Introduction to Informatics

Lecture 14: Logic Circuits and Midterm Review

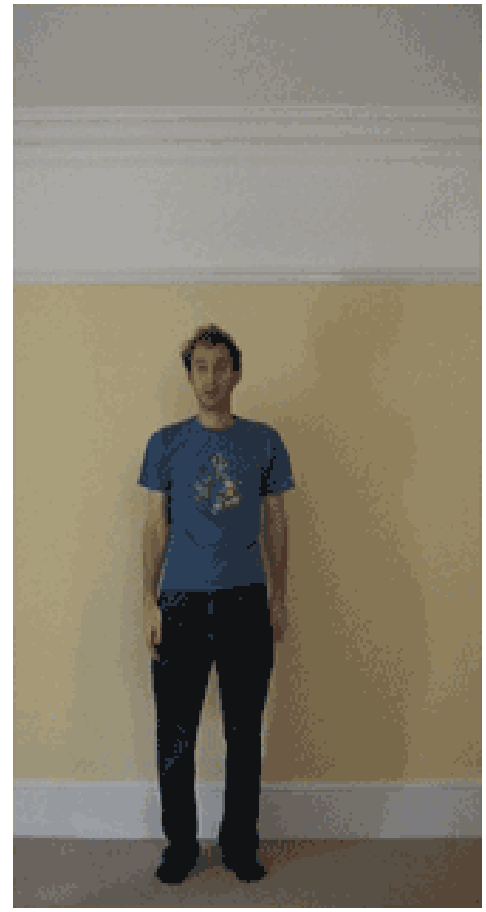
Luis M.Rocha and Santiago Schnell

Readings until now

- Lecture notes
  - Posted online
    - http://informatics.indiana.edu/rocha/i101
      - *The Nature of Information*
      - *Technology*
      - *Modeling the World*
  - @ *infoport*
    - *http://infoport.blogspot.com*
  - From course package
    - Von Baeyer, H.C. [2004]. *Information: The New Language of Science*. Harvard University Press.
      - Chapters 1, 4 (pages 1-12)
    - From Andy Clark's book "*Natural-Born Cyborgs*"
      - Chapters 2 and 6 (pages 19 - 67)
    - From Irv Englander's book "*The Architecture of Computer Hardware and Systems Software*"
      - Chapter 3: Data Formats (pp. 70-86)
    - Klir, J.G., U. St. Clair, and B.Yuan [1997]. Fuzzy Set Theory: foundations and Applications. Prentice Hall
      - Chapter 2: Classical Logic (pp. 87-98)
      - Chapter 3: Classical Set Theory (pp. 99-107)

Assignment Situation

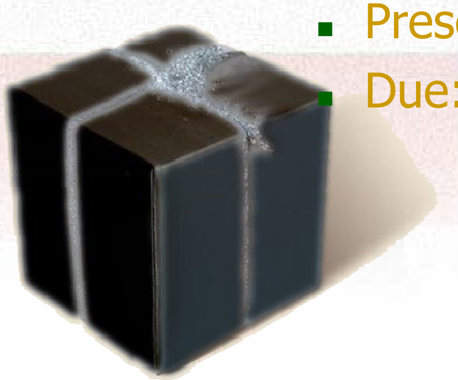
- Labs
  - Past
    - Lab 1: Blogs
      - Closed (Friday, January 19): Grades Posted
    - Lab 2: Basic HTML
      - Closed (Wednesday, January 31): Grades Posted
    - Lab 3: Advanced HTML: Cascading Style Sheets
      - Closed (Friday, February 2): Grades Posted
    - Lab 4: More HTML and CSS
      - Closed (Friday, February 9): Grades Posted
    - Lab 5: Introduction to Operating Systems: Unix
      - Closed (Friday, February 16): Being graded
    - Lab 6: More Unix and FTP
      - Due Friday, February 23
  - Next: Lab 7
    - Logic Gates
      - March 1 and 2, due Friday, March 9
- Assignments
  - Individual
    - First installment
      - Closed: February 9: Grades Posted
    - Second Installment
      - Due: March: 2nd
  - Group Project
    - First installment
      - Presented: March 6, Due: March 9th
- Midterm Exam
  - March 1st (Thursday)



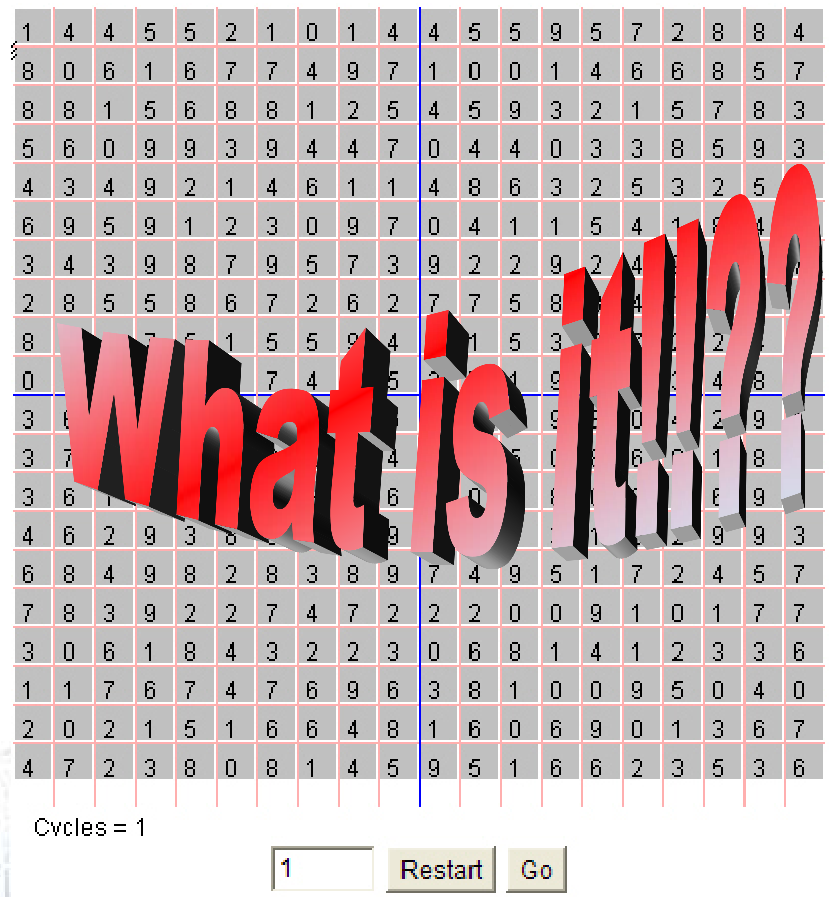
Get a Group!

Individual assignment

- Individual Project
  - 1st installment
    - Presented: February 1st
    - Due: February 9th
  - 2nd Installment
    - Presented: February 15th
    - Due: March: 2nd
  - 3rd Installment
    - Presented: March 8th
    - Due: March 30th
  - 4th Installment
    - Presented: April 5th
    - Due: April 20th



The Black Box



Fundamental Logic Operations/Gates

NOT

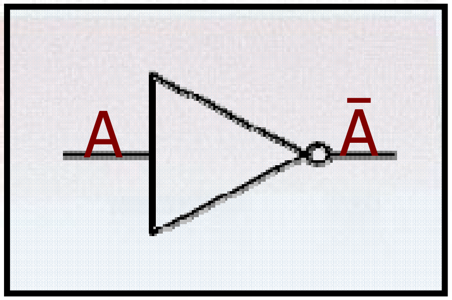
A,p	$X = \bar{A},\ \neg p$
0	1
1	0

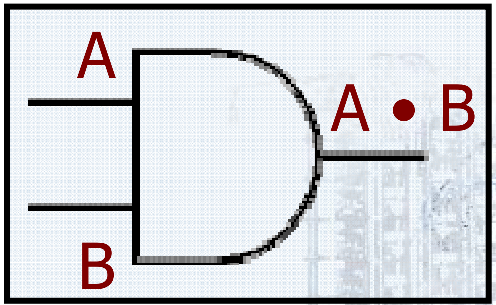
AND

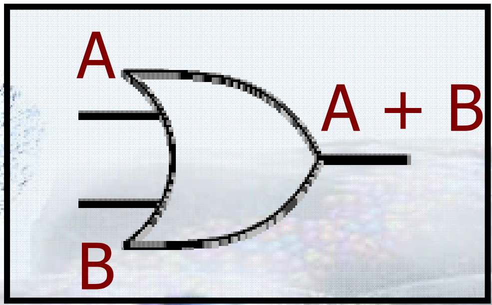
A,p	B,q	$A \bullet B,\ p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

OR

A,p	B,q	$A+B,\ p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1



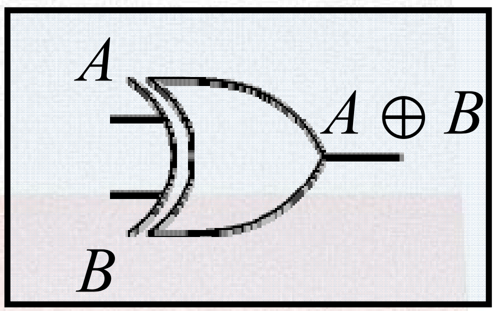




Important Logic Operations/Gates

XOR

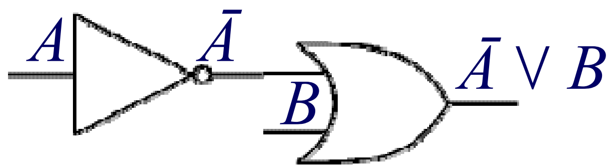
A,p	B,q	$A \oplus B$, $p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0



$$p \oplus q \equiv (p \vee q) \wedge \neg(p \wedge q)$$

IMPLICATION

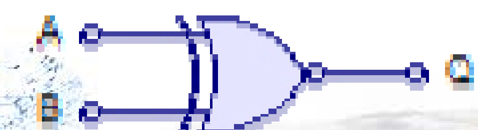
A,p	B,q	$A \Rightarrow B$, $p \Rightarrow q$
0	0	1
0	1	1
1	0	0
1	1	1



$$p \Rightarrow q \equiv \neg p \vee q$$

EQUIVALENCE

A,p	B,q	$A \Leftrightarrow B$, $p \Leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1



$$p \Leftrightarrow q \equiv \neg(p \oplus q)$$

Proofs for De Morgan's Law I

$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

p	q	$\neg p$	$\neg q$	$\neg p \wedge \neg q$	$p \vee q$	$\neg(p \vee q)$
0	0	1	1	1	0	1
0	1	1	0	0	1	0
1	0	0	1	0	1	0
1	1	0	0	0	1	0

