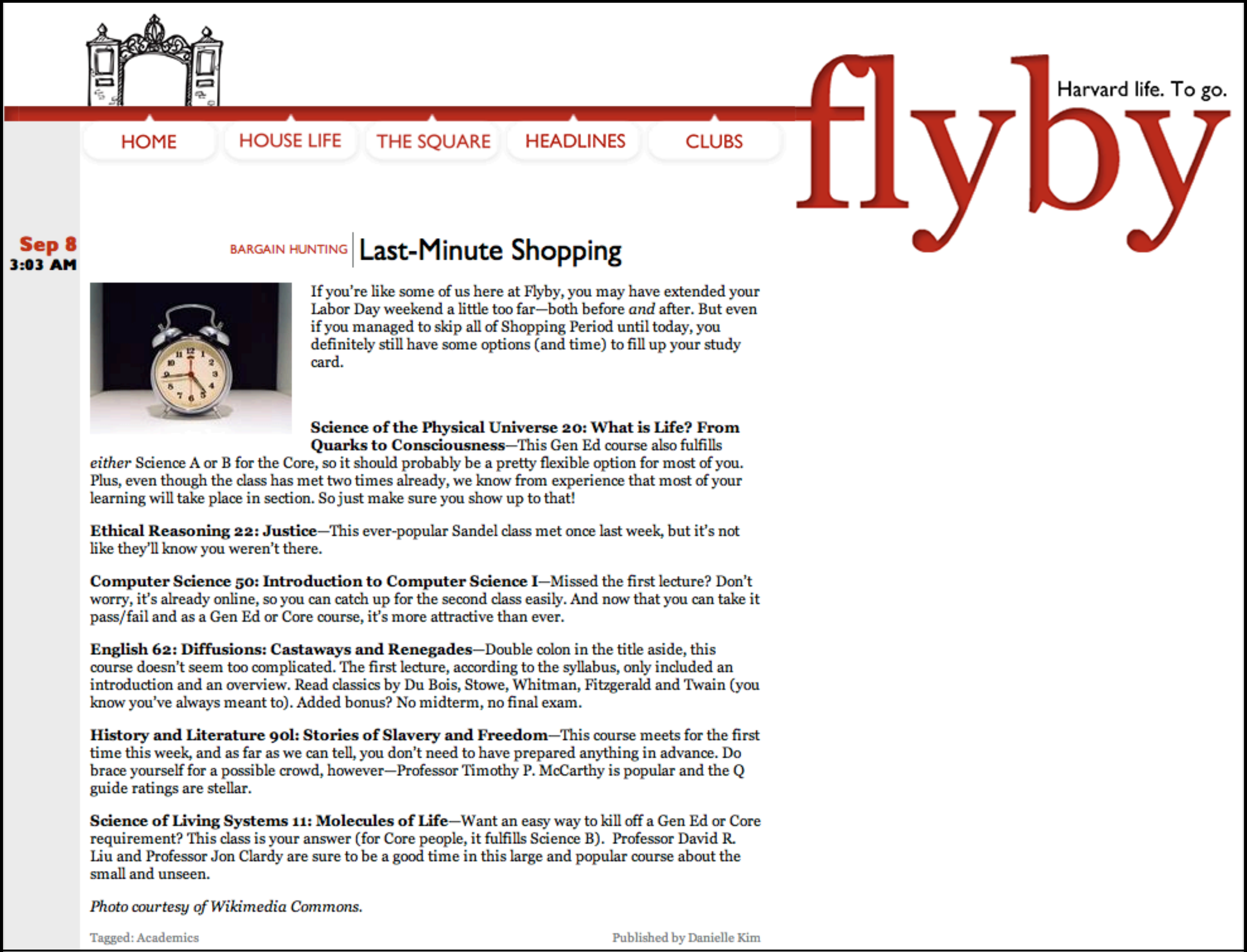
A Cookie Love Story



Sep 8
3:03 AM

BARGAIN HUNTING | Last-Minute Shopping

If you're like some of us here at Flyby, you may have extended your Labor Day weekend a little too far—both before *and* after. But even if you managed to skip all of Shopping Period until today, you definitely still have some options (and time) to fill up your study card.

Science of the Physical Universe 20: What is Life? From Quarks to Consciousness—This Gen Ed course also fulfills *either* Science A or B for the Core, so it should probably be a pretty flexible option for most of you. Plus, even though the class has met two times already, we know from experience that most of your learning will take place in section. So just make sure you show up to that!

