Conditionals

Logical Operators
Boolean Variable
Numerical Precision

Chapter 5, Examples 5-10, 5-11, 5-12, 5-13, 5-15, 5-16, 5-17, 5-18, 5-19, Robot 3 (p. 84-86).
What does this code draw after 10 frames?

```javascript
let a = 0;

function draw() {
  background(200);
  if (a < 5) {
    ellipse(50, 50, 50, 50);
  } else {
    line(0, 0, 99, 99);
  }
  a = a + 1;
}
```

Remember: the default canvas size is 100 by 100
What does this code draw after 10 frames?

```javascript
let a = 0;
function draw() {
  background(200);
  if (a < 5) {
    ellipse(50, 50, 50, 50);
  } else if (a > 5) {
    line(0, 0, 99, 99);
  } else {
    a = a + 1;
  }
}
```

Remember: the default canvas size is 100 by 100.
Recap: Last Class

- Boolean logic
- Boolean expression
Recap: What they allowed us to do

(1) create canvas
(2) set background color to gray
(3) draw a vertical line in the middle of the canvas
(4) if mouse is on the right half of the canvas, then draw red circle
Recap: Relational Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&gt;</code></td>
<td>greater than 50?</td>
<td><code>number &gt; 50</code></td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>less than 40?</td>
<td><code>number &lt; 40</code></td>
</tr>
<tr>
<td><code>===</code></td>
<td>equal to 73?</td>
<td><code>number === 73</code></td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>greater than or equal to 75?</td>
<td><code>number &gt;= 75</code></td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>less than or equal to 37?</td>
<td><code>number &lt;= 37</code></td>
</tr>
<tr>
<td><code>!==</code></td>
<td>not equal to 99?</td>
<td><code>number !== 99</code></td>
</tr>
</tbody>
</table>
Recap: Conditional Statement

- is your number greater than 50?
  (If yes, execute the code block. Otherwise, skip the code block and continue.)

```java
if (number > 50) {
    // code to execute if true
}
```
Recap: Conditional Statement

- is your number greater than 50?
  (If yes, execute the code block. Otherwise, skip the code block and continue.)

```java
if (number greater than 50) {
    // code to execute if true
}
```
But

- Often times, our instructions need to be more specific
Example: and

- My cold brew should have...
  
  (10 pumps vanilla and
   5 pumps caramel syrup and
   5 pumps classic syrup and
   5 pumps caramel sauce and
   5 pumps white mocha and
  ...
  Extra sweet cream and
  Extra whip)
Example: or

- If you are allergic to a shellfish, crab or lobster can cause allergic reaction
- If you are allergic to melon, watermelon or banana or avocado can cause allergic reaction

Credit: [http://allergy.bwh.harvard.edu/](http://allergy.bwh.harvard.edu/)
Example: at least (not less than)

- "You must pass at least the weighted exam average (50%) to pass." is same as saying,

  "you will pass if the weighted exam average is not less than 50%"

Exams

- Midterm Exam
  - Friday Feb 28 at 6:30 PM
  - 20% of grade
- Final Exam
  - 40% of grade

Important:
You must pass the weighted exam average to pass the course.

- If you get 60% on the midterm (12/20) and 40% on the final (16/40), then your weighted exam average is 28/60 and you will not pass the course (regardless of the rest of your course mark).
- If you get 35% on the midterm (7/20) and 60% on the final (24/40),
So

- We will learn how to use **logical operators** *(and, or, not)* to make our instructions more specific

<table>
<thead>
<tr>
<th>so far</th>
<th>today</th>
</tr>
</thead>
<tbody>
<tr>
<td>number greater than 50?</td>
<td>number greater than 50 <strong>AND</strong> less than 60?</td>
</tr>
<tr>
<td>number less than 40?</td>
<td>number less than 40 <strong>OR</strong> greater than 80?</td>
</tr>
<tr>
<td>number less than 32?</td>
<td>number <strong>NOT</strong> less than 32?</td>
</tr>
<tr>
<td>if (condition) {}</td>
<td>if (condition1 ▫ condition2 ... ▫ condition n) {}</td>
</tr>
</tbody>
</table>
| }                              | }
Guessing Game
Last time

