Computer Science 50

Introduction to Computer Science I

Harvard College

Week 0

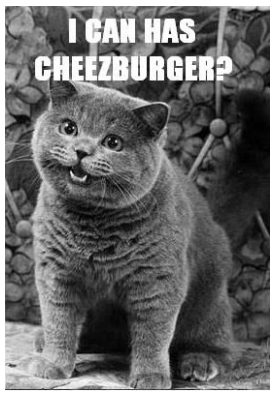
David J. Malan
malan@post.harvard.edu

Roll Call

1) Stand up.
2) Think to yourself “I am #1”.
3) Pair up with someone; add your numbers together; take that sum as your new number.
4) One of you should sit down.
5) GOTO step 3 if still standing.

Divide and Conquer

Lolcats



Who

I'm among "those less comfortable"		30.1%
I'm among "those more comfortable"		20.3%
I'm somewhere in between		49.6%

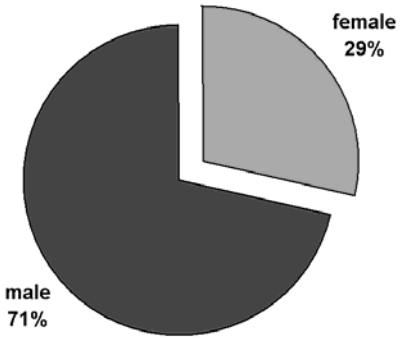
"Those Less Comfortable"

Image from stevens.senate.gov.

Prior Programming Experience

none		43.6%
a little		46.3%
a lot		10.1%

Gender



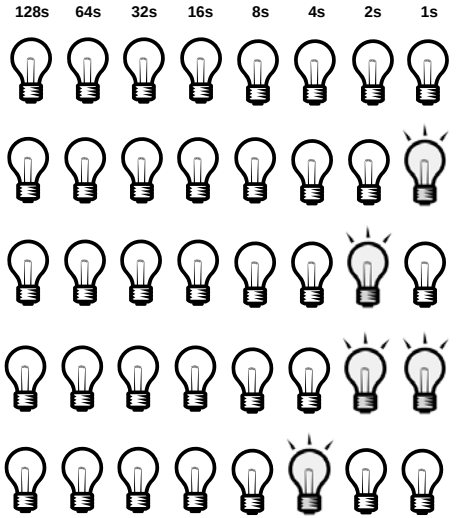
Should I?

:: Today (9/15), 3:30 – 5, Emerson 105
:: Wednesday (9/17), 3:30 – 5, Emerson 105
:: Thursday (9/18), 3:30 – 5, Emerson 105

0

1

Counting in Binary



Counting in Binary

128s	64s	32s	16s	8s	4s	2s	1s
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0

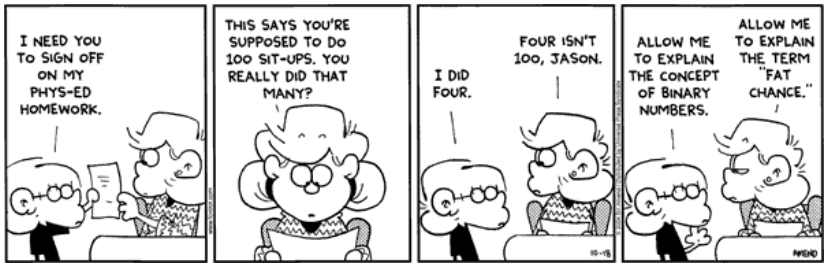
Cambridge in Binary

128s	64s	32s	16s	8s	4s	2s	1s
_	_	_	_	_	_	_	_
_	_	_	_	_	_	_	_
_	_	_	_	_	_	_	_
_	_	_	_	_	_	_	_
_	_	_	_	_	_	_	_

Cambridge in Binary

128s	64s	32s	16s	8s	4s	2s	1s
0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0

FoxTrot in Binary



ASCII

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

This is CS 50.

Introduction to the intellectual enterprises of computer sciences. Algorithms: design, implementation, analysis. Software development: abstraction, encapsulations, data structures, debugging, testing. Architecture of computers: low-level data representation, instructions processing. Computer Systems: programming languages, compilers, operating systems, databases. Computers in the real world: networks, websites, security, forensics, cryptography. This course teaches students how to think more carefully and how to solve problems more effectively. Problem sets involve extensive programming in C as well as PHP and JavaScript.