Rules of replacement

- *De Morgan's Law I*

$$\neg(p \vee q) \equiv \neg p \wedge \neg q$$

- *De Morgan's Law II*

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$

Proof for the distributive rule

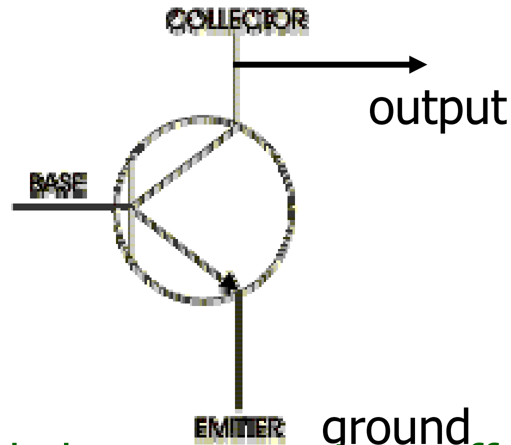
$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

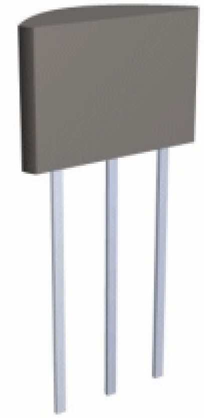
p	q	r	$q \wedge r$	$p \vee (q \wedge r)$	$p \vee q$	$p \vee r$	$(p \vee q) \wedge (p \vee r)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

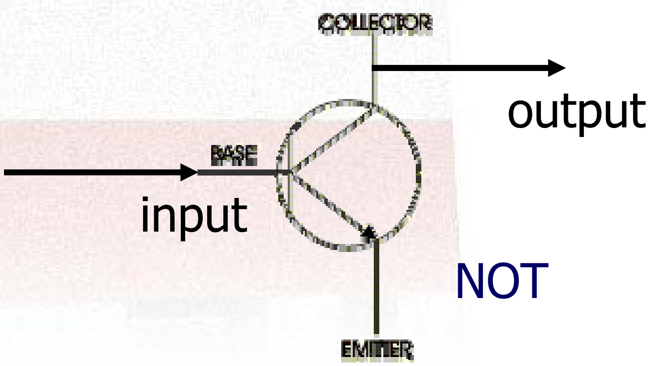
Adapted from C. Heeren

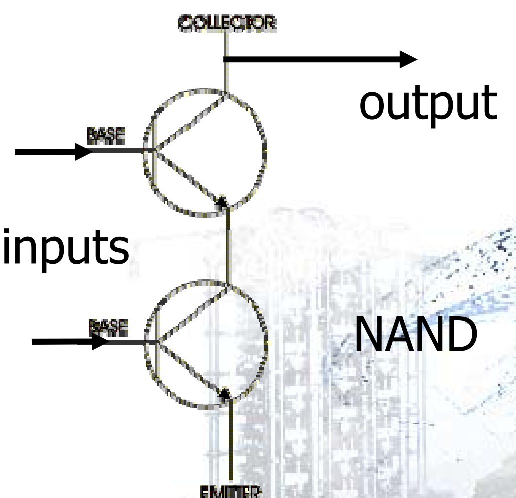
Implementing Logic Gates

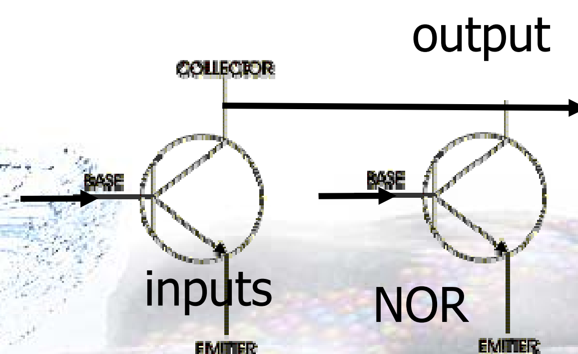
- Transistor
  - Source or collector
    - Produces high voltage
      - 5 volts
  - Base regulates
    - If signal high
      - Source signal gets grounded: turns transistor off (0)
    - If signal is low
      - Source signal stays high: transistor on (1)
  - NOT, NAND and NOR gates easiest to produce











Computing with Logical Gates

Adding Binary Numbers

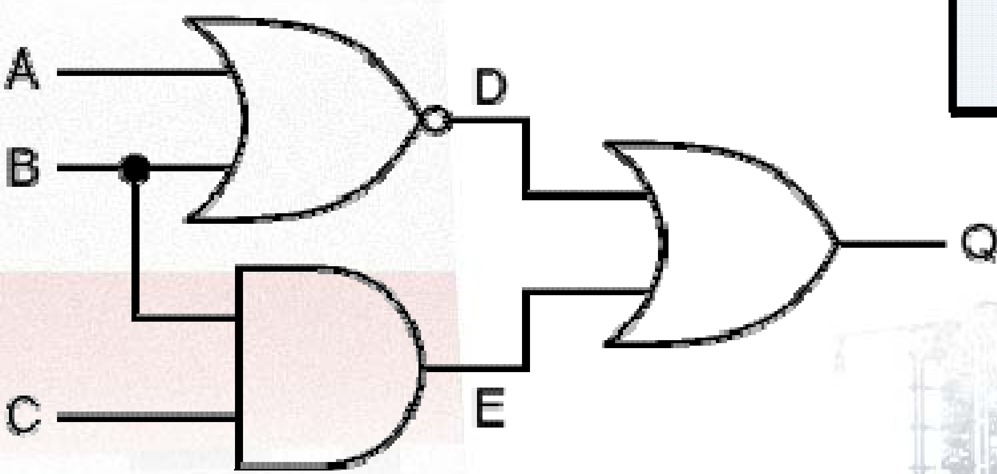
0 + 0 = 0

0 + 1 = 1

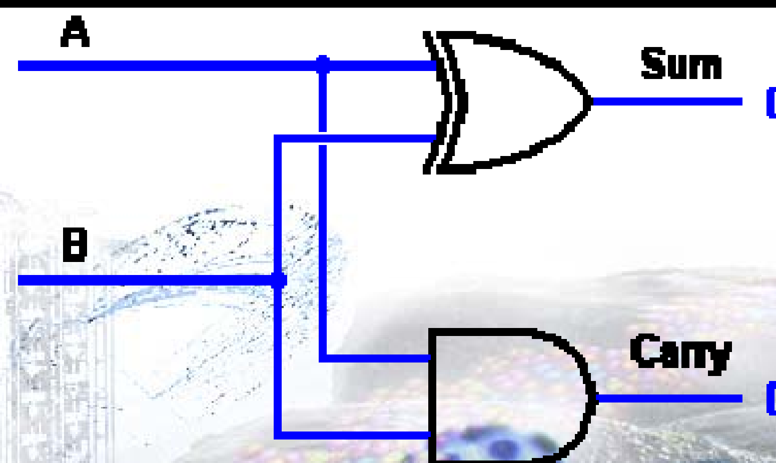
1 + 0 = 1

1 + 1 = 10

A	*B*	CARRY (AND)	SUM (XOR)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



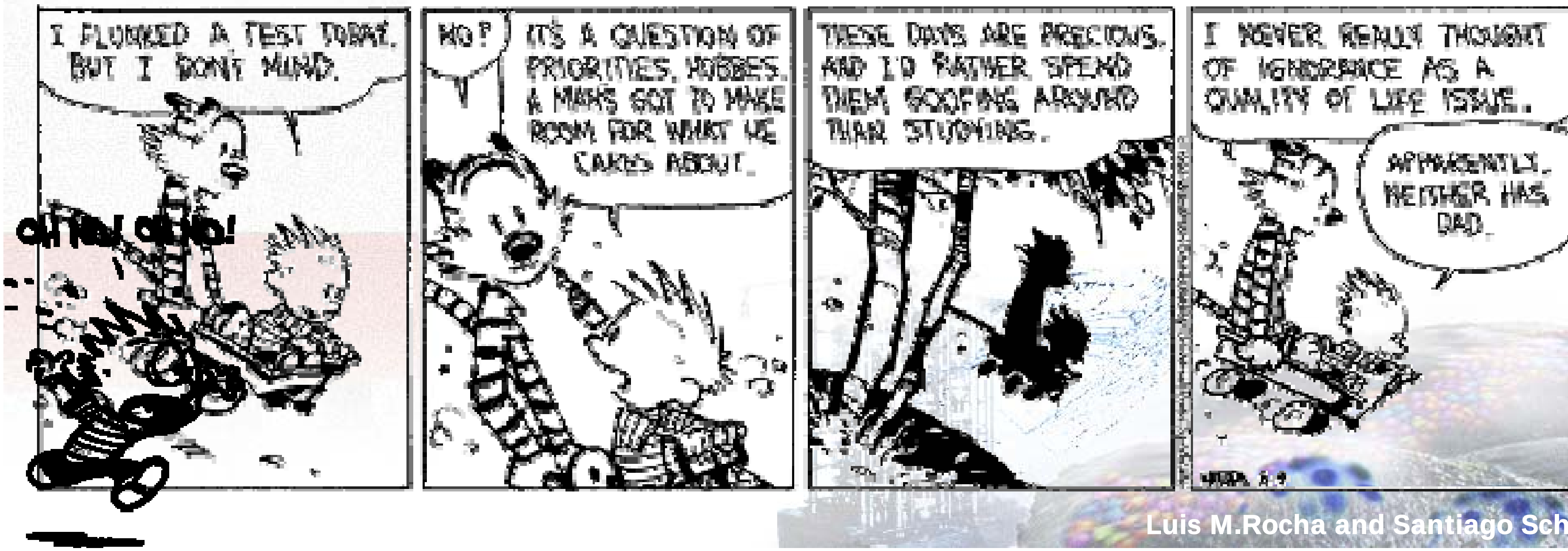
(NOT (A OR B)) OR (B AND C)



Luis M.Rocha and Santiago Schnell

Midterm Exam

- March 1st (Thursday)
  - Regular Class time



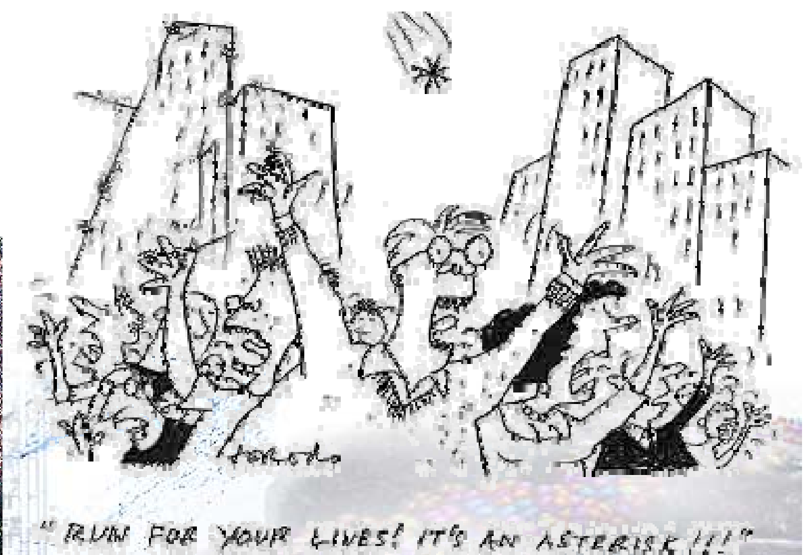
The Nature of Information

Symbols and Information Quantity

Luis M.Rocha and Santiago Schnell

Information is a Relation!

