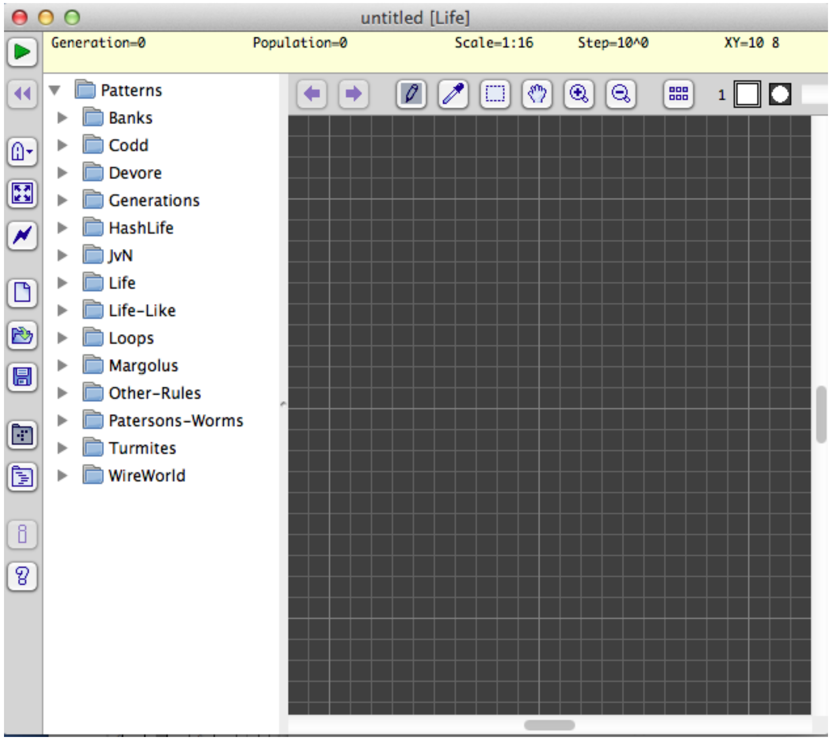
next...?

"The Game of Life"

John Conway, early '70s



http://www.ibiblio.org/lifepatterns/



Generation=0 Population=0 Scale=1:16 Step=10^0 XY=10 8

Golly (downloadable)

□

empty
space

Simple data

▨

living
cells

+

Basic rules

A living cell with ***2 or 3*** living neighbors ***survives***. Others die.

An empty cell with exactly 3 living neighbors ***comes to life***.

=

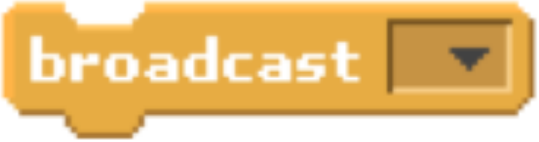
Complex behavior

"Game of Life"

Scratch: Final Projects

Day 3, Session 4

Broadcast Block



The **broadcast** block has a sprite send a **signal**.

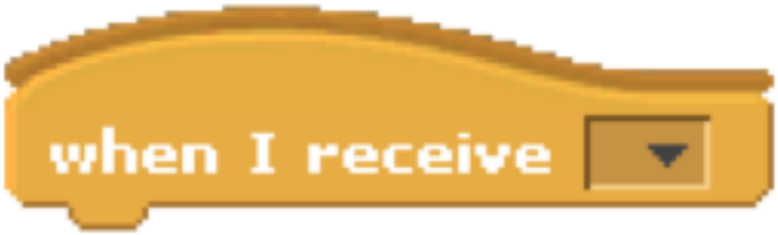
You have to give each signal a **name**.