Ethical Reasoning 22: Justice—This ever-popular Sandel class met once last week, but it's not like they'll know you weren't there.

Computer Science 50: Introduction to Computer Science I—Missed the first lecture? Don't worry, it's already online, so you can catch up for the second class easily. And now that you can take it pass/fail and as a Gen Ed or Core course, it's more attractive than ever.

English 62: Diffusions: Castaways and Renegades—Double colon in the title aside, this course doesn't seem too complicated. The first lecture, according to the syllabus, only included an introduction and an overview. Read classics by Du Bois, Stowe, Whitman, Fitzgerald and Twain (you know you've always meant to). Added bonus? No midterm, no final exam.

History and Literature 90l: Stories of Slavery and Freedom—This course meets for the first time this week, and as far as we can tell, you don't need to have prepared anything in advance. Do brace yourself for a possible crowd, however—Professor Timothy P. McCarthy is popular and the Q guide ratings are stellar.

Science of Living Systems 11: Molecules of Life—Want an easy way to kill off a Gen Ed or Core requirement? This class is your answer (for Core people, it fulfills Science B). Professor David R. Liu and Professor Jon Clardy are sure to be a good time in this large and popular course about the small and unseen.

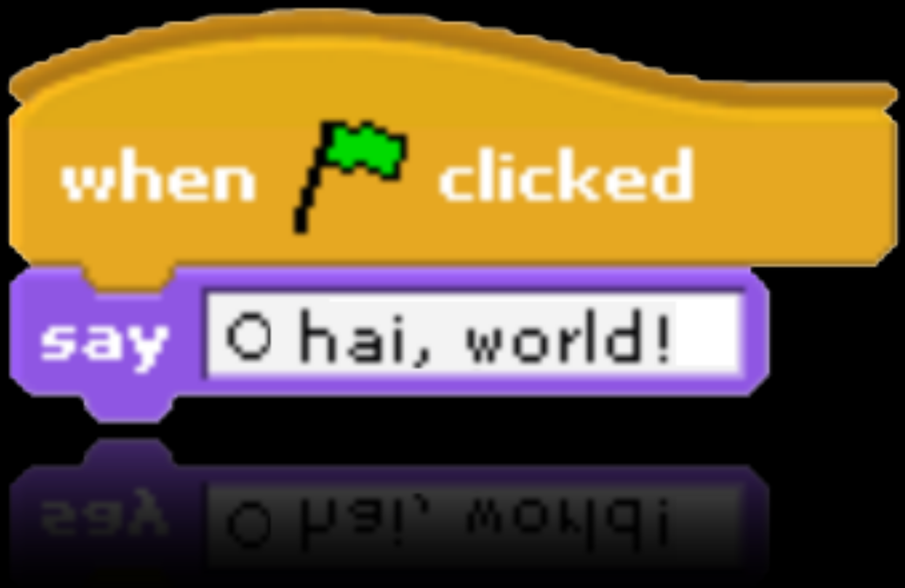
Photo courtesy of Wikimedia Commons.