- The central structure of information is a *relation*
  - among *signs, objects or things*, and *agents* capable of understanding (or decoding) the signs.
- *Agents* are informed by a *Sign* **about** some *Thing*.



Semiotics and Informatics

Information

- Semantics
  - the content or **meaning** of the *Sign* of a *Thing* for an *Agent*
    - Relations between signs and objects for an agent
    - the study of meaning.
- Syntax ⟸ **Information Technology**
  - the characteristics of *signs* and symbols devoid of meaning
    - Relations among signs such as their rules of operation, production, storage, and manipulation.
- Pragmatics
  - the context of signs and repercussions of sign-systems in an environment
    - it studies how context influences the interpretation of signs and how well a signs-system represents some aspect of the environment

Informatics

(Peirce's) Typology of Signs

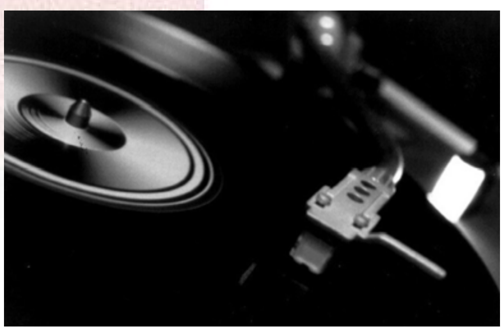
- ***Icons*** are direct representations of objects.
  - Similar to the thing they represent.
  - Pictorial road signs, scale models, computer icons.
    - A footprint on the sand is an icon of a foot.
- ***Indices*** are indirect representations of objects, but necessarily related.
  - Smoke is an index of fire, the bell is an index of the tolling stroke
    - a footprint is an index of a person.
- ***Symbols*** are arbitrary representations of objects
  - Require exclusively a social convention to be understood
    - Convention establishes a code, agreed by a group of agents, for understanding (decoding) the information contained in symbols.
    - Smoke is an index of fire, but if we agree on an appropriate code (e.g. Morse code) we can use smoke signals to communicate symbolically.

The Bit

- Shannon used the binary system because it is the most economical
  - Uses less memory
  - *Information quantity* depends on the number of alternative message ***choices*** encoded in the binary system
- Bit (short for *binary digit*) is the most elementary choice one can make
  - Between two items: "0' and "1", "heads" or "tails", "true" or "false", etc.
  - Bit is equivalent to the choice between two equally likely choices
    - Example, if we know that a coin is to be tossed, but are unable to see it as it falls, a message telling whether the coin came up heads or tails gives us one bit of information

Digital versus Analog

- *Digital* is used to convey the notion of discrete objects/values
  - Things we can count
  - The word digit comes from the Latin word for finger (*digitus*)
  - Digital information is equivalent to symbolic information
    - Any symbol system requires a set of discrete symbols for setting up an arbitrary semantic relation
- Analog (or Analogue)
  - Information transmission via electrical, mechanical, hydraulic, and sound signals
    - Continuously varying signals which are not countable
    - What was used up until Shannon
  - Instead of messages being arbitrarily encoded, analog signals rely on some physical property of the medium
  - It implies an analogy between cause and effect, input and output
    - Voltage as an "analogy" to sound in analog synthesizer
    - But it cannot encode any sound whatsoever!
    - Sounds depend on the physical properties of electricity, the transducer and equipment used

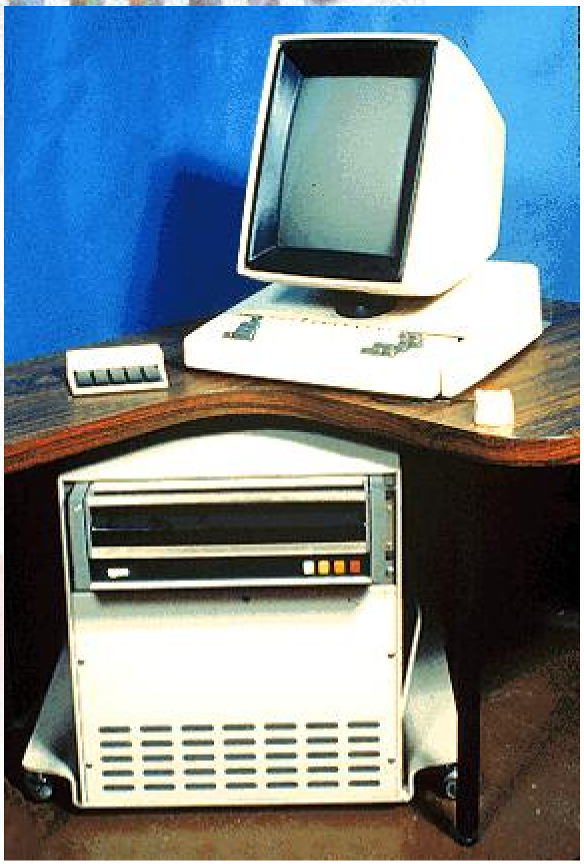


Questions

- What is informatics?
- What is the difference between an "index" and an "symbol"?
- Examples of Analogue vs. Digital Information?
- How does Information Technology relate to semiotics?

Technology

Tools, Cyborgs and History of IT



Luis M.Rocha and Santiago Schnell

Transparent Technology

- So well fitted to, and integrated with, our own lives, biological capacities, and projects as to become almost invisible in use (Andy Clark)
  - Glasses, wrist-watches, driving cars, mobile phones, pens, sports and musical equipment: human-centered
    - Not the same as easy to understand
- Opaque Technology
  - Highly visible in use: technology-centered
    - Computers, industrial machines
  - Opaque technology can become transparent with practice
    - But it works better when biologically suited
      - Natural fit, ergonomics

http://www.baddesigns.com/examples.html

http://www.jnd.org/

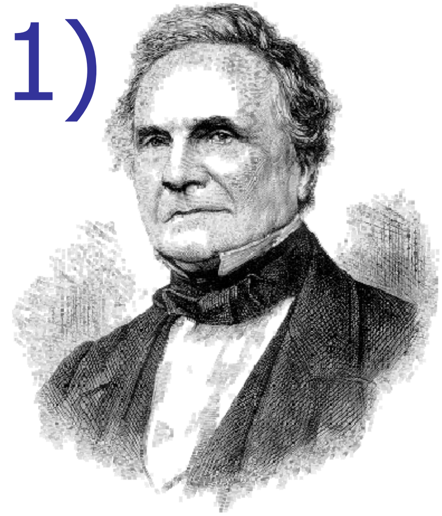
(Donald Norman)

Natural-born Cyborgs?

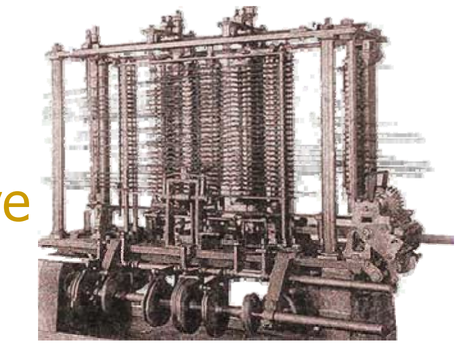
- Humans more than using, incorporate technology
  - We know we "know" the time, simply because we are equipped with a watch
  - As more portable computing devices become available, will we incorporate easily accessible collective knowledge as our own?
    - Transparent ***knowledge*** technology
      - Example: Google SMS
      - Adaptive Knowledge Technology (Clark, Chapter 6)

http://www.google.com/sms/howtouse.html#top

Charles Babbage (1791 – 1871)



- Analytical Engine
  - Working with Ada Lovelace (daughter of Lord Byron) designed what was to have been a general-purpose mechanical digital computer.



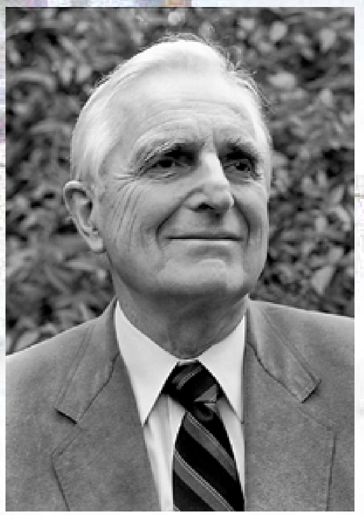
    - With a memory store and a central processing unit (or 'mill') and would have been able to select from among alternative actions consequent upon the outcome of its previous actions
      - Conditional branching: Choice, information
    - Programmed with instructions contained on punched cards

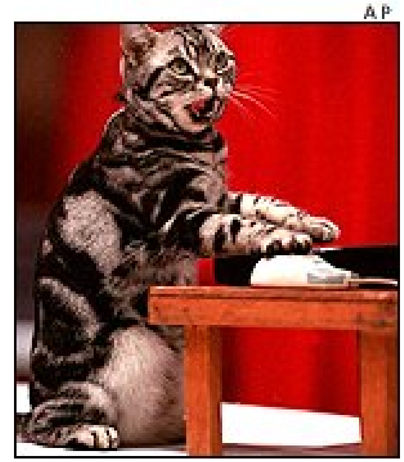
The First Personal Computer

- In 1971, Intel released the first microprocessor.
  - Able to process four bits of data at a time!
- The Altair 8800 (1975)
  - by a company called *Micro Instrumentation and Telementry Systems* (MITS) sold for $397
  - Came as a kit for assembly who had to to write software for the machine
    - in machine code!
  - 256 byte memory --about the size of a paragraph

