CS 101:
Preview of Computer Science

Professor
Zoran Duric cs.gmu.edu/~zduric Innovation Hall 103

Teaching Assistants
Nalini Vishnoi (Graduate), Michael Bowen, Sam Gelman, Anna Papadogiannakis

Who Can Take CS101?

- **CS / ACS majors** or **intended majors** must take this class along with their first “CS” class at GMU.
    - You can’t get out of CS101.
    - If you’re **considering being a CS / ACS major**, talk to me first.
    - You must be in CS 112 (or have credit for it)
- **If you don’t intend to be a CS major**, you cannot take this class. Talk to me about other options.
- CS 101 **does not count** for any GE or IT credit.

To Get an S (pass) You Must

- Attend the class > 90% of the time (you can miss 3 classes).
- Perform six sanctioned outside activities.
- Do a group project to our satisfaction.
- Meet your advisor and discuss your *plan of study/class schedule/remaining requirements*

Class Web Page

- **https://piazza.com/gmu/fall2013/cs101/home**
- **http://cs.gmu.edu/syllabus/syllabi-fall13/CS101DuricZ.html**
- We will post sanctioned outside activities.
- We'll post some lecture notes there.
- You'll post project proposals, etc.

Class Structure

- **Tuesday**
  I lecture on a topic of computer science, plus other stuff
- **Thursday**
  Guest faculty lecture on a cutting-edge research area in that topic
- Some weeks we may switch faculty lecture to Tuesday
- Towards the end of the semester you will present your projects on Tuesdays and Thursdays

Guest Faculty Lectures

- **CS Overview (Aug. 29)**
  Sanjeev Setia
- **Security (Sep. 19)**
  Damon McCoy
- **Software Engineering**
  Paul Amman
- **Theory (Sep. 5)**
  Dana Richards
- **Intellectual Property**
  David Grossman
- **Robotics**
  Sean Luke
- **Virtual Reality**
  Zoran Durić
- **Graphics**
  Jyh-Ming Lien
- **Distributed Networking**
  Robert Simon
- **Bioinformatics (Oct. 1 - Tuesday)**
  Huzefa Rangwala
- **Game Development (Sep. 26)**
  Yortam Gingold
- **Computer Vision**
  Jana Kosecka
- **Data Mining**
  Jessica Lin
- **Android Programming**
  Elizabeth White

The Project

- Will use one of these:

- Will be in Python.
    - Don’t know Python and didn’t take / not taking CS 112? Talk to us about what to do.
- You will propose a project.
    - Project difficulty will match your current ability.
    - We may have some warm-up assignments.

Some Definitions

- Computer Science **is ...**

  The study and analysis of computing devices and their use.

- Computer Science **is not ...**

  The study of computers.

> ***Computer Science***
> ***is no more about computers***
> ***than Astronomy***
> ***is about telescopes.***
>
> **— Edsger Dijkstra**

- The study of computers (that is, **hardware**), is called **Computer Engineering.** It's taught by another department.

Computing Devices: Automata

- **Automata** are devices (in real life or existing only abstractly in math) which are capable of following a collection of rules.

- **A computer is an automaton.**

- **Fingers,**
  Playing Cards,
  and **Abaci**

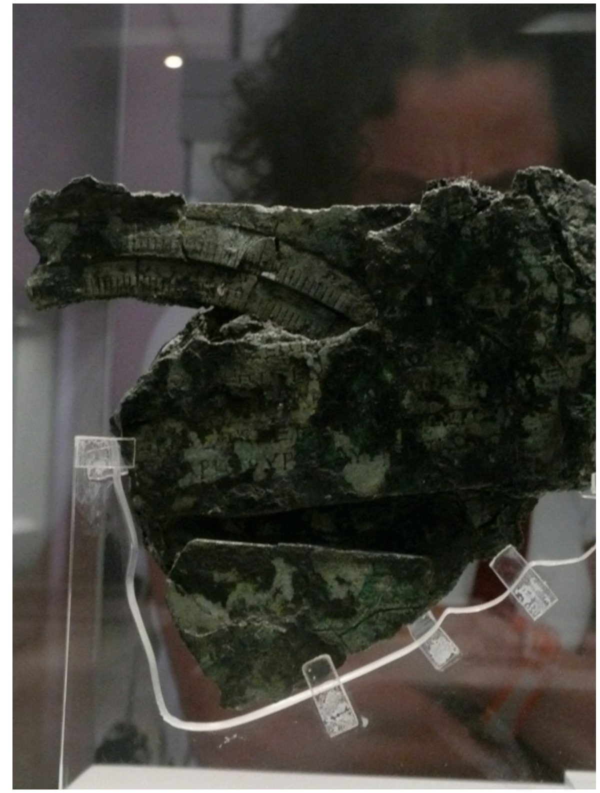
  ... **are also automata** if you use them with a set of rules decided in your mind.