Tagged: Academics

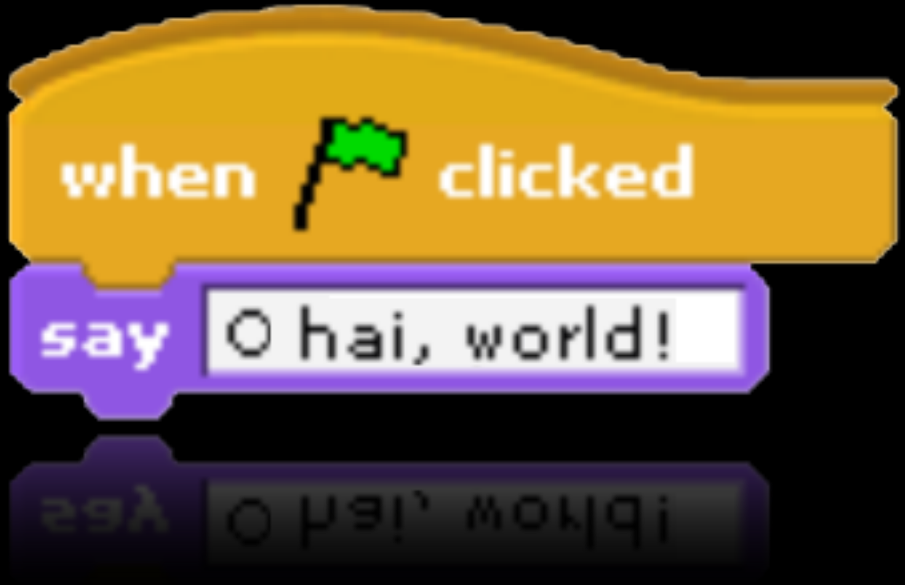
Published by Danielle Kim

office hours

sections

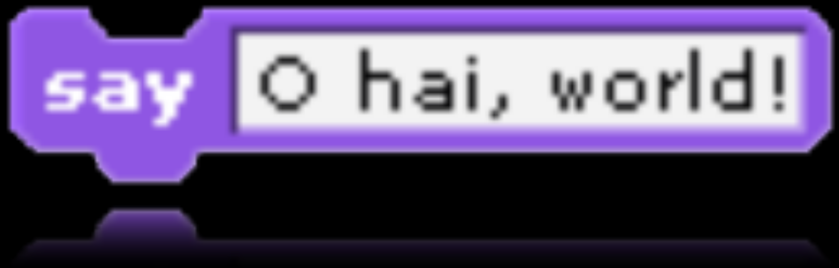


```
when [flag] clicked
say [O hai, world!]
```

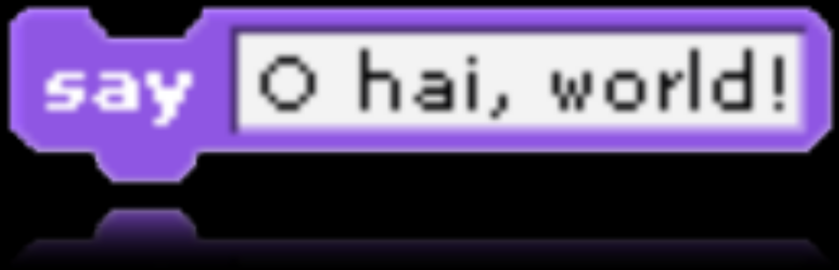


```
int
main()
{
    printf("O hai, world!\n");
}
```

statements

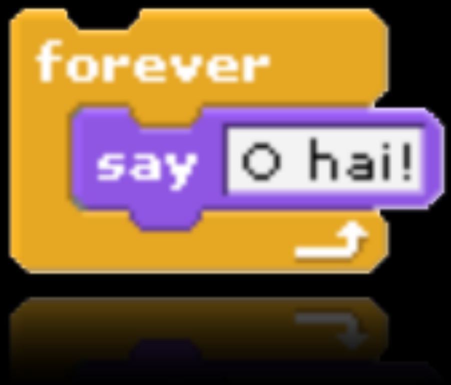


statements

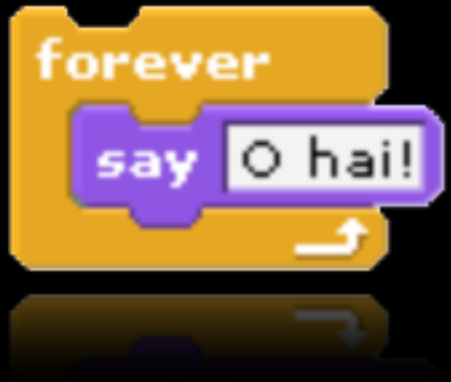


```
printf("O hai, world!\n");
```

loops

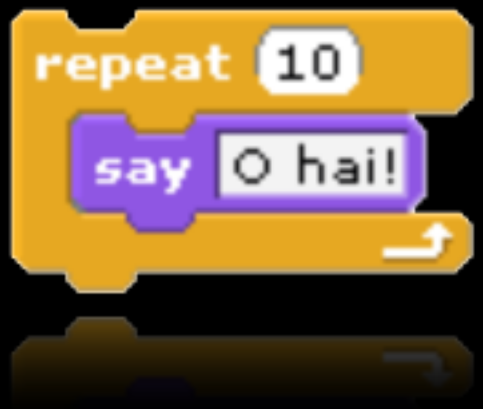

forever
say O hai!