No previous programming experience required.

This course, when taken for a letter grade, meets the Core area requirement for Quantitative Reasoning.

Expectations*

:: Attend all lectures and sections
:: Complete nine problem sets
:: Take two quizzes
:: Produce a final project

* No final exam.

Grades*

:: Problem Sets (best 8 out of 9) 60%
:: Quizzes 30%
:: Final Project 10%

* You may take the course pass/fail.

Website

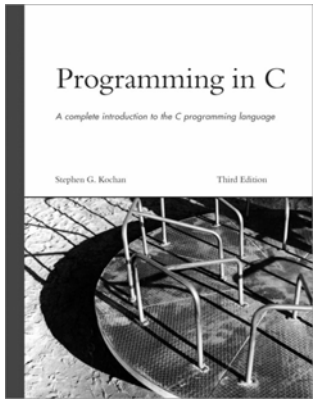
http://cs50.net/

Recommended

For Those Less Comfortable

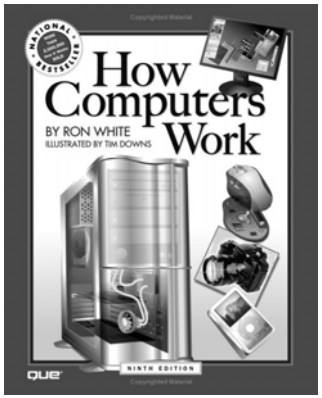
Recommended

For Those More Comfortable



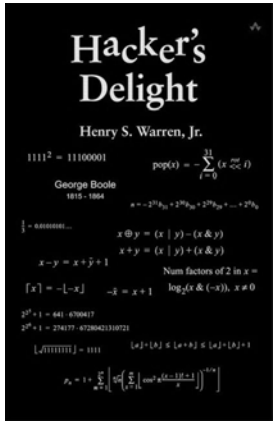
Recommended

For Everyone



Recommended

For Aspiring Hackers



Lectures*

* “Lunch with David” on Fridays (rsvp@cs50.net)

Sections

:: For “those less comfortable”
:: For “those more comfortable”
:: For those somewhere in between

Staff

:: Teaching Fellows
:: Course Assistants
:: Virtual Teaching Fellows
:: Sysadmins
:: Scribes
:: Producers, Videographers, AV
:: me

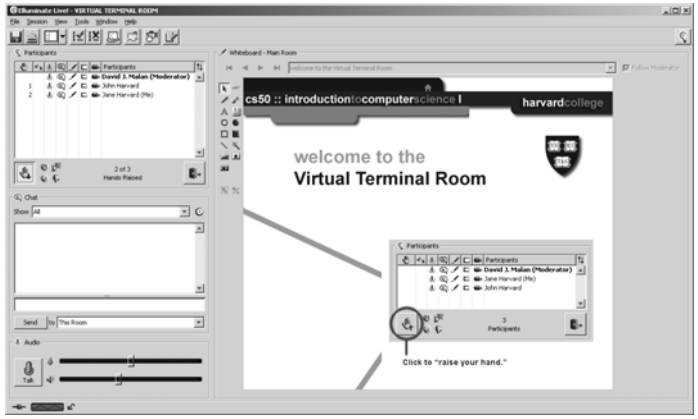
Head TF

Cansu Aydede '11
aydede@fas.harvard.edu
+1-617-832-5239

Applied Math, Computer Science
Eliot House

Office Hours

Virtual Office Hours



Workload

Lectures

Week 0

Introduction. Bits. Binary. ASCII. Programming. Algorithms. Scratch. Statements. Boolean expressions. Conditions. Loops. Variables. Threads. Events.

Lectures

Week 1

C. Source code. Compilers. Object code. SSH. SFTP. GCC. Functions. Comments. Standard output. Arithmetic operators. Precedence. Associativity. Local variables. Types. Casting. Standard input. Libraries. Boolean expressions, continued. Conditions, continued. Loops, continued.

```
printf("hai, world\n");
```


Lectures

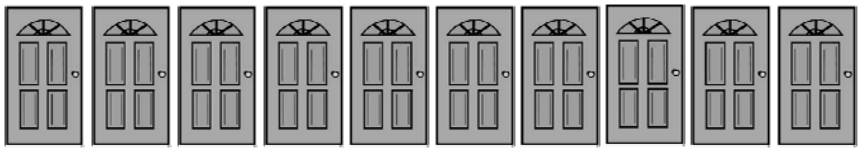
Week 2

Functions, continued. Global variables. Parameters. Return Values. Stack. Frames. Scope. Arrays. Strings. Command-line arguments. Cryptography.