An Ancient Automaton

At the Athens Archeological Museum (ca. 150 BC)

The Antikythera Mechanism





Probably 37 gears.

Computes the location of the **Sun**, the **Moon**, the **Zodiac**, **Lunar Phase**, likely **Planetary Motions,** and possibly the **Dates for the Olympics.**

Another Automaton

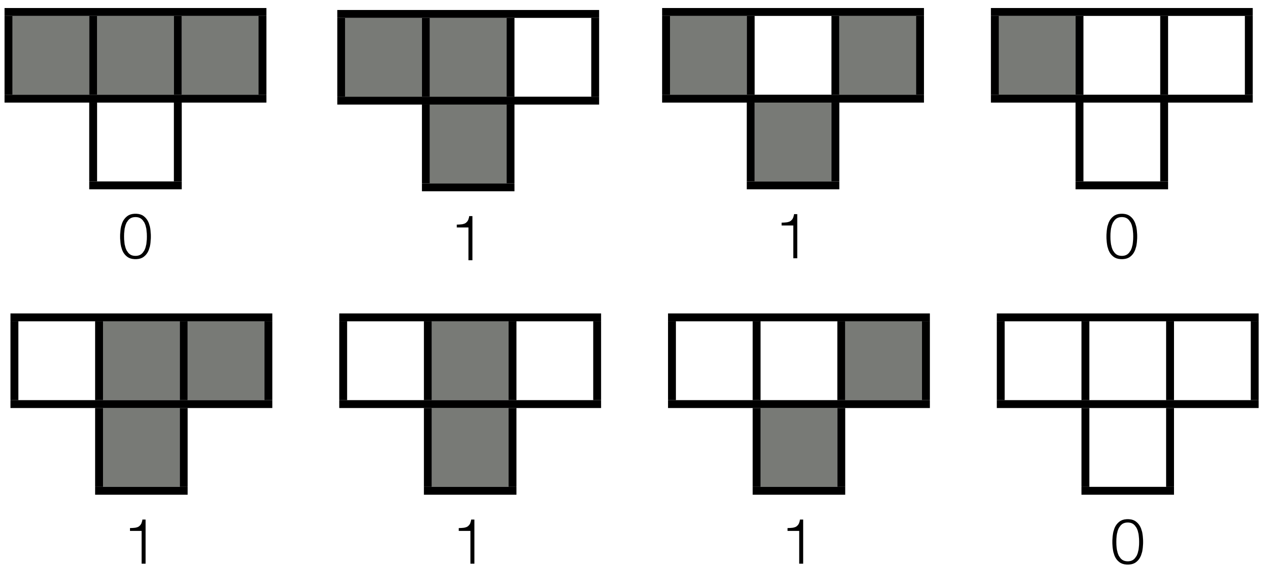
Chose Jurors in Athens

Algorithms and Programs

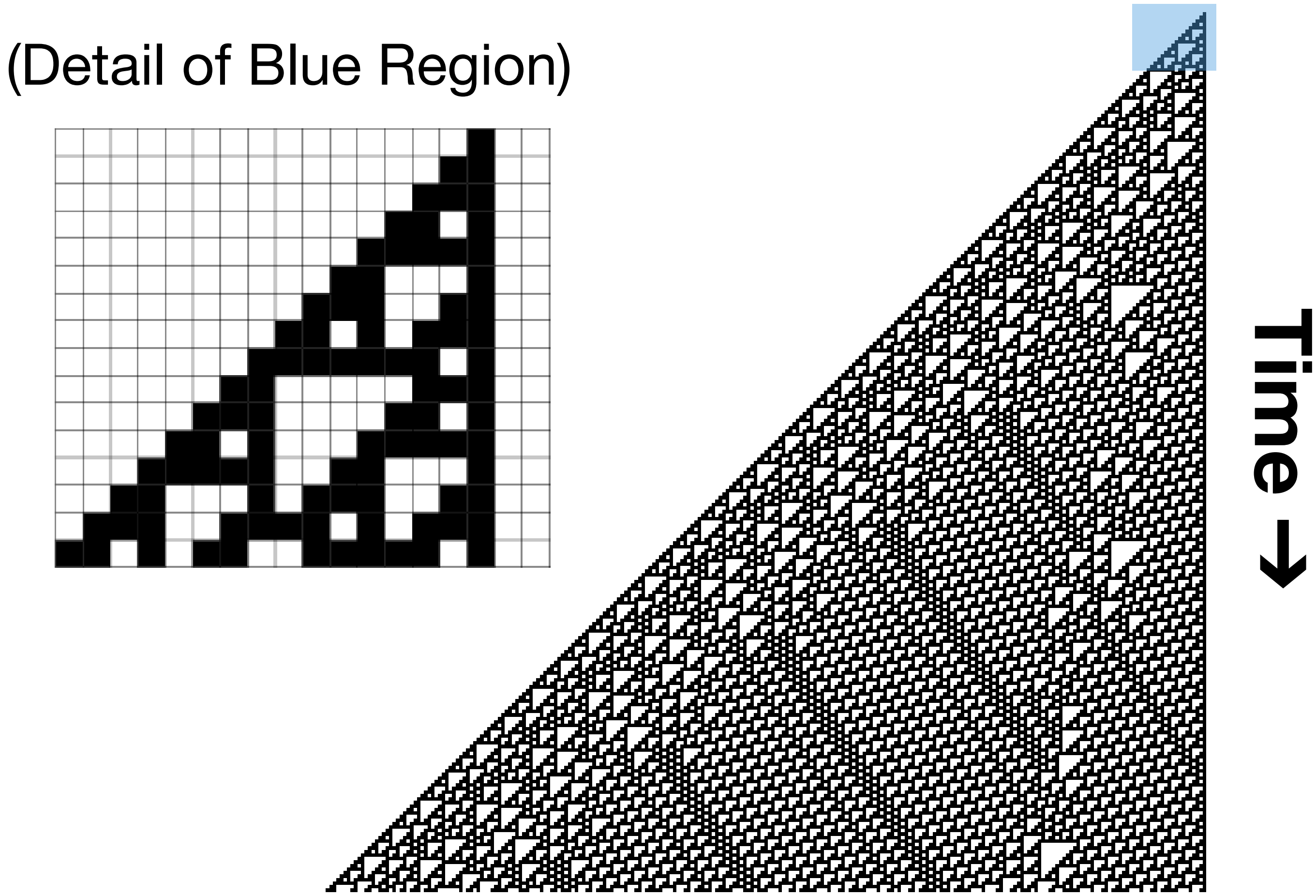
- An **algorithm** is a set of rules for an automaton to follow
    - **Al-Kwarizimi** wrote *On Calculation with Hindu Numerals,* later translated (poorly) into Latin as ***Algoritmi** on the Numbers of the Indians.*
    - His name then got mangled into **Algorithm.**
- A **program** is the **instantiation of an algorithm** designed for an actual computer to follow.

An Algorithm

- **Rule 01101110** of a one-dimensional binary 3-neighborhood cellular automaton.
  http://mathworld.wolfram.com/ElementaryCellularAutomaton.html



Results



Look Familiar?

Cone Snail

Another Algorithm

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0$$

$$F_1 = 1$$

Results

- 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025 121393 196418 317811 514229 832040 1346269 2178309 3524578 5702887 9227465 14930352 24157817 39088169 63245986 102334155 165580141 267914296 433494437 701408733 1134903170 1836311903 2971215073 4807526976 7778742049 12586269025 20365011074 32951280099 53316291173 86267571272 139583862445 225851433717 365435296162 591286729879 956722026041 1548008755920 2504730781961 4052739537881 6557470319842 10610209857723 17167680177565 27777890035288 44945570212853 72723460248141 117669030460994 190392490709135 308061521170129 498454011879264 806515533049393 1304969544928657 2111485077978050 3416454622906707 5527939700884757 8944394323791464 14472334024676221 23416728348467685 37889062373143906 61305790721611591 99194853094755497 160500643816367088 259695496911122585 420196140727489673 679891637638612258 1100087778366101931 1779979416004714189 2880067194370816120 4660046610375530309 7540113804746346429 12200160415121876738 19740274219868223167 31940434634990099905 51680708854858323072 83621143489848422977 135301852344706746049 218922995834555169026