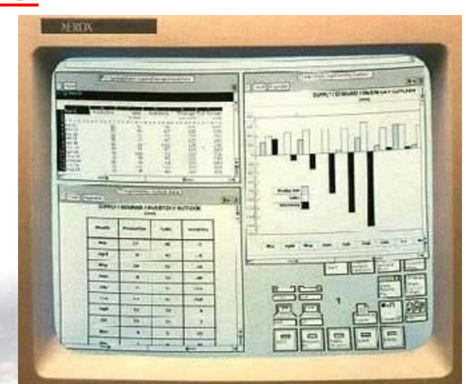
- Microsoft
  - Was born to create a BASIC compiler for the Altair
    - Beginners All-purpose Symbolic Instruction Code

graphical user interface (GUI)





- On-Line System (NLS) (1960s)
  - Doug Engelbart's *Augmentation of Human Intellect* project @ Stanford Research Institute
    - pioneer of human-computer interaction
    - also developed *hypertext*
  - Incorporated a mouse-driven cursor and multiple windows.
    - WIMP (windows, icons, menus and pointers)
    - See his demo
      - http://sloan.stanford.edu/MouseSite/1968Demo.html



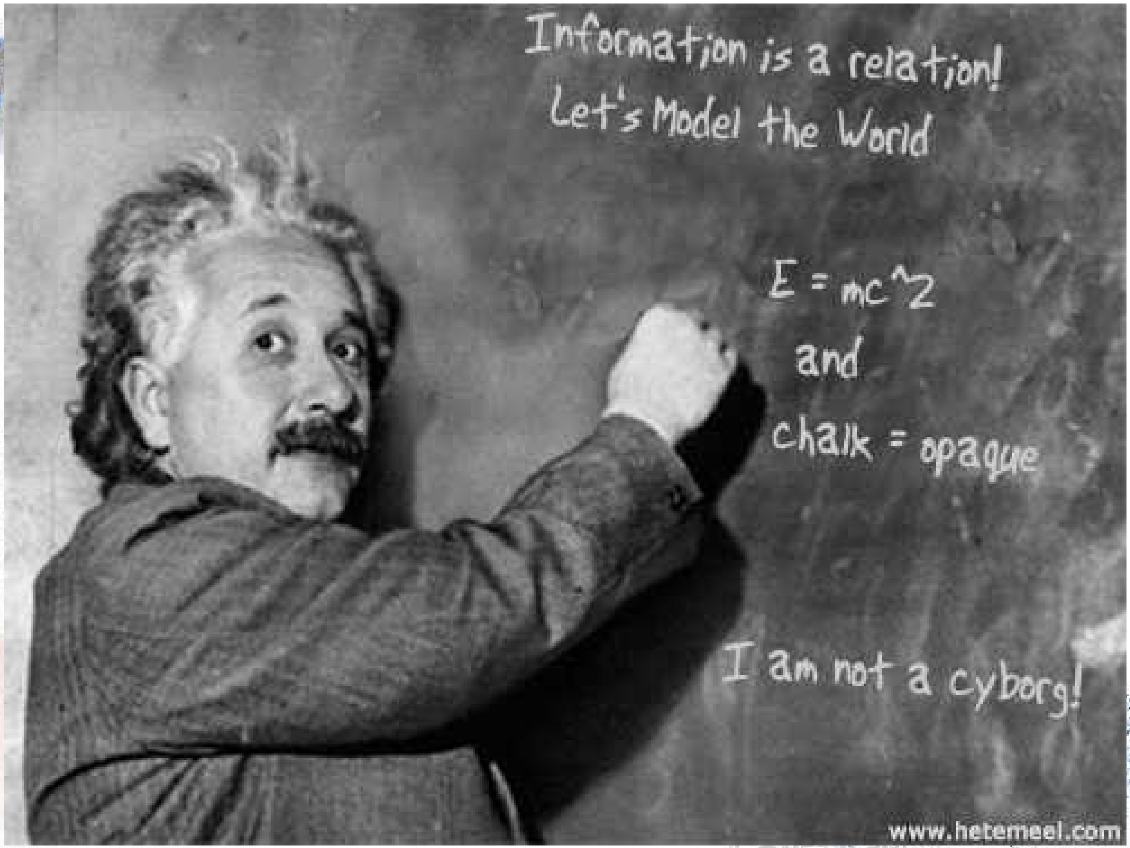
- XEROX PARC
  - Xerox Alto (1973)
    - first computer to use the *desktop metaphor* and GUI

Questions

- Transparent vs Opaque Technology?
- Describe two computing devices used before the XX century.
- What is a GUI?
- Which computer first featured the mouse and the desktop metaphor GUI?

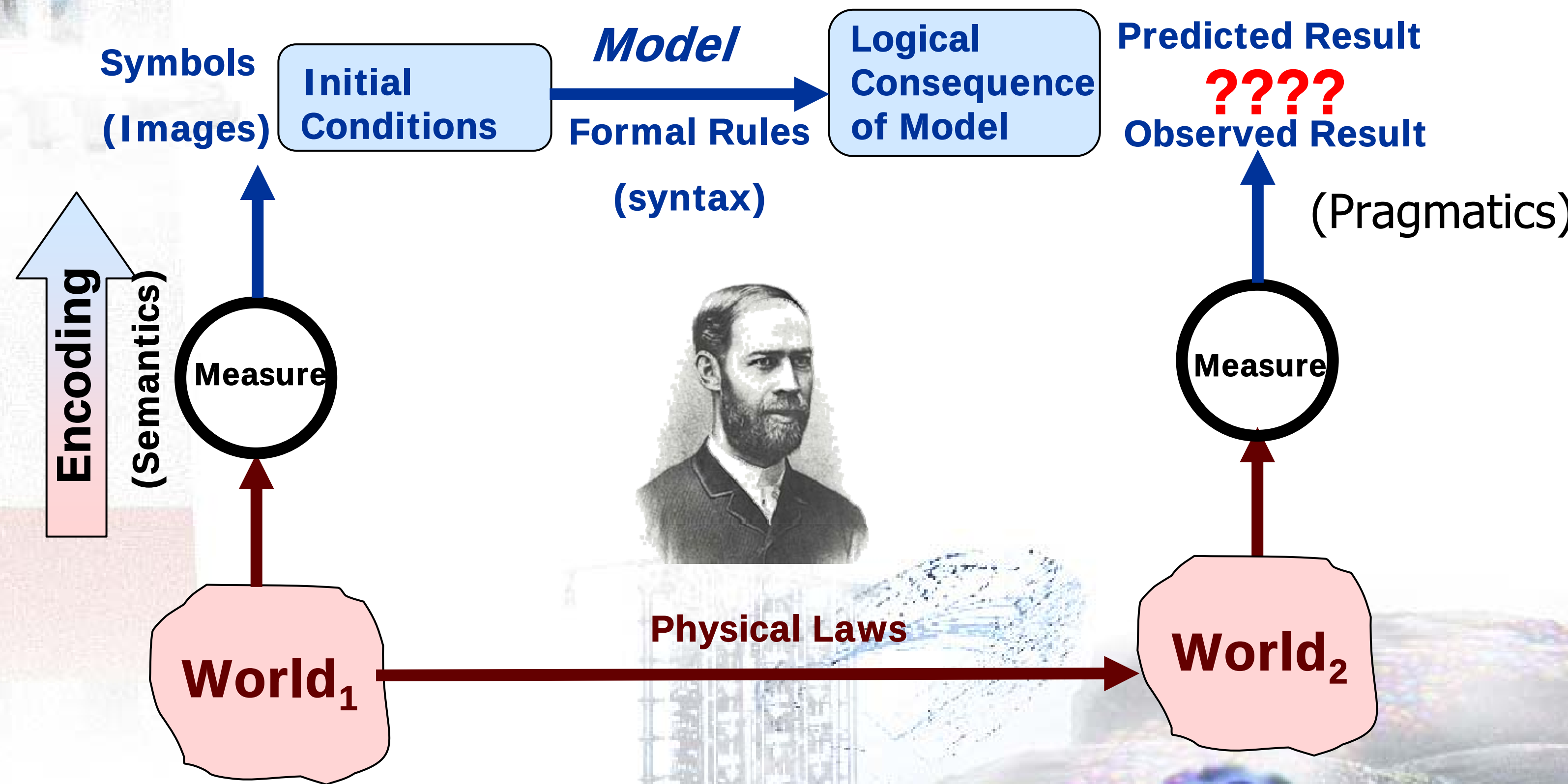
Modeling

Describing and Understanding the World



The Hertzian Modeling Paradigm

"The most direct and in a sense the most important problem which our conscious knowledge of nature should enable us to solve is the ***anticipation of future events***, so that we may arrange our present affairs in accordance with such anticipation". (Hertz, 1894)

