

# Art 486: Introduction to Interactive Media

[mcdo@umbc.edu](mailto:mcdo@umbc.edu)

Lecture 2, Sept 8

# Schedule

- Take Roll
- booleans, conditionals again
- self-testing
  
- loops: while, for

# Boolean algebra

- Boolean variables can hold either TRUE or FALSE
- operations on Booleans: AND, OR, NOT, and some others
- written as && (AND), || (OR), ! (NOT)

# the table of boolean operations

- Like a multiplication table, but for a different mathematical operation
  - yeah, you gotta memorize it

A	B	A && B	A    B	!A
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

# comparison operators

- take integers and floats and return Booleans
- “==” -> is equal to
  - different from “=”, which assigns!!
- >, <, >=, <=,
  
- (12>45) = FALSE
- (12<45) = TRUE

# compound Boolean operators

- parentheses work the same as before
- $((a < b) \&\&(a < c))$ 
  - first evaluate  $(a < b)$ ,  $(a < c)$ , get X and Y
  - then calculate X AND Y, return that
- $((((a < b) \&\&(a < c)) \|\| (b < c)))$ 
  - $Z = ((a < b) \&\&(a < c))$
  - result is  $(Z \|\| (b < c))$

# test yourself

- $((1+2)*(1+3)) = ?$
- $((3+4)+(6*6)) = ?$
  
- $((12<45)\&\&(45<12)) = ?$
- $((12<45)\|\|(45<12)) = ?$
- $((12<45)\&\&(13<45)) = ?$
- $((12<45)\&\&(!(45==12))) = ?$

# test yourself– true or false?

- $((1+2)*(1+3)) = (3*4) = 12$
- $((3+4)+(6*6)) = (7+36) = 43$
  
- $((12<45)\&\&(45<12)) = (T \&\& F) = F$
- $((12<45)\|\|(45<12)) = (T \|\| F) = T$
- $((12<45)\&\&(13<45)) = (T \&\& T) = T$
- $((12<45)\&\&(!(45==12))) = (T\&\&(!F)) = (T\&\&T) = T$

# nested “if” statements

```
if (a==b) {  
    if (a>c) {  
        trace “caterpie”;  
    } else {  
        trace “pikachu”;  
    }  
} else {  
    trace “geodude”;  
}
```

# good practices

- you don't have to use curly brackets if they would only enclose one statement
  - `if (a==b) trace "mew2";`
- **always use curly brackets!**
  - `if (a==b) if (a<c) trace "mew"; else trace "squirtle";`
  - does that "else" go with the first "if" or the second?

# good practices

- use indenting to track what's inside of what
- have a standard way of indenting, and use it!

```
if (a==b) { if (a<c)
{ trace "chimchar"
; } } else { trace
"mew"; }
```

```
if (a==b) {
    if (a<c) {
        trace "chimchar";
    }
} else {
    trace "mew";
}
```

# while loops

- cause the computer to do things over and over
- in general, they look like
  - “while ( ... ) { ... }”
- first, evaluate what’s in the ()
- if it’s TRUE, do the stuff in the {}
- repeat until the () returns FALSE
- quite possible to make one go forever
  - that’s called an “infinite loop”

# examples

```
a=0;
```

```
while (a<10) {
```

```
    a = a+1;
```

```
    trace(a);
```

```
}
```

# use a table to understand loops

```
a=0; while (a<10) { a = a+1; trace(a); }
```

<b>a</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>
a<10?	T	T	T	T	T	T	T	T	T	T	F!
a=a+1	1	2	3	4	5	6	7	8	9	10	-
trace a	1	2	3	4	5	6	7	8	9	10	-

# practice

- `b=10; while (b<15) { trace(b); b=b+1; }`
  - `b=10; while (b>5) { trace(b); b=b-1; }`
  - `b=10; while (b>5) { trace(b); b=b-2; }`
  - `b=10; while (b<20) { b=b+10; trace(b); }`
- 
- one of these will be on the test

# use a table to understand loops

```
b=10; while (b<15) { trace(b); b=b+1; }
```

Note that the order of print/increment is switched

<b>b</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	
b<15?	T	T	T	T	T	F	-
trace b	10	11	12	13	14	-	
b=b+1	11	12	13	14	15	-	

# use a table to understand loops

```
b=10; while (b>5) { trace(b); b=b-1; }
```

- hey! it goes backwards!

<b>b</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>
b>5?	T	T	T	T	T	F!
trace b	10	9	8	7	6	-
b=b-1	9	8	7	6	5	-

# use a table to understand loops

```
b=10; while (b>5) { trace(b); b=b-2; }
```

minus **two**; b decreases faster per loop;

fewer loops

<b>b</b>	<b>10</b>	<b>8</b>	<b>6</b>	<b>4</b>
b>5?	T	T	T	F!
trace b	10	8	6	-
b=b-2	8	6	4	-

# use a table to understand loops

```
b=10; while (b<20) { b=b+10; trace(b); }
```

plus 10, even faster

<b>b</b>	<b>10</b>	<b>20</b>
b<20?	T	F!
b=b+10	20	-
trace b	20	-

# Time Out

- did you see this? nice:
  - <http://www.kongregate.com/games/fastgames/little-wheel>
- Can you draw like that? Probably.
- Can you invent drawings like that? How would you go about doing that much invention?
  - collections of historical-period work
  - imagining manufacture
- Note sound awesomeness
  - collect drawings and sounds

# for loops

- while loops are error-prone
  - did you leave out the initialization?
  - do you change the counter each time?
- “for” loops are while loops with training wheels
  - “for ( ... ; ... ; ... ) { ... }”
  - first expression sets a counter
  - second tests it
  - third increments it
  - {}’s : same as with “while”

# examples

- `for (i=0; i<10; i=i+1) { trace(i); }`
  - prints 0,1,2,3,4,5,6,7,8,9
  - test happens before `{`
  - increment happens after `}`
- `for (i=10; i>0; i=i-1) { trace(i); }`
  - prints 10,9,8,7,6,5,4,3,2,1