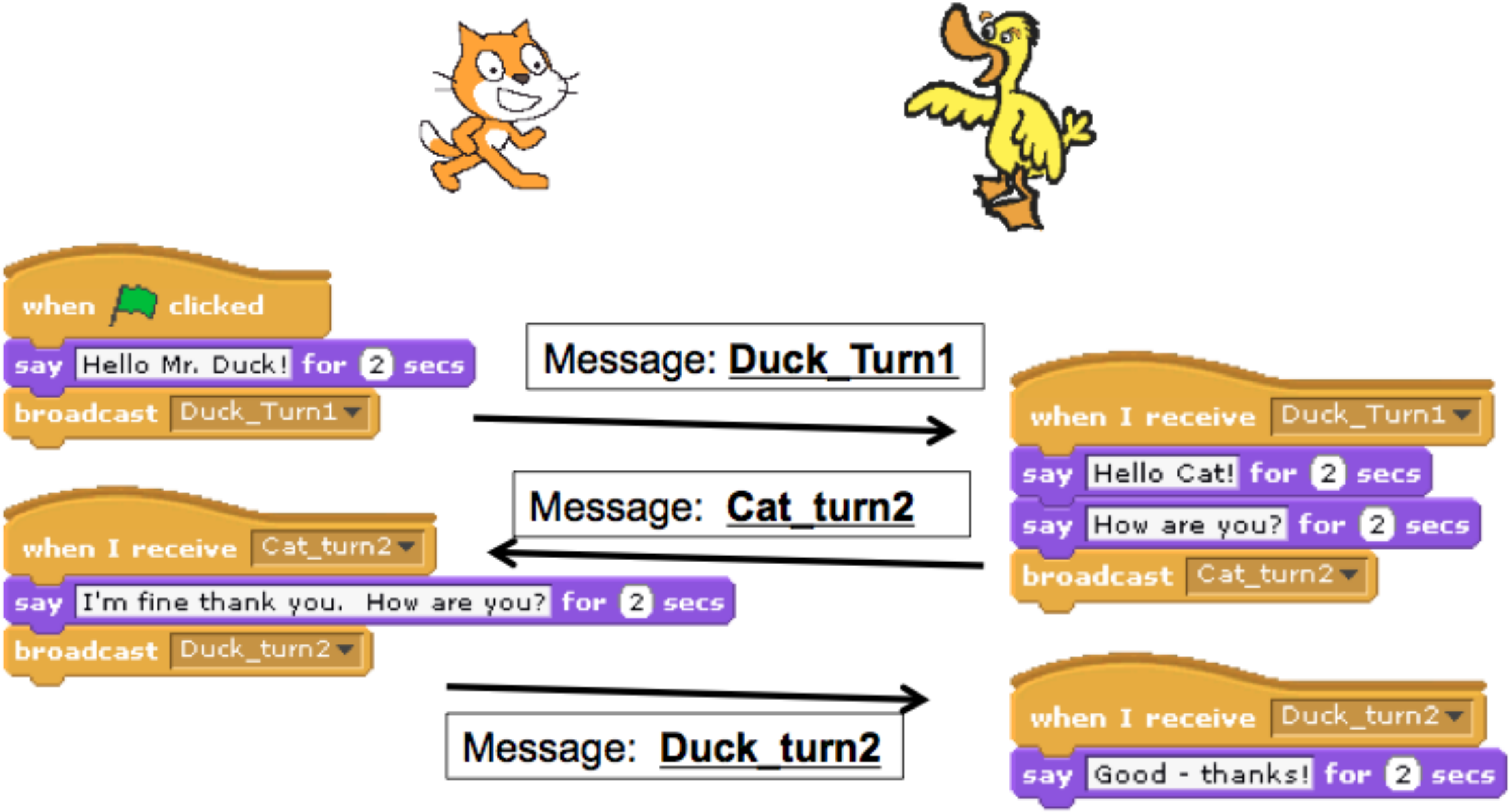
When another sprite **hears** that signal, it can start running a **new script**.

This can be helpful when you don't want one script to **start** until another script is **finished**.

Receiving a Broadcast Signal

To make a sprite listen for a signal, use the **<u>receive</u>** block, and choose which message to have it wait for.





```
when ⚑ clicked
say [Hello Mr. Duck!] for (2) secs
broadcast [Duck_Turn1 ▾]
```

Message: **<u>Duck_Turn1</u>**

```
when I receive [Duck_Turn1 ▾]
say [Hello Cat!] for (2) secs
say [How are you?] for (2) secs
broadcast [Cat_turn2 ▾]
```

Message: **<u>Cat_turn2</u>**

```
when I receive [Cat_turn2 ▾]
say [I'm fine thank you.  How are you?] for (2) secs
broadcast [Duck_turn2 ▾]
```

Message: **<u>Duck_turn2</u>**

```
when I receive [Duck_turn2 ▾]
say [Good - thanks!] for (2) secs
```

Congratulations!

ACTIVITY

You've gone through an abridged version of the Scratch curriculum!

In the time remaining, team up with a partner and create a small project to show off your Scratch skills to everyone else.