Lectures

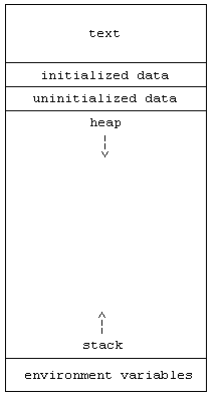
Week 3

Linear search. Binary search. Asymptotic notation. Recursion. Pseudorandomness. Bubble sort. Selection sort. Insertion sort. Merge sort.



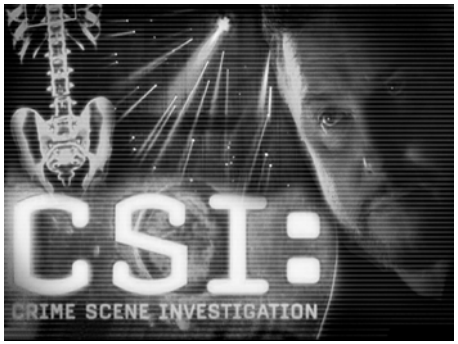
Lectures
Week 4

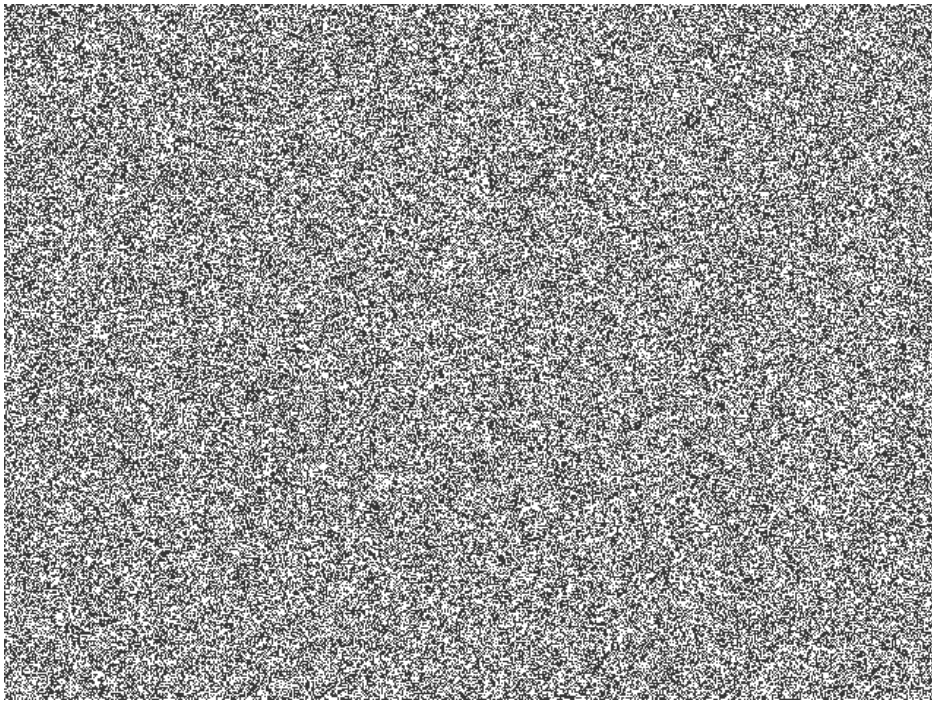
Structures. Dynamic memory allocation. Pointers.



Lectures
Week 5

Pointers, continued. Heap. Debugging. File I/O.
Forensics.





Lectures
Week 6

Linked lists.

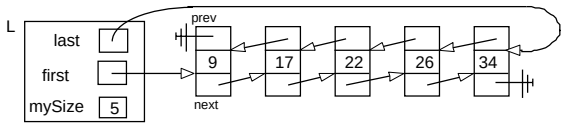


Figure from http://cs.calvin.edu/books/c++/ds/1e/.