loops



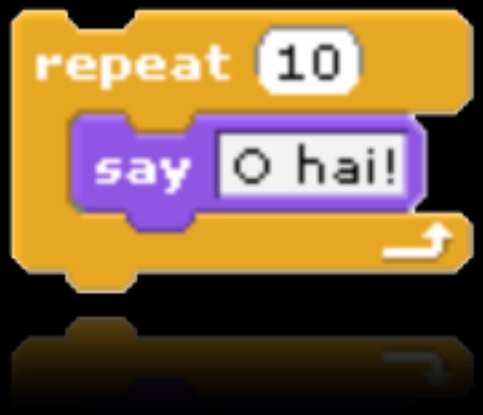
```
while (true)
{
    printf("O hai!\n");
}
```

loops



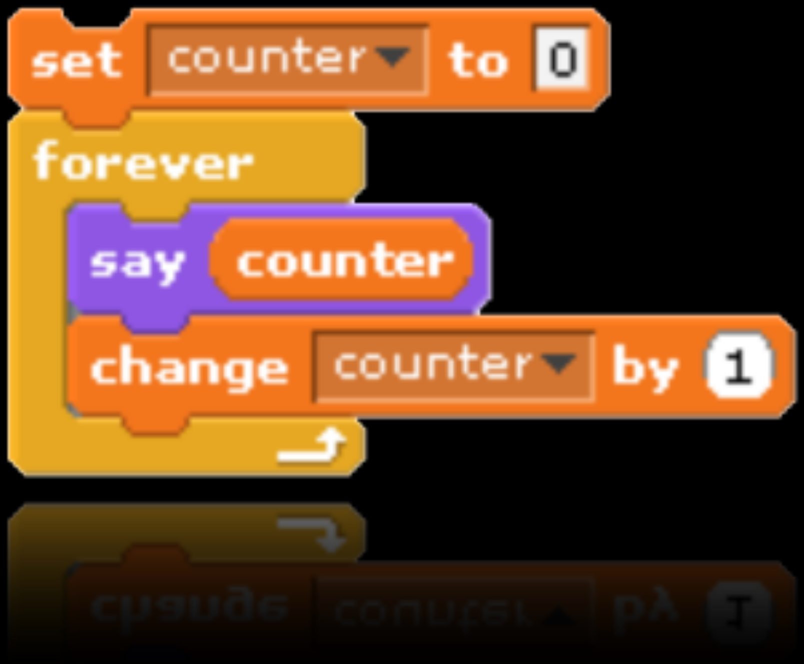
```
repeat 10
    say O hai!
```

loops



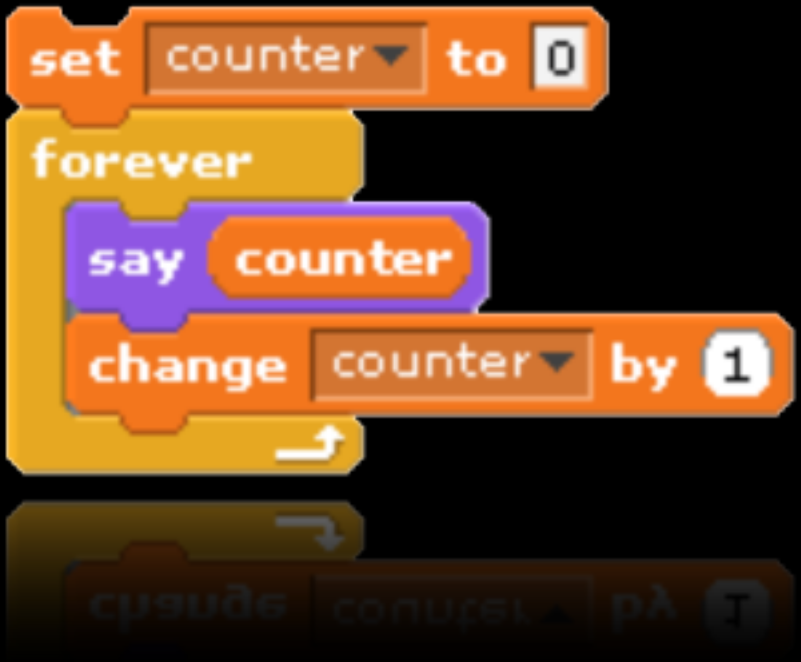
```
for (int i = 0; i < 10; i++)
{
    printf("O hai!\n");
}
```

variables



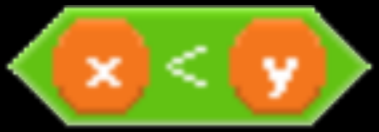
```
set counter to 0
forever
    say counter
    change counter by 1
```