- Boolean expressions:
  - is your number greater than 50?
  - is your number less than 40?
  - is your number equal to 73?
  - is your number greater than or equal to 75?
  - is your number less than or equal to 37?
  - is your number not equal to 99?
This time

- Boolean expressions with logical operators:
  - is your number greater than 50 and less than 60?
  - is your number less than 40 or greater than 80?
  - is your number not less than 32?
Logical Operators

&& means "and"

|| means "or"

! means "not"
This time

- Boolean expressions *with logical operators*:
  - is your number greater than 50 and less than 60?
  - is your number less than 40 or greater than 80?
  - is your number not less than 32?
This time

- Boolean expressions with logical operators:
  - is your number greater than 50 and less than 60?
    \( \text{number} > 50 \land \text{number} < 60 \)
  - is your number less than 40 or greater than 80?
    \( \text{number} < 40 \lor \text{number} > 80 \)
  - is your number not less than 32?
    \( \neg (\text{number} < 32) \)
# Logical Operator “Truth Table”

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Evaluates To</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>true &amp;&amp; true</code></td>
<td>= true</td>
</tr>
<tr>
<td><code>true &amp;&amp; false</code></td>
<td>= false</td>
</tr>
<tr>
<td><code>false &amp;&amp; false</code></td>
<td>= false</td>
</tr>
<tr>
<td>`true</td>
<td></td>
</tr>
<tr>
<td>`true</td>
<td></td>
</tr>
<tr>
<td>`false</td>
<td></td>
</tr>
<tr>
<td><code>!true</code></td>
<td>= false</td>
</tr>
<tr>
<td><code>!false</code></td>
<td>= true</td>
</tr>
</tbody>
</table>
### Logical Operator “Truth Table”

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Evaluates To</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>true &amp;&amp; true</code></td>
<td><code>true</code></td>
</tr>
<tr>
<td><code>true &amp;&amp; false</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>false &amp;&amp; false</code></td>
<td><code>false</code></td>
</tr>
</tbody>
</table>

## and

- `true && true` = `true`
- `true && false` = `false`
- `false && false` = `false`
Logical Operator “Truth Table”

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Evaluates To</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>true</td>
<td></td>
</tr>
<tr>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

Credit: [http://allergy.bwh.harvard.edu/](http://allergy.bwh.harvard.edu/)
# Logical Operator “Truth Table”

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Evaluates To</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>= false</td>
</tr>
<tr>
<td>false</td>
<td>= true</td>
</tr>
</tbody>
</table>

**not**

- !true
  - = false
- !false
  - = true
Logical Operator Examples

let a = 15;
let b = 5;

(a > 10 && b < 10)

(a < b || b < a)

(a === 10 && a < 20 && b > 0)

(a < 10 || 12 > b * 2)

(a + b > 16 || b - a > 0)

(a > 10 && a < 12 || b > 3 && b < 7)

!(a > 10)
Use Cases with Logical Operator

- Specifying an area
Which of the instruction sets is needed to create this?

(a) when mouseX is greater than 50 and mouseY is greater than 50, change fill to red
(b) when mouseY is less than 50 and mouseX is less than 50, change fill to blue
(c) when mouseX is greater than 50 and mouseY is less than 50, change fill to no fill
(d) when mouseY is greater than 50 and mouseX is less than 50, change fill to blue
(e) when mouseX is greater than 50 and mouseY is less than 50, change fill to red
(f) when mouseY is greater than 50 and mouseX is greater than 50, change fill to no fill

The canvas size is 100 x 100.

A (a) (b) (c)  
B (d) (e) (c)  
C (a) (b) (f)  
D (d) (e) (f)

NOTE: Find the best answer.
The canvas size is 100 x 100.