Another Algorithm

Start:	**A**		
Rules:	**A**	**→**	**B – A – B**
	B	**→**	**A + B + A**
	–	**→**	**–**
	+	**→**	**+**

I Cheated

```
(defun l–expand (seq &rest rules)
   (reduce #'append (mapcar
         (lambda (item) (second (apply #'assoc item rules)))
         seq)))

(defvar *rules*       '((A (B – A – B))
                        (B (A + B + A))
                        (– (–))
                        (+ (+))))

(defvar *seq*         '(A))

(loop (print (setf *seq* (l–expand *seq* *rules*))))
```

Results

A

B - A - B

A + B + A - B - A - B - A + B + A

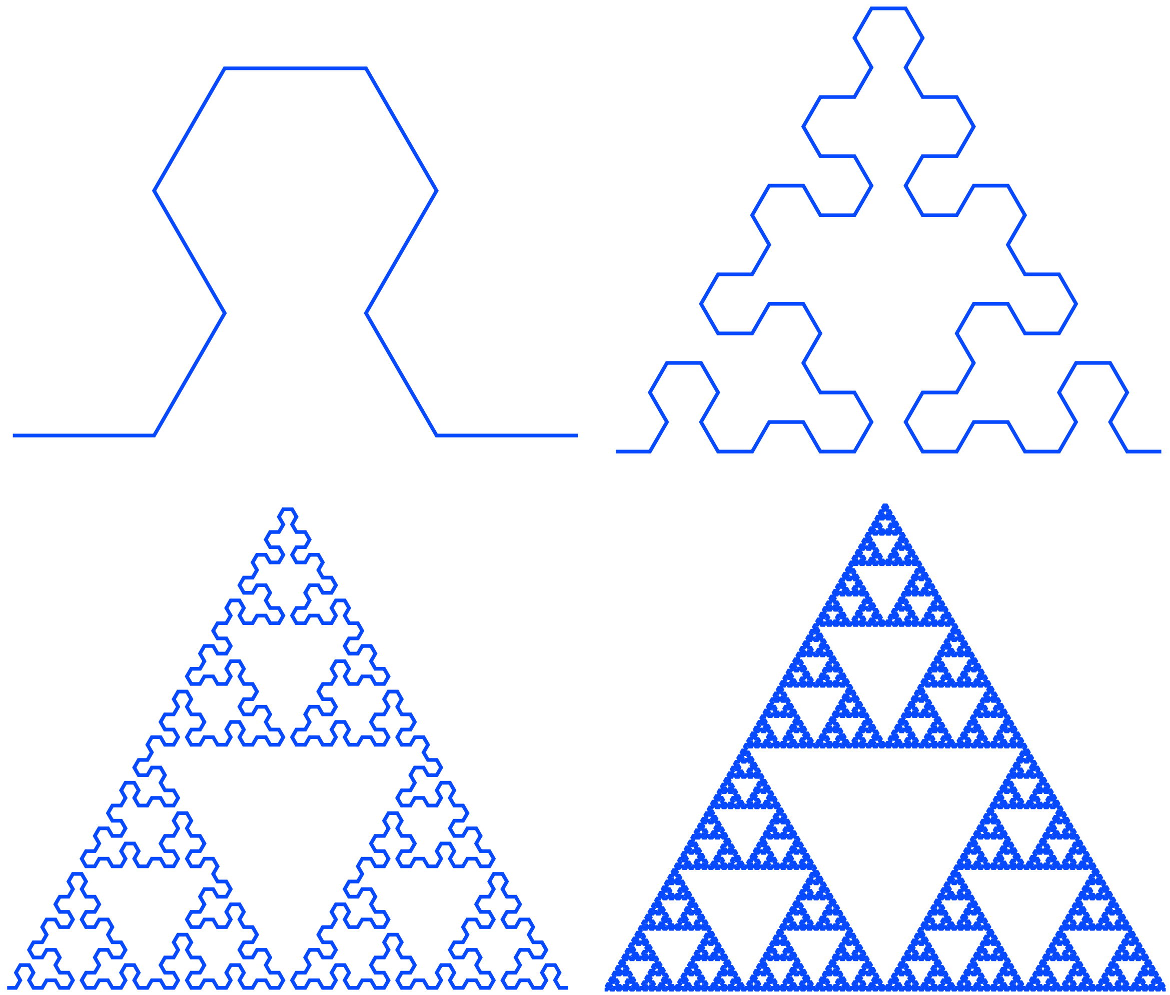
B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A - B - A - B + A + B
+ A + B - A - B