Lectures
Week 7

Hash tables. Binary search trees. Huffman coding. Heaps. Heapsort. Tries.

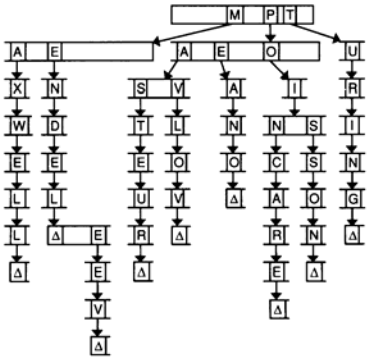


Image from *Data Structures & Their Algorithms.*

Lectures
Week 8

TCP/IP. HTTP. XHTML. PHP. SQL.

Lectures
Week 9

DOM. CSS. Inheritance. JavaScript. Events, continued. OOP. Ajax.

Image from www.gearthblog.com.

Lectures
Week 10

Preprocessing. Compiling. Assembling. Linking. CPUs.

Image from regmedia.co.uk.

Lectures

Week 11

Enterprise architectures. Virtualization. Cloud computing. Sneak previews.

Image from amazon.com.

Lectures

Week 12

Exciting conclusion.

Problem Sets*

- Problem Set 0: Scratch
- Problem Set 1: C
- Problem Set 2: Crypto
- Problem Set 3: To Be Named
- Problem Set 4: To Be Named
- Problem Set 5: Forensics
- Problem Set 6: Mispellings
- Problem Set 7: XHTML + PHP + SQL
- Problem Set 8: JavaScript

* Hacker Editions too

Final Project*

- Build something of interest to you.
- Make something useful.
- Solve an actual problem.
- Somehow impact campus.

* CS 50 Fair

kthxbai

Algorithms

```
1)  let socks_on_feet = 0
2)  while socks_on_feet != 2
3)          open sock drawer
4)          look for sock
5)          if you find a sock then
6)                  put on sock
7)                  socks_on_feet++
8)                  look for matching sock
9)                  if you find a matching sock then
10)                         put on matching sock
11)                         socks_on_feet++
12)                         close sock drawer
13)                 else
14)                         remove first sock from foot
15)                         socks_on_feet--
16)         else
17)                 do laundry and replenish sock drawer
```


O Hai, C!

hai.c

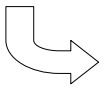
```
#include <stdio.h>

int
main(int argc, char * argv[])
{
    printf("O hai, world!\n");
}
```


O Hai, C!

```
#include <stdio.h>

int
main(int argc, char * argv[])
{
    printf("O hai, world!\n");
}
```



```
10000011 00000001 00010001 00000000 00111101 11111100 01110100 00111101
00000000 01000000 00000000 00000000 00000000 00000000 00000000 00000000
10010000 00000000 00000000 00000000 01010000 00000000 00000111 00110000
00001011 00000001 00001011 00000011 00001010 00000000 00000000 00000000
00000000 00100000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00100000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
01110000 00010000 00000000 00100000 00000001 00000000 00000000 00000000
00000000 00000000 00000000 00100000 00000001 00000000 00000000 00000000
00000000 00000000 00000000 01000000 00000001 00000000 00000000 00000000
00000000 00100000 00000000 01000000 00000001 00000000 00000000 00000000
11111111 11111111 11111111 11111111 11111111 11111111 11111111 11111111
10010000 10000000 00000000 01000000 00000001 00000000 00000000 00000000
00101110 01100100 01111001 01101110 01100001 01101101 01101001 01100011
10110000 00000100 00000000 00100000 00000001 00000000 00000000 00000000
10110000 00000100 00000000 00100000 00000001 00000000 00000000 00000000
10100000 00000001 00000000 00000000 00000000 00000000 00000000 00000000
10110000 00000100 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00100000 00000000 00000000
[...]
```

O Hai, Scratch!