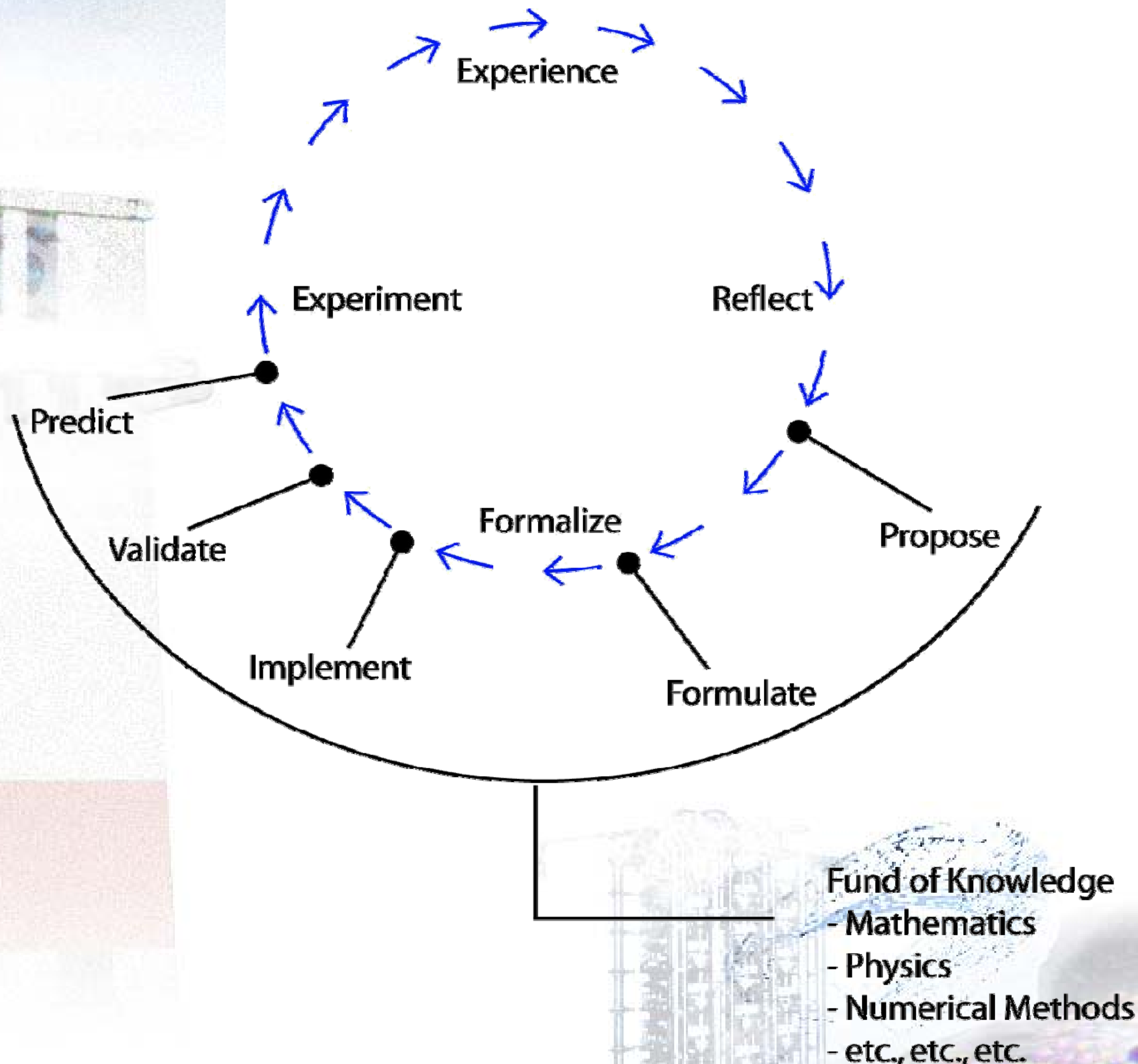


Polya Method: How To Solve It

1. Understanding The Problem
   - **First**. You have to understand the problem.
   - What is the thing you want to find to answer the problem (the unknown)?
   - Explain the question to other people
   - What are the data? What is the condition?
   - Draw a figure. Introduce suitable notation.
2. Devising A Plan (A ***Model***)
   - **Second**. Find the connection between the data and the unknown. You may need to consider auxiliary problems
   - Have you seen it before? Do you know a related or analogous problem?
   - Could you restate the problem? Could you solve a part of the problem?
   - Could you derive something useful from the data?
3. Carrying Out The Plan
   - **Third**. Calculate the model using all data and conditions.
   - Do all the calculations, and check them as they go along.
   - Ask: "Can I see it is right?" and then, "Can I prove it is right?"
4. Looking Back
   - **Fourth**. Examine the solution obtained.
   - Can you check the result?
   - Can you derive the solution differently?

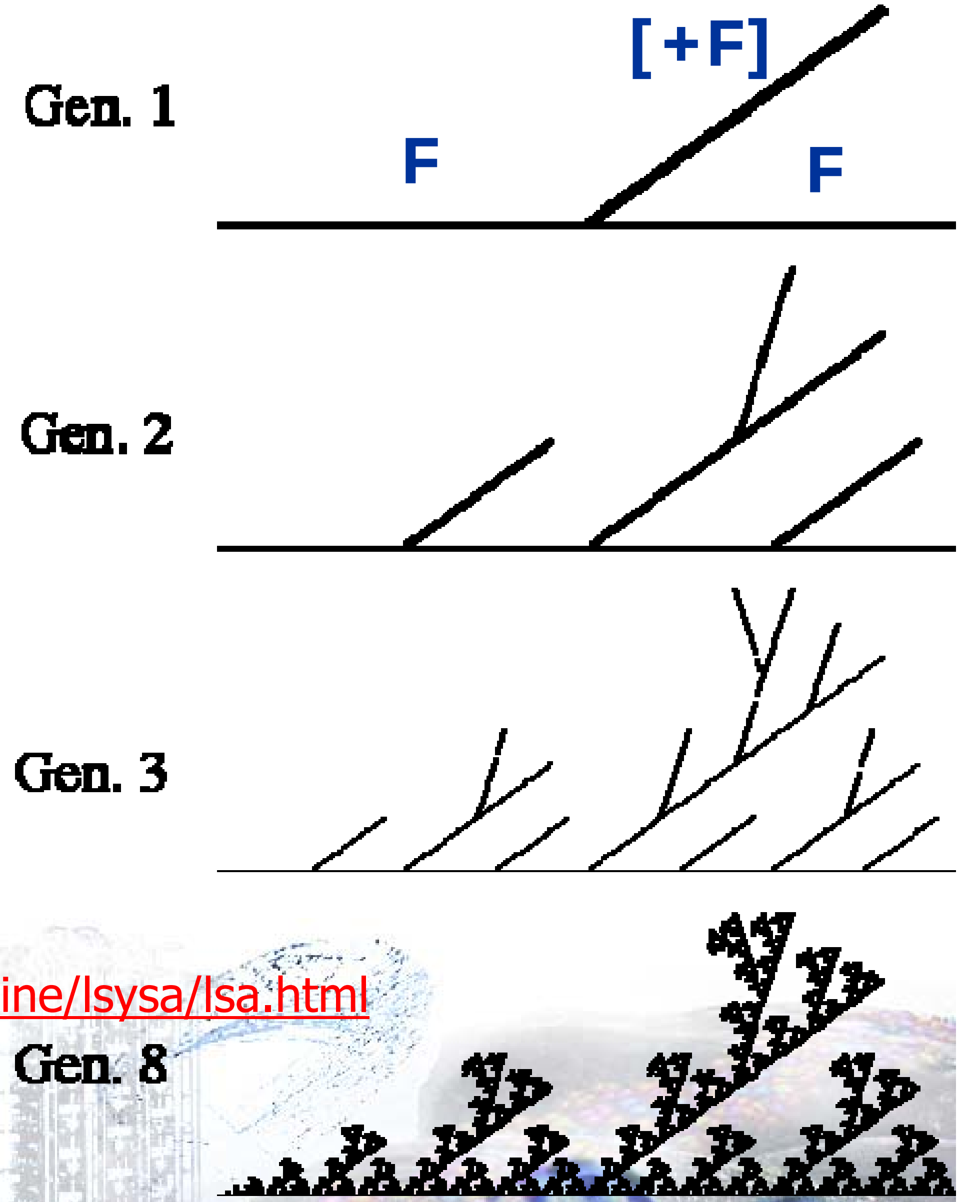
If you can't solve a problem, then there is an easier problem you can solve: find it.

The process of modeling



Branching L-Systems

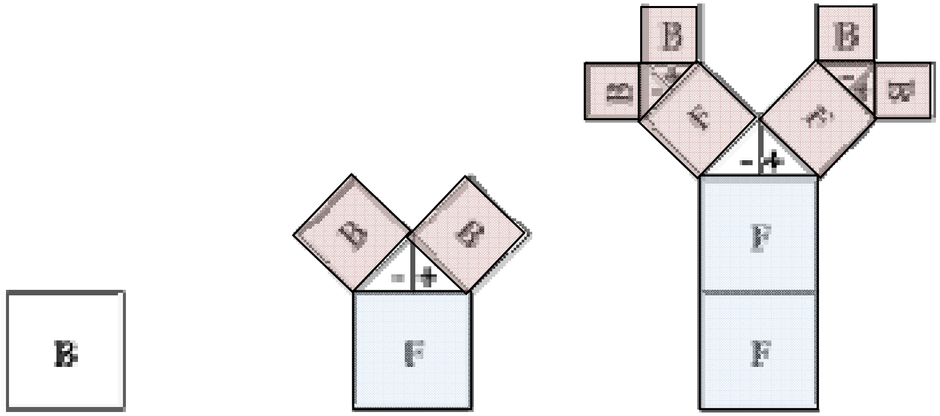
- Add branching symbols [ ]
  - simple example
    - Main trunk shoots off one side branch
  - Angle 45
  - Axiom F
  - F=F[+F]F



http://www.info.univ-angers.fr/pub/gh/vitrine/lsysa/lsa.html

L-system with 2 cell types

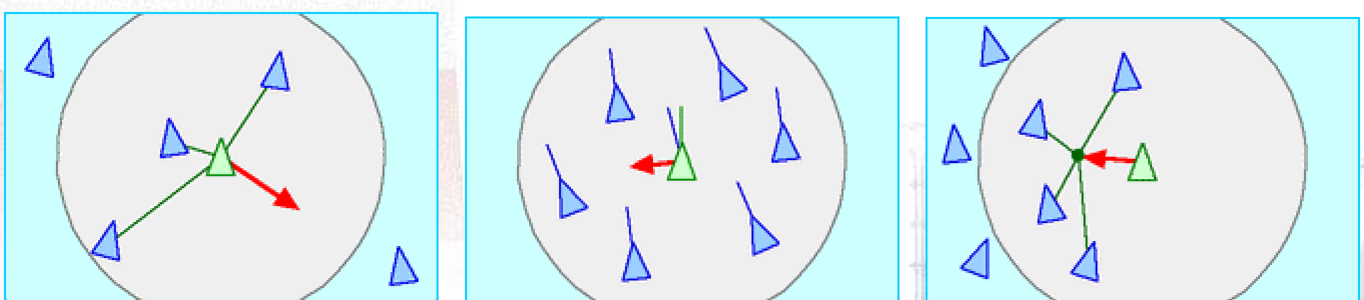
- Axiom
  - B
- Cell Types
  - B,F
- Rules
  - B → F[-B]+B
  - F → FF

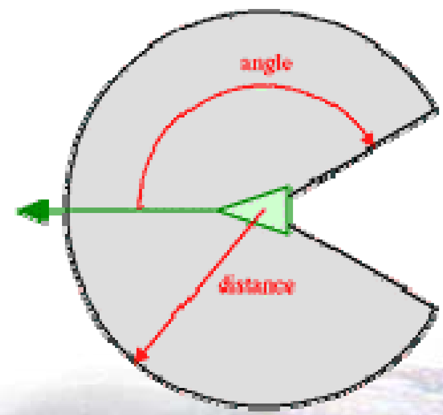


Depth	Resulting String
0	B
1	F[-B]+B
2	FF[-F[-B]+B]+ F[-B]+B
3	FFFF[-FF[- F[-B]+B]+ FF[-B]+B]+ F[- F[-B]+B]+ F[-B]+B