A + B + A - B - A - B - A + B + A + B - A - B + A + B + A + B - A - B + A + B + A - B - A - B - A + B + A - B - A - B +
A + B + A + B - A - B - A + B + A - B - A - B - A + B + A - B - A - B + A + B + A + B - A - B - A + B + A - B - A - B -
A + B + A + B - A - B + A + B + A + B - A - B + A + B + A - B - A - B - A + B + A

B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A - B - A - B + A + B + A + B - A - B + A + B + A -
B - A - B - A + B + A + B - A - B + A + B + A + B - A - B + A + B + A - B - A - B - A + B + A + B - A - B + A + B + A
+ B - A - B - A + B + A - B - A - B - A + B + A - B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A
+ B - A - B + A + B + A + B - A - B + A + B + A - B - A - B - A + B + A - B - A - B + A + B + A + B - A - B - A + B + A
- B - A - B - A + B + A - B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A + B - A - B + A + B + A
+ B - A - B + A + B + A - B - A - B - A + B + A - B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A
- B - A - B + A + B + A + B - A - B + A + B + A - B - A - B - A + B + A + B - A - B + A + B + A + B - A - B + A + B +
A - B - A - B - A + B + A + B - A - B + A + B + A + B - A - B - A + B + A - B - A - B - A + B + A - B - A - B + A + B +
A + B - A - B

Results

- A, B   Draw Forward
- +   Turn Left 60°
- –   Turn Right 60°
- Iterations: 2, 4, 6, 9



One More

Start:	**A**		
Rules:	**A**	➔	**B**
	B	➔	**A B**

Results

A

B

A B

B A B

A B B A B

B A B A B B A B

A B B A B B A B A B B A B

B A B A B B A B A B B A B B A B A B B A B

A B B A B B A B A B B A B B A B A B B A B A B B A B B A B A B B A B

Are these the same?

Start: **A**

Rules:

A ➔ **B**

B ➔ **A B**

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0$$

$$F_1 = 1$$

Solitaire: A Cryptographic Algorithm (Schneier)

- You and your buddy each have a **pack of cards with Jokers**, randomly shuffled but in the exact same card order.
- To encrypt the next letter in your message, you follow certain card movement rules described later, then you note a certain card in a **special position.**
- If this card is an **Ace**, it counts as a **1**. A **Jack**, is an **11**. A **Queen** is a **12**. A **King** is a **13**. **Numbered cards** are **their own value**. If the card is a **Joker**, repeat the card movement rules until you get a non-Joker.
- If the card is **black**, add 13. Thus your possible values are **1** (red Ace) through **26** (black King).
- To encrypt the next letter, convert it to a number (**A ➔ 1, B ➔ 2, ...**), then add your card value. If the result is over 26, subtract 26. Convert back to a letter. Send that letter to your buddy.
- To decrypt, take the “letter” you got, convert to a number, and *subtract* your card value. If the result is less than 1, add 26. Convert back to a letter.

http://www.schneier.com/solitaire.html

The Card Movement Operation

1. Move the topmost joker underneath the card just below it.

2. Move the bottommost joker underneath the card **two cards** below it.

3. Exchange the chunk of cards **above but not including** the current topmost joker with the chunk of cards **below but not including** the bottommost joker.

4. Read the card at the bottom of the deck. Convert it into its **number** (we'll call it **N**). Remove the chunk of N cards from the top of the deck and put them near the bottom of the deck, just in front of the bottom-most card.

5. Read the top card. Convert it into its **number** (we'll call it **M**). Find the **M**th card from the top. **This is the card in the special position.**