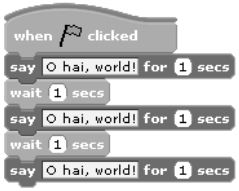
Hai1.sb

Statements

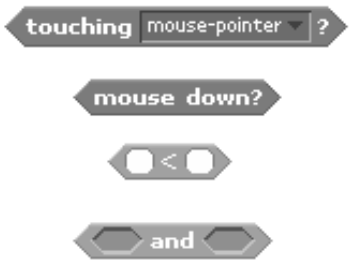
• • •

Statements

Hai{2,3}.sb

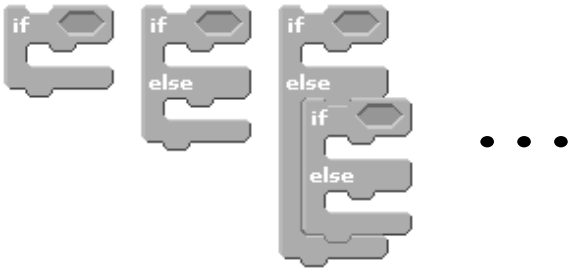


Boolean Expressions



• • •

Conditions



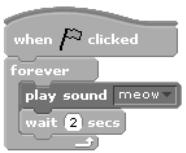
Conditions

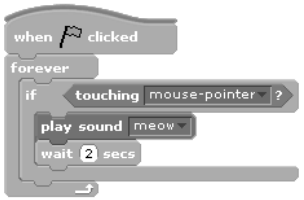
Hai{4,5}.sb

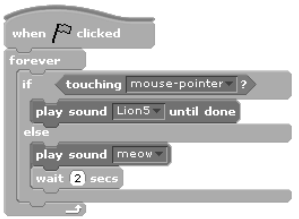
Loops

Loops

Hai{6,7,8}.sb

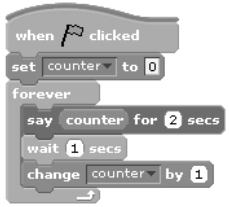




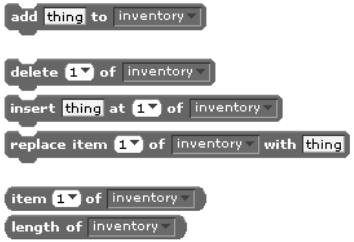


Variables

Count{1,2}.sb



Arrays



Arrays

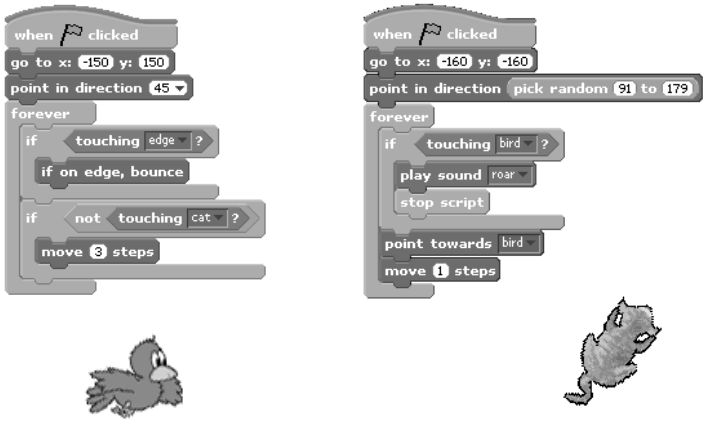
FruitcraftRPG.sb

Threads

Move1.sb

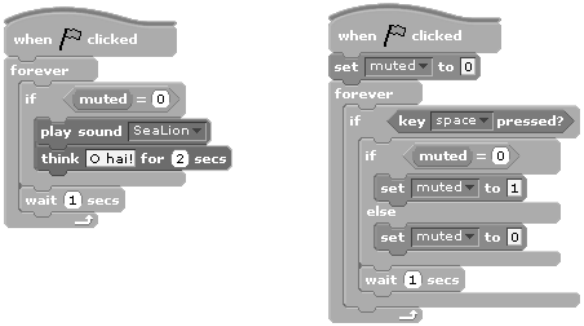
Threads

Move2.sb



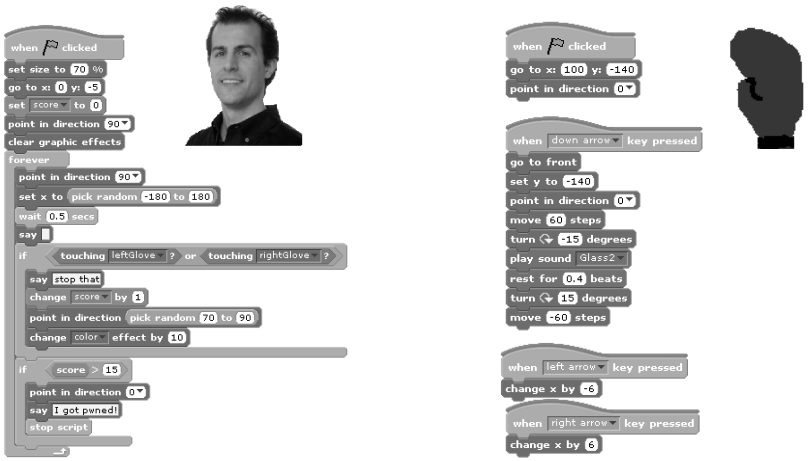
Threads

Hai10.sb



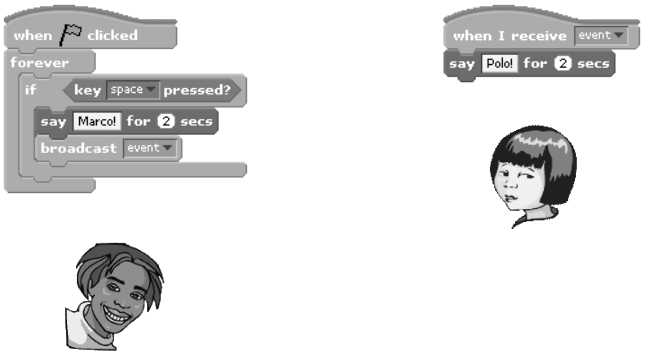