Flocking Behavior

- Boids by Craig Reynolds (1986)
  - 3 Steering behaviors
    - **Separation**: steer to avoid crowding local flockmates
      - Maintain minimum distance to others
    - **Alignment**: steer towards the average heading of local flockmates
      - Adjust speed according to others in vicinity
    - **Cohesion**: steer to move toward the average position of local flockmates
  - Each boid sees only flockmates within a certain small neighborhood around itself.
  - http://www.red3d.com/cwr/boids/

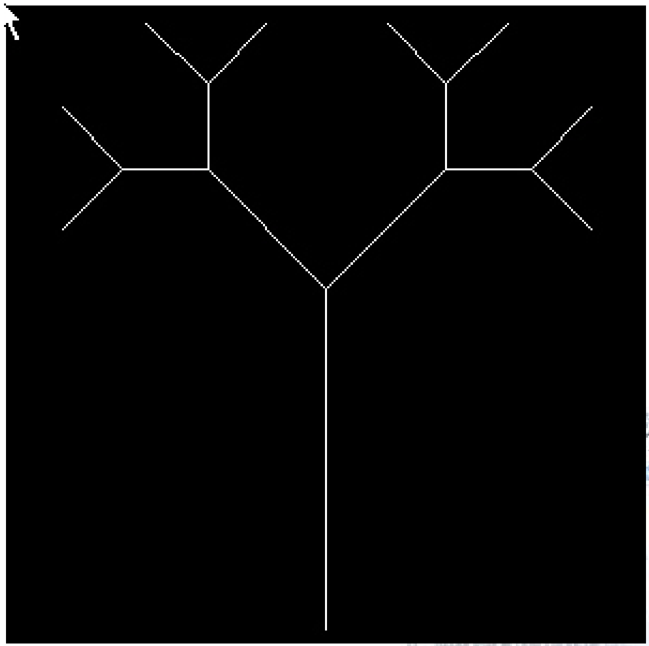




Luis M.Rocha and Santiago Schnell

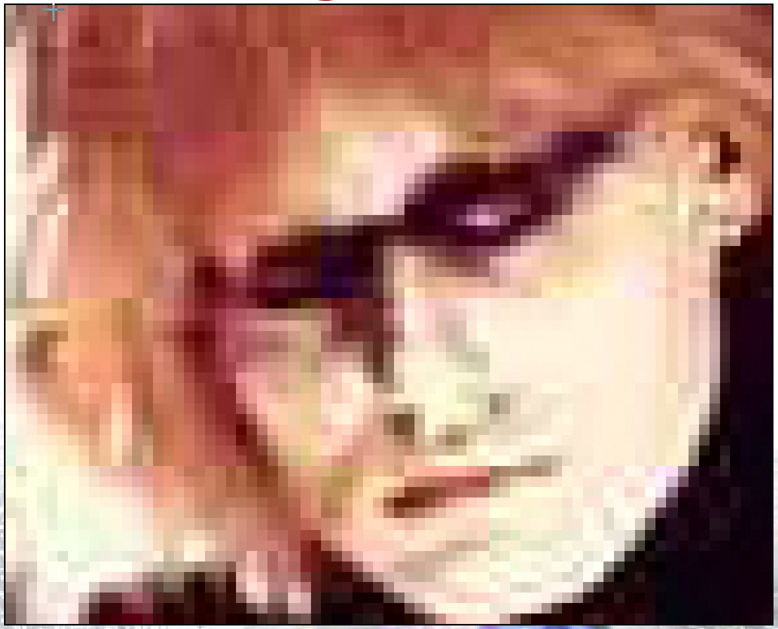
Possible Questions

1. Describe the Hertz Modelling Process
2. What are Boids and how do they work?
3. Propose an L-System Rule to draw the following artificial plant



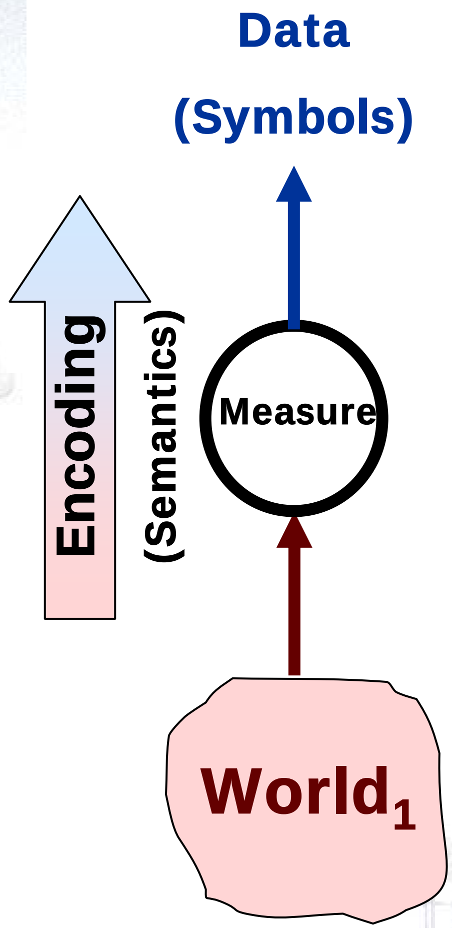
Data Representation

Encoding the World



Pixels: picture elements

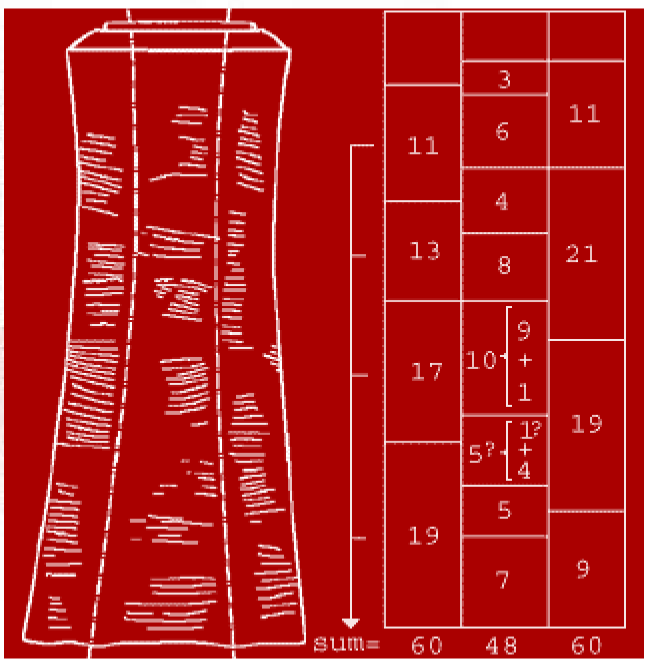
Encoding in the Modeling Relation

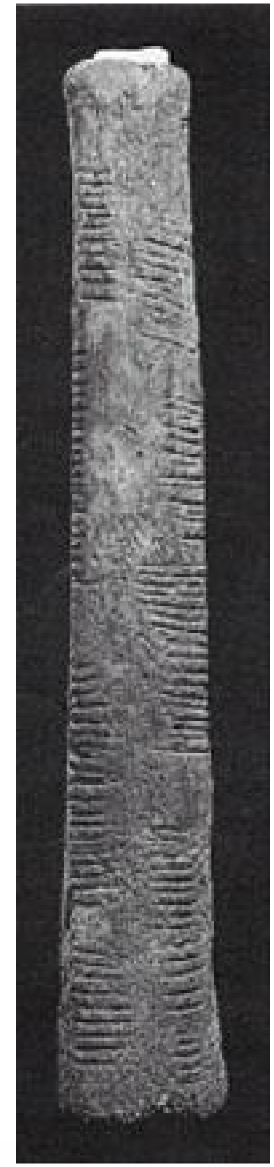


- How to encode data?
  - What is data?
    - Information without context and knowledge
    - Part of Syntax
  - Keeping Numbers
    - The most fundamental need for modeling and information

Encoding Numbers:Counting

- Tallying is the earliest form of modeling
  - Fingers (digits), stones (Lt "calculus"= Pebble), bones
  - ***Lebombo bone***
    - Oldest counting tool is a piece of baboon fibula with 29 notches from 35,000 BC, discovered in the mountains between South Africa and Swaziland
    - Probably representing the number of days in a Moon Cycle (A Model!)
  - Czechoslovakian wolf's bone
    - with 55 notches in groups of 5, from 30,000 BC.





The ***Ishango Bone***

- Oldest Mathematical Artefact?
  - 10,000 BC, border of Zaire and Uganda
  - Used as a counting tool?
    - 9,11,13,17,19, 21: odd numbers
    - 11, 13, 17, 19: prime numbers
    - 60 and 48 are multiples of 12

http://www.simonsingh.com/The_Ishango_Bone.html

Converting Binary to Decimal

- $2^8 = 256$
- $2^7 = 128$
- $2^6 = 64$
- $2^5 = 32$
- $2^4 = 16$
- $2^3 = 8$
- $2^2 = 4$
- $2^1 = 2$
- $2^0 = 1$

2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	1	0	0	1	0	0	1
	128	+ 64			+ 8			+ 1

201

$$\ldots d_4 d_3 d_2 d_1 d_0 =$$
$$\ldots + d_4 \times 2^4 + d_3 \times 2^3 + d_2 \times 2^2 + d_1 \times 2^1 + d_0 \times 2^0$$

Base Conversion

- Decimal to Binary
  - Repeated Division by 2
    - Divide the decimal number by 2
    - If the remainder is 0, on the side write down a 0
    - If the remainder is 1, write down a 1
    - Continue until the quotient is 0
    - Remainders are written beginning at the least significant digit (right) and each new digit is written to more significant digit (the left) of the previous digit.

decimal	quotient	Remain.	binary
58	29	0	0
29	14	1	10
14	7	0	010
7	3	1	1010
3	1	1	11010
1	0	1	**111010**

Dealing with rational numbers

- $2^4 = 16$
- $2^3 = 8$
- $2^2 = 4$
- $2^1 = 2$
- $2^0 = 1$
- $2^{-1} = 0.5$
- $2^{-2} = 0.25$
- $2^{-3} = 0.125$

2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}
1	1	0	0	1	0	0	1

16 + 8 + 1 + .125

25.125

$$\ldots d_2 d_1 d_0 . d_{-1} d_{-2} \ldots =$$
$$\ldots + d_2 \times 2^2 + d_1 \times 2^1 + d_0 \times 2^0 + d_{-1} \times 2^{-1} + d_{-2} \times 2^{-2} + \ldots$$

Binary Arithmetic

- Addition Rules
  - 0+0 = 0, with no carry,
  - 1+0 = 1, with no carry,
  - 0+1 = 1, with no carry,
  - 1+1 = 0, and you carry a 1

```
111__
 1010
 1110
-------
11000
```

(10+14=24)

```
1____
 1010
 1100
-------
10110
```

(10+12=22)

Binary Multiplication

```
    1010
    1110
 -------
 111____
    0000
   1010_
  1010__
 1010___
-----------
10001100
```

(10+14=140)

```
    1010
    1100
 -------
    0000
   0000_
  1010__
 1010___
-----------
 1111000
```

(10×12=120)

Luis M.Rocha and Santiago Schnell

Hexadecimal

- Base 16
  - 16 symbols: 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F
  - Easy to convert to and from Binary
    - 16 is a power of 2: $16 = 2^4$
    - It takes 4 binary digits for every hexadecimal one
    - Good to represent binary in compressed form!