Automata as Models of Life

Start: **X**

Rules: **X → F – [[X] ++ X] + F [+ F X] – X**

F → F F

– → –

+ → +

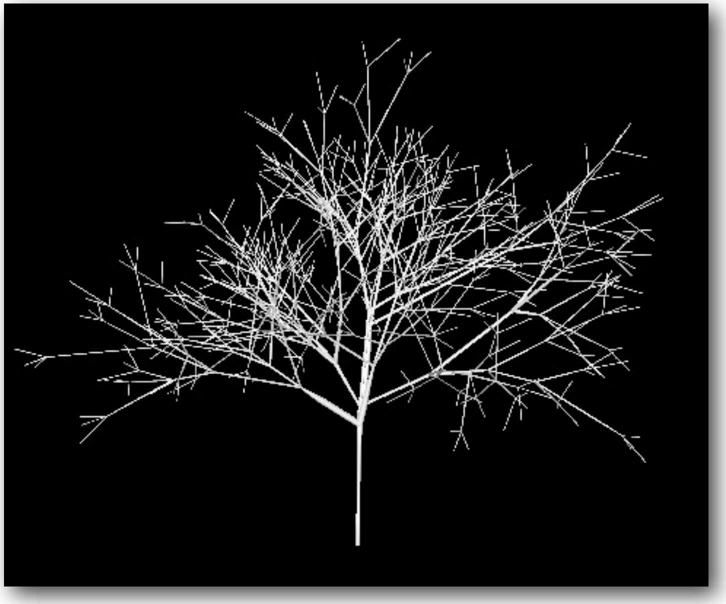
[→ [

] →]

Results

- F Draw Forward
- X Do Nothing
- + Turn Right 25°
- – Turn Left 25°
- [Remember this position
-] Teleport to most recently remembered current position (and forget it)
- Iterations: 6