Threads

David.sb



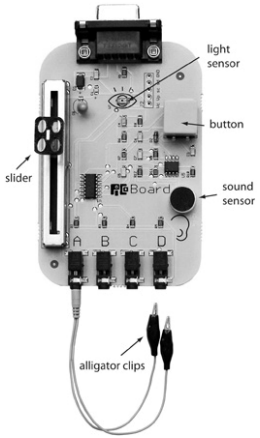
Events

Marco.sb



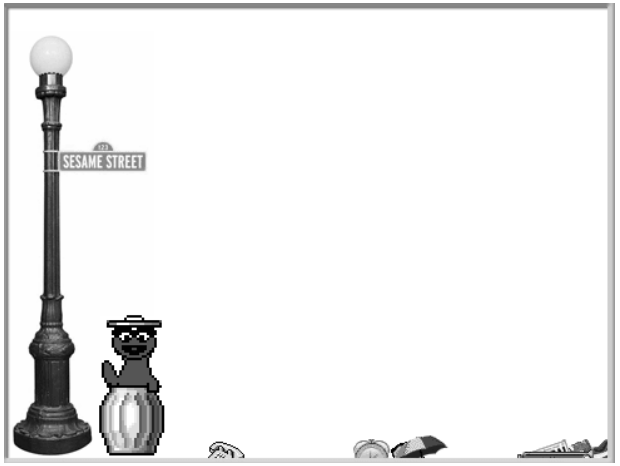
Sensors

singer.sb, Masquerade.sb, davidwu.sb, Electricity.sb



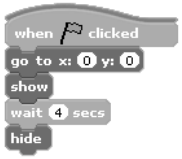
Oscartime

Oscartime.sb



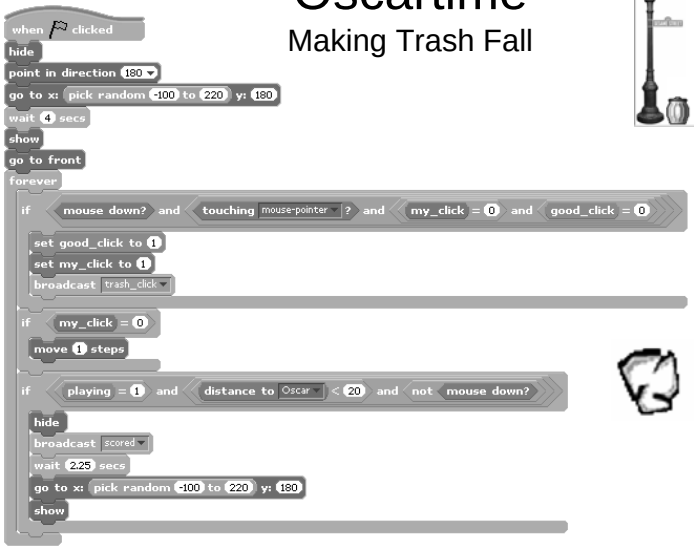
Oscartime

Displaying the Instructions



Oscartime

Making Trash Fall



Oscartime

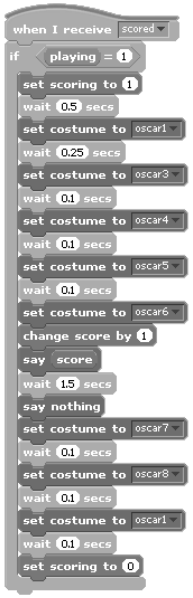
Implementing Dragging

Oscartime

Imposing a Time Limit

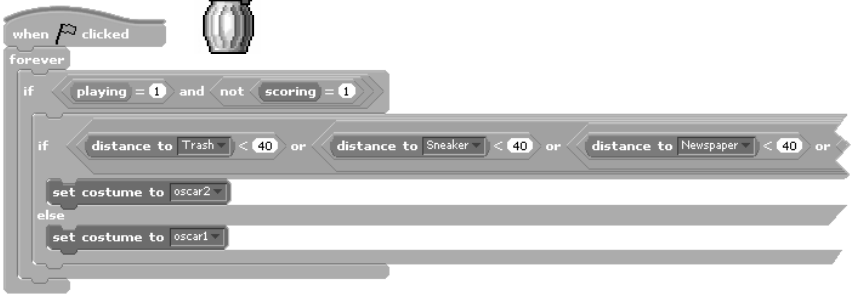
Oscartime

Keeping Score



Oscartime

Raising Oscar's Lid



Scratch Meets C


```
int
main(int argc, char * argv[])
{
    printf("O hai, world!\n");
}
```


Statements

Scratch v. C


```
printf("O hai, world!\n");
```

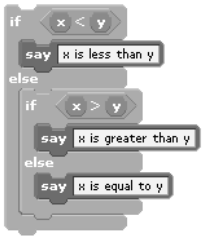