Hex	Bin	Hex	Bin	Hex	Bin	Hex	Bin
0	0000	4	0100	8	1000	C	1100
1	0001	5	0101	9	1001	D	1101
2	0010	6	0110	A	1010	E	1110
3	0011	7	0111	B	1011	F	1111

Encoding Text

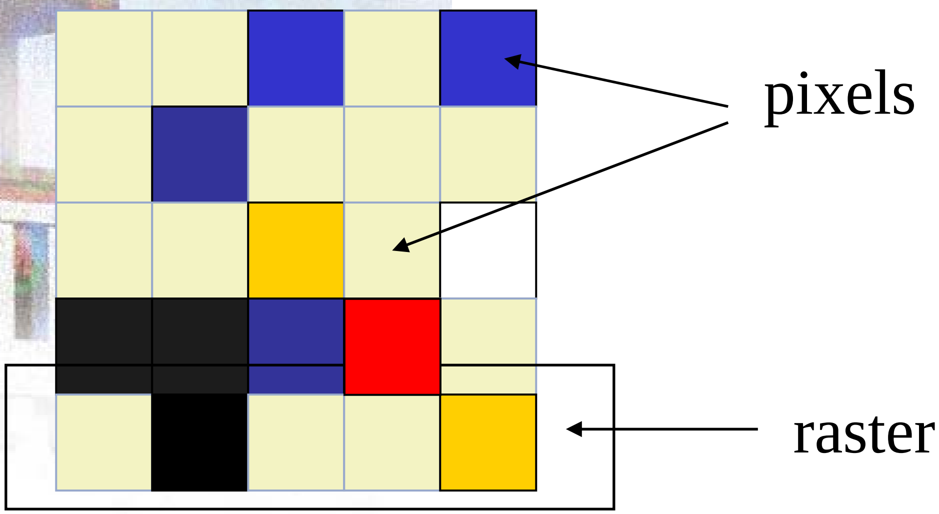
- ASCII
  - American Standard Code for Information Interchange
    - between binary numbers and computer and roman symbols
    - Standard to allow computers to communicate textual data
  - Uses 7 bits to encode 128 symbols or characters
    - $2^7 = 128$. It fills a byte, but the 8th bit is used to encode additional symbols for other languages and graphics
    - Usually described in hexadecimal
  - 4 groups of 32 characters
    - 00 to 1F: **control characters**
      - Mostly printer/display operations: *carriage return* (0Dh), *line feed* (0Ah), *back space* (08h), etc.
    - 20 to 3F: punctuation, numeric, and special characters
      - Space (20h), digits 0-9 (30h-39h)
        - Arranged so that by subtracting 30h from the ASCII code for any digit, we obtain the numeric equivalent of the digit
    - 40 to 5F: uppercase letters, plus some special characters
    - 60 to 7F: lowercase letters, plus some special characters and a control character (DEL)

ASCII Table

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH	(start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX	(start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX	(end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT	(end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ	(enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK	(acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL	(bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS	(backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB	(horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT	(vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR	(carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO	(shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI	(shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE	(data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1	(device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2	(device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3	(device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4	(device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK	(negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN	(synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB	(end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN	(cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM	(end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB	(substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC	(escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS	(file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS	(group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS	(record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US	(unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.asciitable.com

Bitmap Image



- Representation of a two-dimensional image as a finite set of digital values
- Picture elements or ***pixels***
  - **Resolution:** number of pixels in an image
    - 1024 x 768
  - Each defined by one or more numbers
    - Color, intensity

2^{24} (about 16 million)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	0	0	1

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	0	0	1

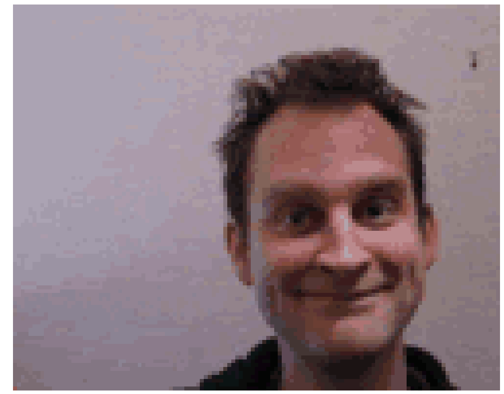
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	0	0	1

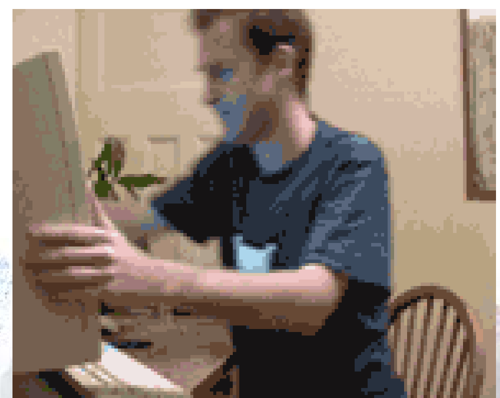
RGB: 3 Bytes, one for each primary colors: Red, Green, and Blue: *16,777,216* colors total

Adapted from Cathy Wyss (I308)

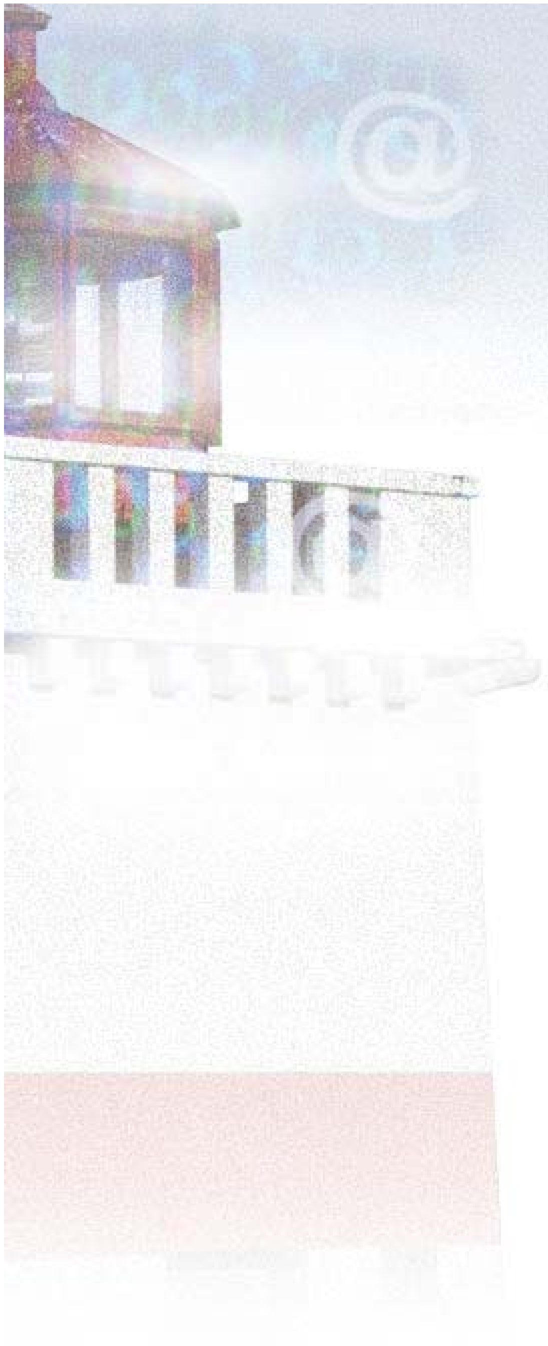
Graphics Interchange Format

- Developed by CompuServe in 1987 (GIF87a)
  - Developed to facilitate exchange across computing platforms
  - Allows transparency
  - GIF89a
    - 1989: allows animated GIF images
- Uses LZW (Lempel-Ziv-Welch) data compression
  - More efficient than plain bitmaps
  - Large images downloaded quicker
  - Lossless compression
  - 256 colors only
  - Patent owned by Unisys until 2003
    - Compuserve did not know that LZW was covered by a patent.
    - Before 1994, Unisys was not aware that GIF used LZW.
  - Builds a dictionary of previously seen strings in the information being compressed.
    - The dictionary does not have to be transmitted





http://sheepfilms.co.uk




Luis M.Rocha and Santiago Schnell

Questions

- What is a positional number system? Give an example of a number system that is not positional, and an example of one that is positional.
- Convert 1001001101.01 from binary to decimal. Please show your calculations.
- What is the ASCII encoding of the word PLATO (Uppercase) in Decimal?
  - 84 85 82 73 78
  - 73 78 70 79 82 77 65 84 73
  - 65 82 73 83 84 79 84 76
  - 80 76 65 84 79
- How many bytes do you need to encode a bitmap figure with resolution 300 x 600 using the RGB format?
  - 960,000
  - 180,000
  - 480,000
  - 540,000