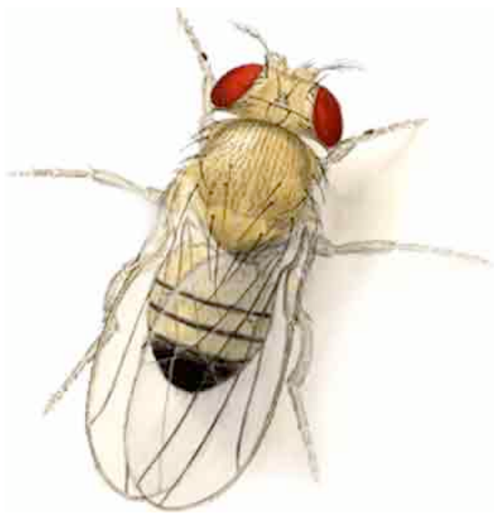
More Examples

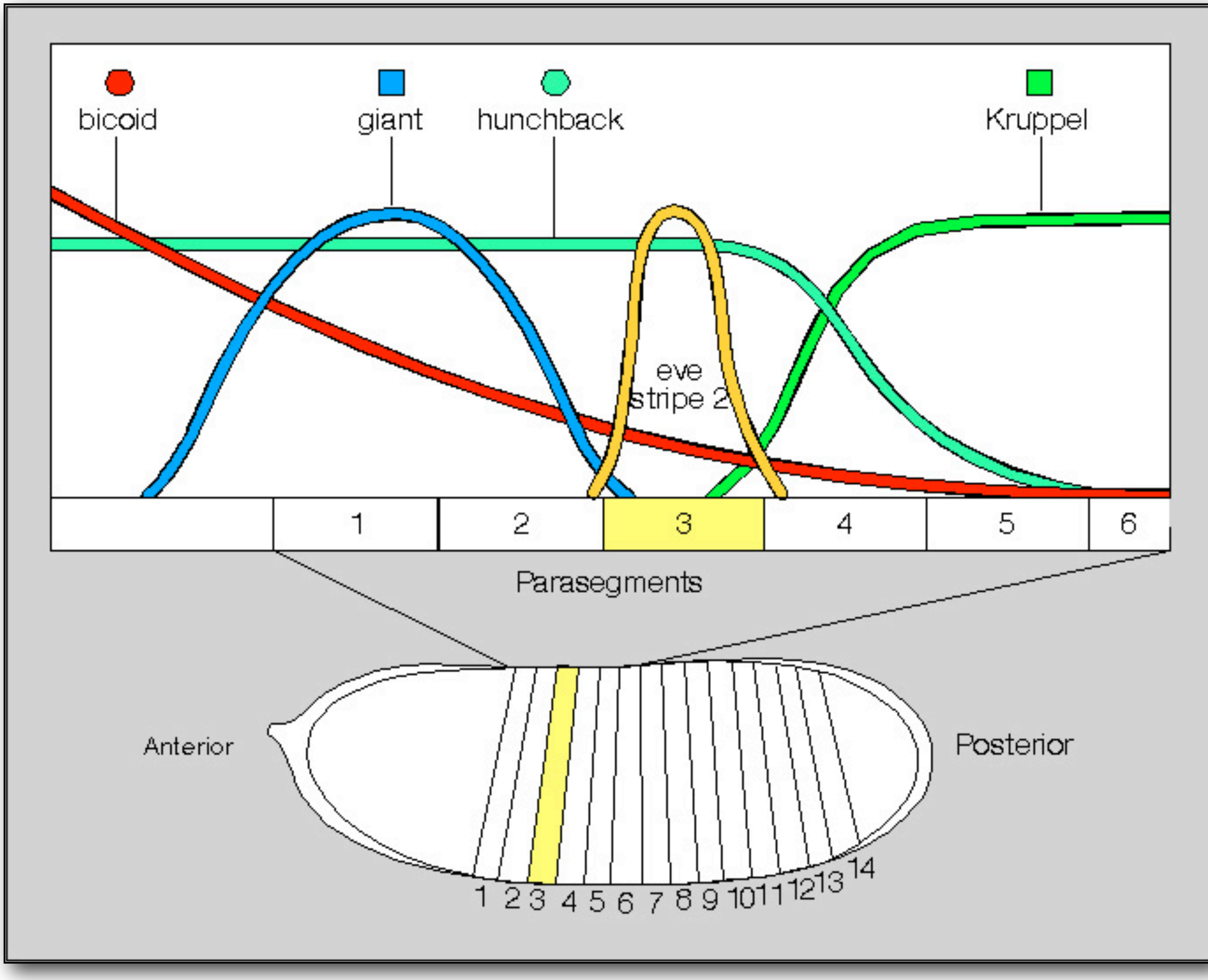


Yet Another Automaton

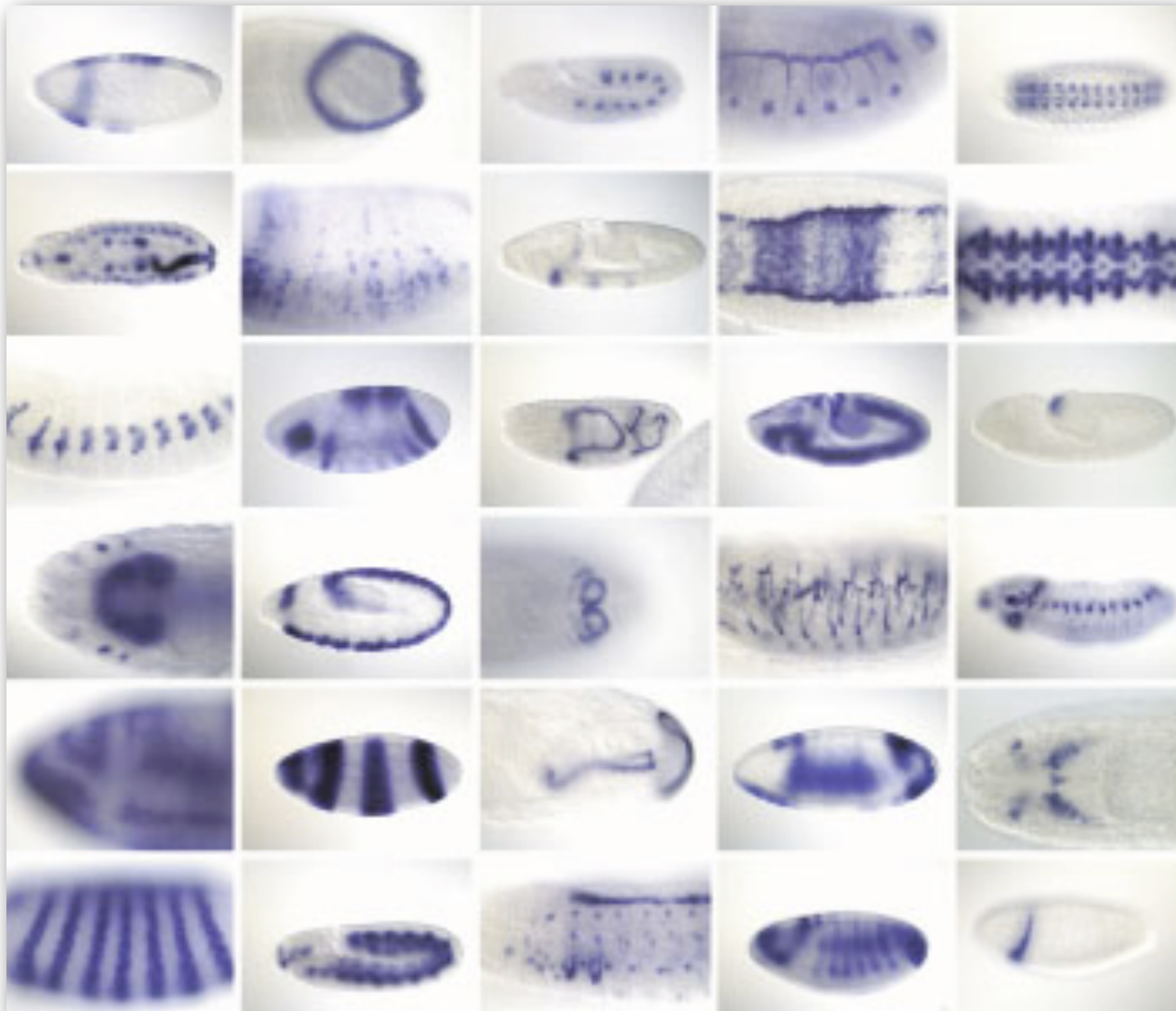
- Gene Regulatory Networks
  - DNA mostly makes RNA
  - RNA mostly makes Protein
  - Various RNA and Protein can **inhibit** or **promote** the production of RNA from DNA, or of Protein from RNA.
  - These regulations are often **mutual.**