Boolean Expressions

Scratch v. C


```
(x < y)
((x < y) && (y < z))
```


Conditions

Scratch v. C




```
if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
 printf("x is equal to y\n");
}
```

Loops

Scratch v. C


```
while (1)
{
    printf("O hai!\n");
}
```


```
for (int i = 0; i < 10; i++)
{
    printf("O hai!\n");
}
```


Variables

Scratch v. C

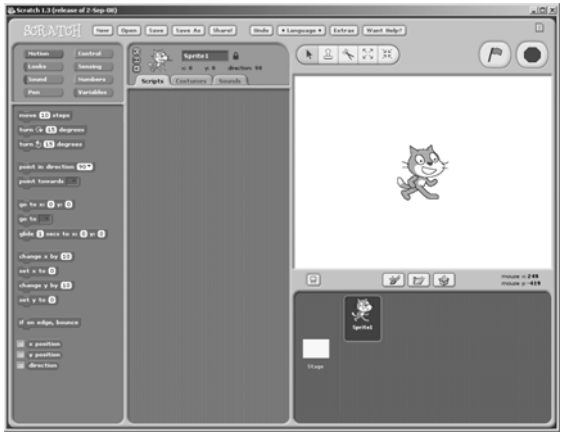

```
int counter = 0;
while (1)
{
    printf("%d\n", counter);
    counter++;
}
```


Arrays

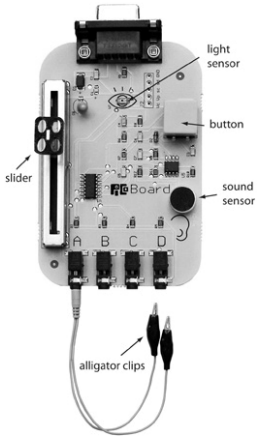
Scratch v. C


```
char *inventory[SIZE];
inventory[i] = "Orange";
```


Problem Set 0



Problem Set 0
HACKER EDITION



Computer Science 50

Introduction to Computer Science I

Harvard College

Week 0

David J. Malan
malan@post.harvard.edu