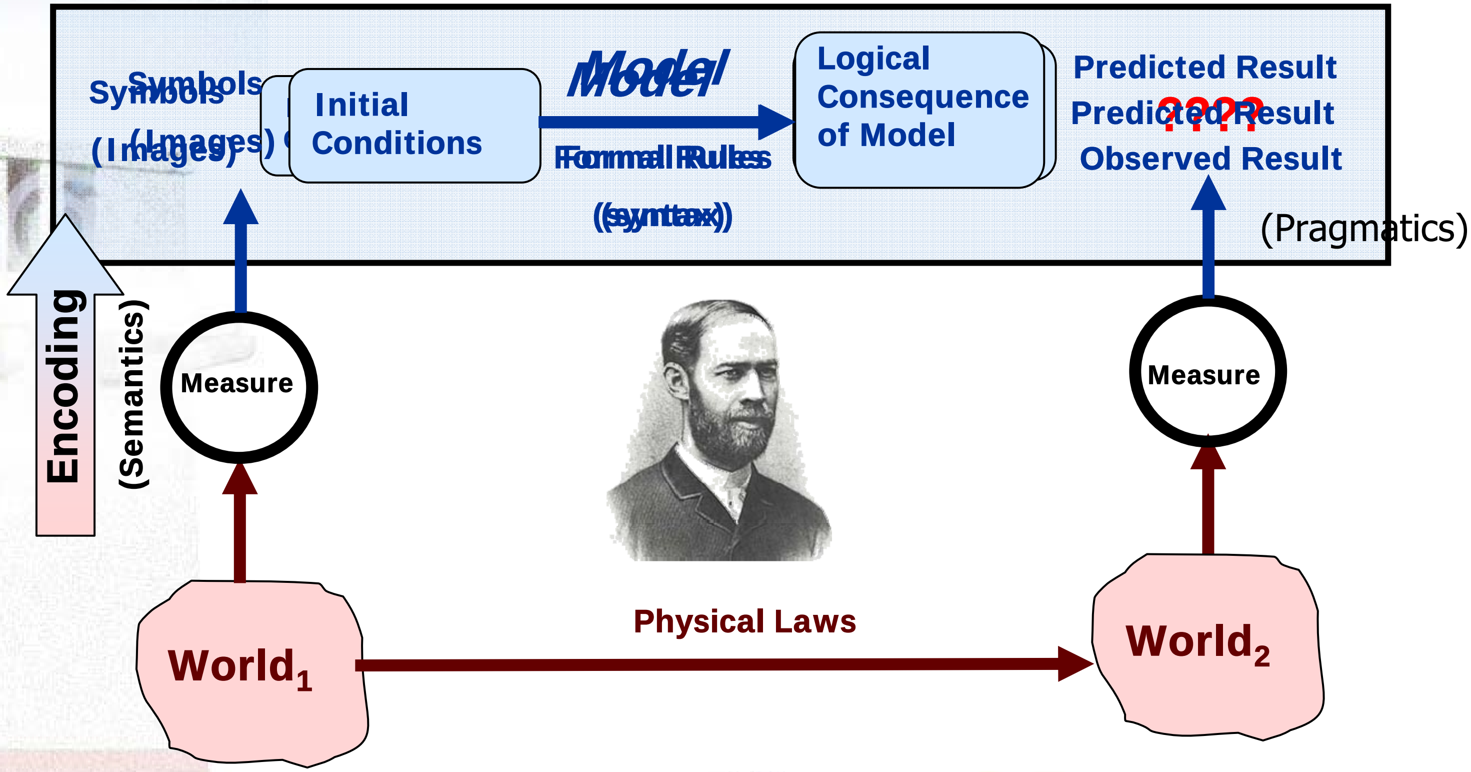
Deductive Modeling

Logic, Sets, Deduction

Logic: another thing that penguins aren't very good at.

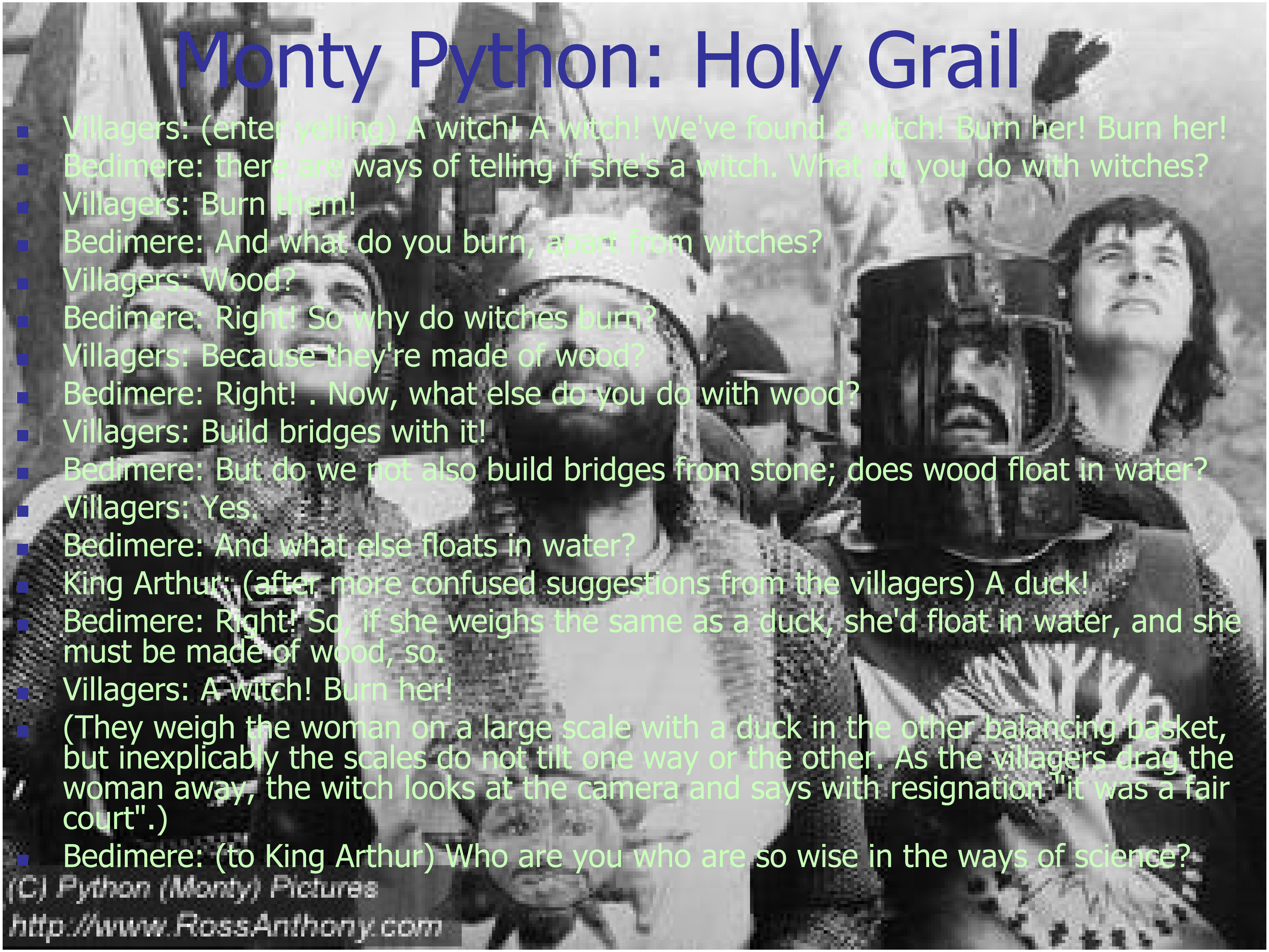
The Modeling Relation

Hertz' Modeling Paradigm



- Formal Rules
  - From symbolic representations of observables
  - Produce Conclusions

Monty Python: Holy Grail



- Villagers: (enter yelling) A witch! A witch! We've found a witch! Burn her! Burn her!
- Bedimere: there are ways of telling if she's a witch. What do you do with witches?
- Villagers: Burn them!
- Bedimere: And what do you burn, apart from witches?
- Villagers: Wood?
- Bedimere: Right! So why do witches burn?
- Villagers: Because they're made of wood?
- Bedimere: Right! . Now, what else do you do with wood?
- Villagers: Build bridges with it!
- Bedimere: But do we not also build bridges from stone; does wood float in water?
- Villagers: Yes.
- Bedimere: And what else floats in water?
- King Arthur: (after more confused suggestions from the villagers) A duck!
- Bedimere: Right! So, if she weighs the same as a duck, she'd float in water, and she must be made of wood, so.
- Villagers: A witch! Burn her!
- (They weigh the woman on a large scale with a duck in the other balancing basket, but inexplicably the scales do not tilt one way or the other. As the villagers drag the woman away, the witch looks at the camera and says with resignation "it was a fair court".)
- Bedimere: (to King Arthur) Who are you who are so wise in the ways of science?

Deduction vs. Induction

- Propositional Logic is used to study ***inferences***
  - Lists of propositions
  - How conclusions can be reached from premises
- Deductive Inference ← **Logic**
  - If the premises are true, we have absolute ***certainty*** of the conclusion
    - February has 29 days only in leap years
    - Today is February 29th
    - This year is a leap year
- Inductive Inference ← **Uncertainty**
  - Conclusion supported by ***good evidence*** (significant number of examples/observations) but not full certainty -- ***likelihood***
    - Ran WhiteBox for 1000 cycles, "dead box" observed
    - Ran WhiteBox for 1000 cycles, "dead box" observed
    - Ran WhiteBox for 1000 cycles, "dead box" observed
    - ……………………………………………
    - Ran WhiteBox for 1000 cycles, "dead box" observed
    - "Dead Box" always appears after 1000 cycles

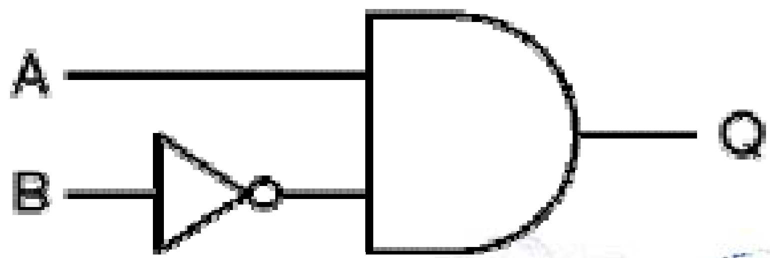
The structure of propositional logic

- *Simple propositions* are represented by single, lower case letters
  - Bloomington is a town – *p*
  - Indiana is a state - *q*
- *Complex propositions* are constructed by applying logical operations to simple propositions
  - Bloomington is a town and Indiana is a state – *p* and *q*
- *Logic Operations*

Conjunction	[and]	$\wedge$
Disjunction	[or]	$\vee$
Negation	[not]	$\neg$
Conditional	[implies]	$\Rightarrow$ (if, then)
Biconditional	[equivalent]	$\Leftrightarrow$ (if and only if)

Questions

- Build the truth tables of the following:
  - $\neg\neg a$
  - $a \wedge (b \wedge \neg a)$
  - $(a \Rightarrow b) \Rightarrow \neg b$
- What is the logic expression for the following diagram:



  - Build its truth table