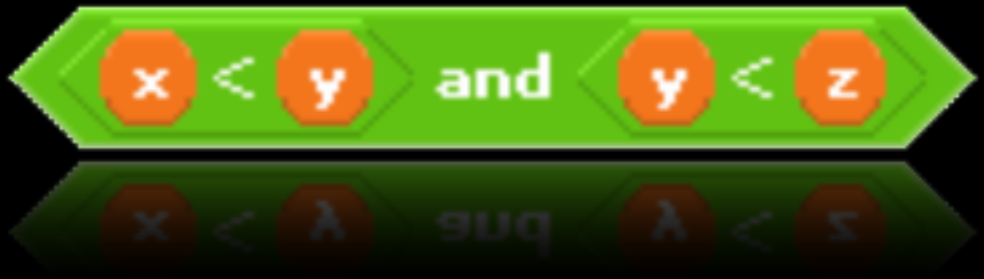
variables



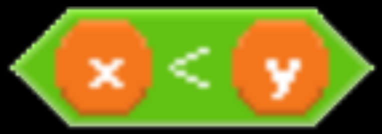
```
int counter = 0;
while (true)
{
    printf("%d\n", counter);
    counter++;
}
```

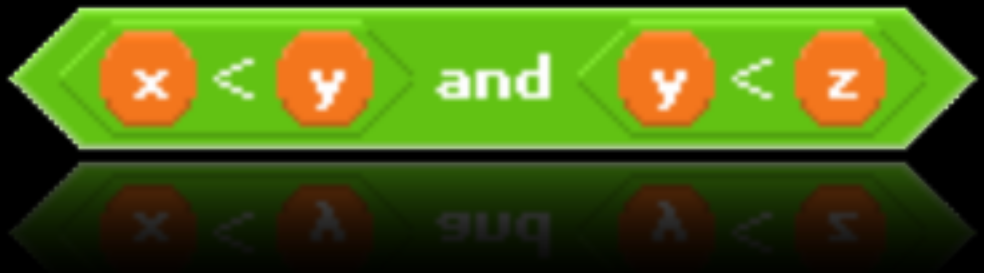
Boolean expressions





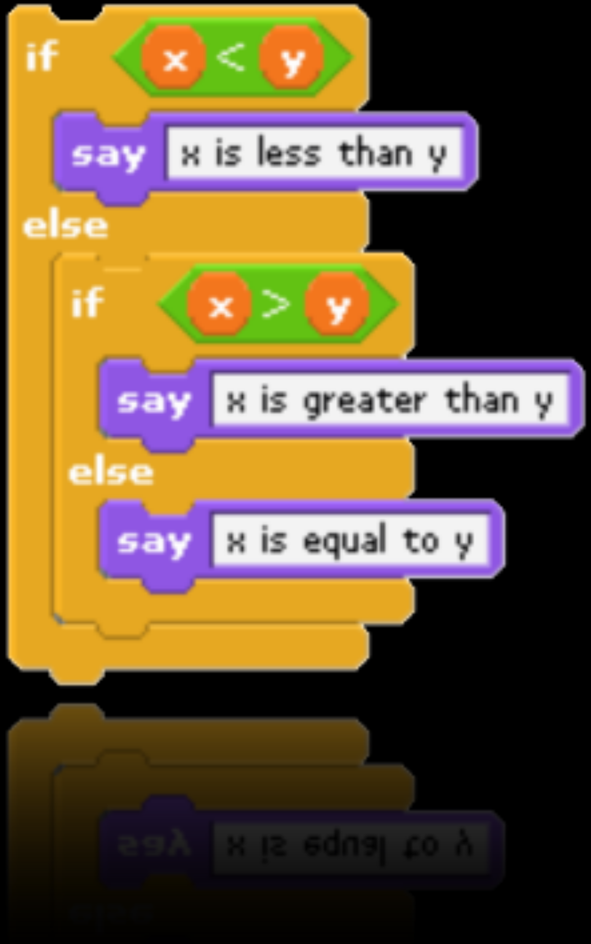
Boolean expressions





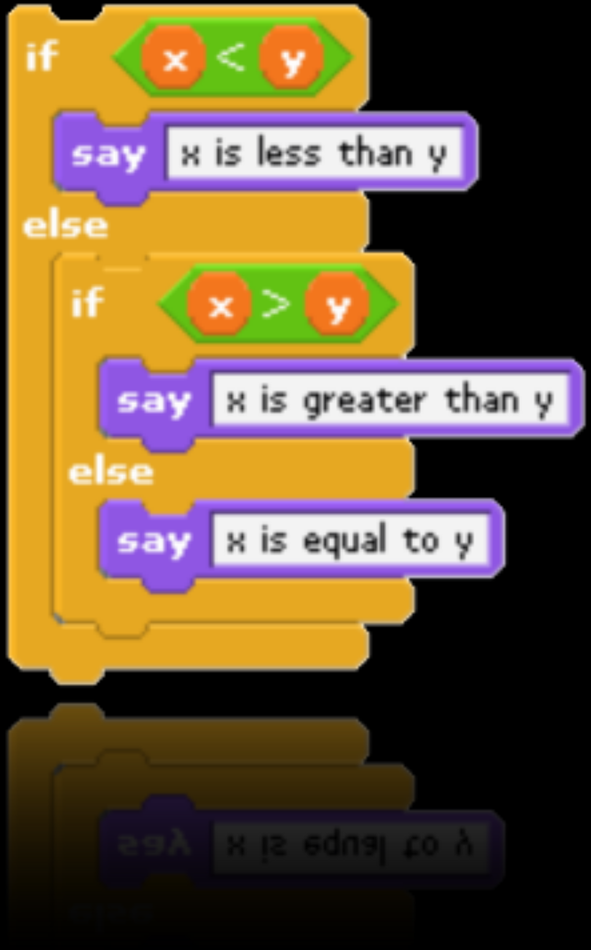
```
(x < y)
((x < y) && (y < z))
```

conditions



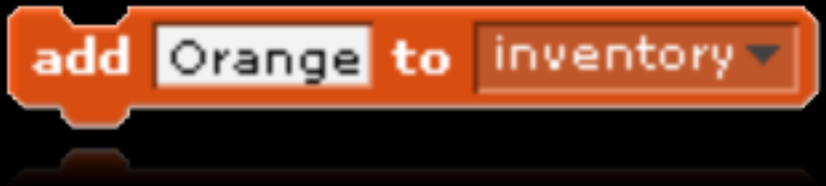
```
if <(x) < (y)>
    say [x is less than y]
else
    if <(x) > (y)>
        say [x is greater than y]
    else
        say [x is equal to y]
```

conditions



```
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");
}
```

arrays


add Orange to inventory

```
string inventory[1];
inventory[0] = "Orange";
```

```
#include <stdio.h>

int
main()
{
    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
                                ...
```

how to write a program

1. `nano hello.c`
2. `gcc hello.c`
3. `./a.out`

how to write a program

1. `nano hello.c`
2. `gcc -o hello hello.c`
3. `./hello`

how to write a program

1. `nano hello.c`
2. **`make hello`**
3. `./hello`

functions

`main   ...`

Standard Library

```
printf

...
```

CS50 Library

```
GetChar

GetDouble

GetFloat

GetInt

GetLongLong

GetString
```