Top-level gene regulatory network in *Drosophila melanogaster* embryo



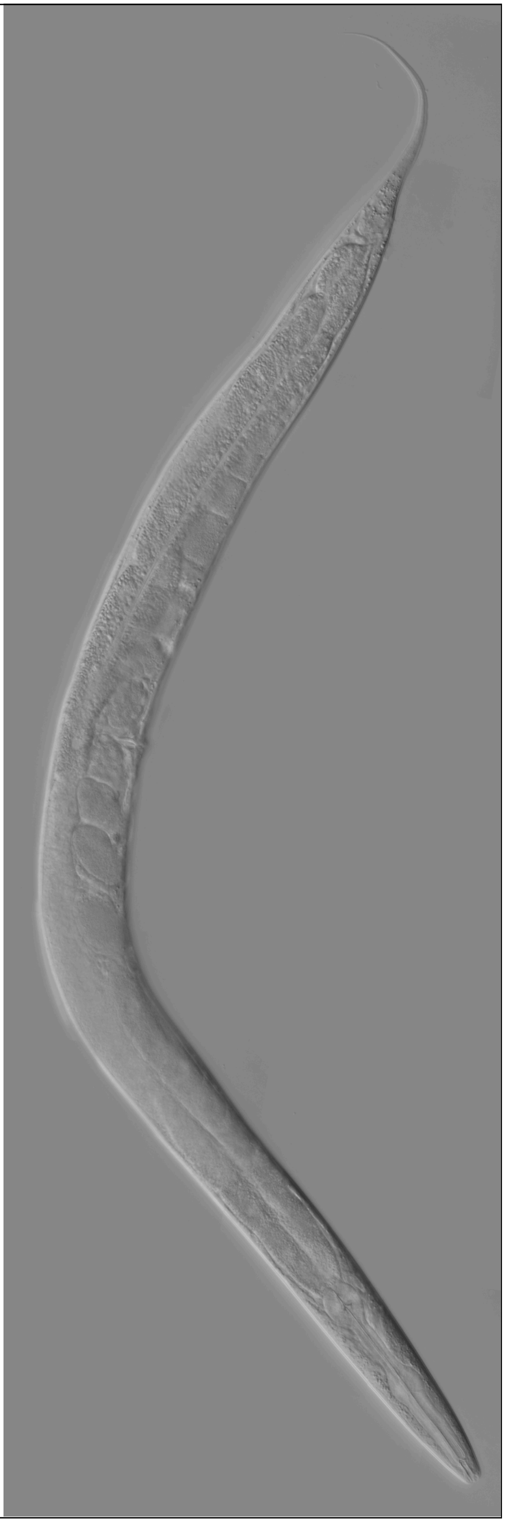


Results



One Last Automaton

- *Caenorhabditis elegans*
- Smallest differentiated multicellular organism
- 959 cells
- 302 neurons (it's ⅓ brain!)
- ~7000 synapses



Are These the Same?

- The neurons in *Caenorhabditis elegans* above are **remarkably similar** to the nodes in the automaton below.
- *Caenorhabditis elegans* has about **300 neurons**.
- The **Human Brain** has about **100 million neurons**
- So are we automata? Is it just an issue of scaling up?

$$F_n = F_{n-1} + F_{n-2}$$
$$F_0 = 0$$
$$F_1 = 1$$