`printf`

```
%c   %d   %f   %lld   %s   ...
```

escape sequences

```
\n   \r   \t   \'   \"   \\   \0   ...
```

math

+   -   *   /   %

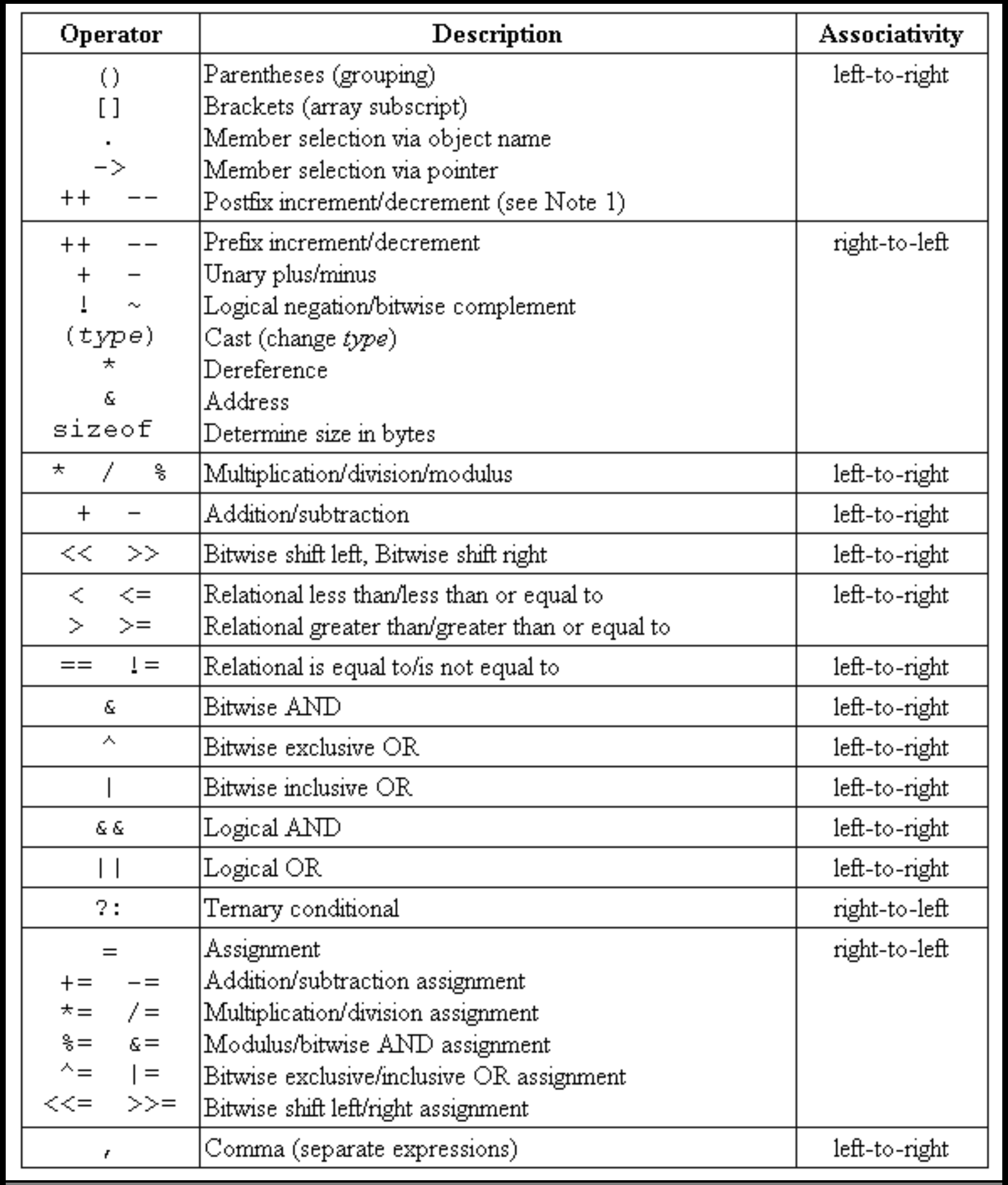
primitive types

`char   double   float   int   long long   ...`

CS50 types

`bool`   `string`   `...`

precedence



Operator	Description	Associativity
() [] . -> ++ --	Parentheses (grouping) Brackets (array subscript) Member selection via object name Member selection via pointer Postfix increment/decrement (see Note 1)	left-to-right
++ -- + - ! ~ (*type*) * & sizeof	Prefix increment/decrement Unary plus/minus Logical negation/bitwise complement Cast (change *type*) Dereference Address Determine size in bytes	right-to-left
* / %	Multiplication/division/modulus	left-to-right
+ -	Addition/subtraction	left-to-right
<< >>	Bitwise shift left, Bitwise shift right	left-to-right
< <= > >=	Relational less than/less than or equal to Relational greater than/greater than or equal to	left-to-right
== !=	Relational is equal to/is not equal to	left-to-right
&	Bitwise AND	left-to-right
^	Bitwise exclusive OR	left-to-right
|	Bitwise inclusive OR	left-to-right
&&	Logical AND	left-to-right
||	Logical OR	left-to-right
?:	Ternary conditional	right-to-left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment Addition/subtraction assignment Multiplication/division assignment Modulus/bitwise AND assignment Bitwise exclusive/inclusive OR assignment Bitwise shift left/right assignment	right-to-left
,	Comma (separate expressions)	left-to-right

http://www.difranco.net/cop2220/op-prec.htm

how to write a program

1. `nano hello.c`
2. `gcc -o hello hello` **`-lcs50`**
3. `./hello`

to be continued...