A (a) (b) (c) 
B (d) (e) (c) 
C (a) (b) (f) 
D (d) (e) (f) 

NOTE: Find the best answer.
Use Cases with Logical Operator

- Specifying an area
- Accepting additional keys
extend shaper to work with CAPS LOCK on

// if little c OR big C
if (key === 'c' || key === 'C') {
    shape = 1;
...

https://editor.p5js.org/cs105.sketches/d69-jjtUT
What is drawn on the 6th frame?

```
let a = 0;
let b = 0;

function draw() {
  background(220);
  if (a < 10 || b > 20) {
    ellipse(25, 25, 30, 30);
  } else {
    rect(60, 60, 30, 30);
  }
  a = a + 1;
}

function mousePressed() {
  b = 100;
}
```

Remember: the default canvas size is 100 by 100
What is drawn on the 27th frame?

```javascript
let a = 0;
let b = 0;

function draw() {
  background(220);
  if (a < 10 || b > 20) {
    ellipse(25, 25, 30, 30);
  } else {
    rect(60, 60, 30, 30);
  }
  a = a + 1;
}

function mousePressed() {
  b = 100;
}
```

Remember: the default canvas size is 100 by 100
What is drawn 28 frames after a mouse press?

```
let a = 0;
let b = 0;
function draw() {
  background(220);
  if (a < 10 || b > 20) {
    ellipse(25, 25, 30, 30);
  } else {
    rect(60, 60, 30, 30);
  }
  a = a + 1;
}
function mousePressed() {
  b = 100;
}
```

Remember: the default canvas size is 100 by 100
Use Cases with Logical Operator

- Specifying an area
- Accepting additional keys
Use Cases with Logical Operator

- Specifying an area
- Accepting additional keys
Use Cases with Logical Operator

- Specifying an area
  - e.g., **rectangle hit test** for buttons on websites
- Also called a “rollover” or “mouseover”

Creating a rectangle hit test

(28, 30)  (72, 30)
(28, 43)  (72, 43)

Hover Me

(99, 99)
rect-hittest

...  

// the hit test  
if (mouseX >= x && mouseX <= x + w &&  
    mouseY >= y && mouseY <= y + h) {
    fill("#ff0000"); // red
} else {
    fill("#FFFFFF"); // white
}

...
Boolean Variables

- User-defined variables that store true or false
- They provide additional means to organize our program (see demo)

let a = 9 < 10;
let b = false;
print(a); // prints true
print(b); // prints false
Boolean Variables Analogy

- Light switch on/off

let lightOn = true;
Boolean Variables Analogy

- Toggle button **on/off**

```javascript
let toggleOn;

// Turn on Wi-Fi
toggleOn = true;

// Turn off Wi-Fi
toggleOn = false;
```
Boolean Variables Represent State (cf. Lecture 07)

- Current settings of all drawing attributes (e.g. fill, stroke, strokeWeight, etc.)
- What line of the program is executed next? (i.e. Sequential Control Flow)
- Current values of all variables (use variables to “save” and “update” state of things)
Boolean Variables Represent Two Possible States

- When designing programs, we can think about what can be represented in two different states
rect-hittest with boolean variable

...  

// the hit test
if (mouseX >= x && mouseX <= x + w &&
    mouseY >= y && mouseY <= y + h) {
    fill("#ff0000"); // red
} else {
    fill("#FFFFFF"); // white
}

...

Starter: https://editor.p5js.org/sanghosuh/sketches/nmoszZH
https://editor.p5js.org/sanghosuh/sketches/Ryblkhy-
let day = true;

function draw() {
    if (day === true) {
        background("#83E7FF"); // blue sky
    } else {
        background("#01396F"); // night sky
    }
}

function mousePressed() {
    // "not day" means the opposite
    // of day's current boolean value
    day = !day;
}

Starter: https://editor.p5js.org/cs105/sketches/om4M9GtrJ
https://editor.p5js.org/cs105/sketches/dzPjG_5yR
What is drawn on the 3rd frame?

```javascript
let b = true;
function draw() {
  background(200);
  if (b) {
    ellipse(50, 50, 50, 50);
  } else {
    line(0, 0, 99, 99);
  }
  b = !b;
}
```
Built-in boolean variables

- `mousePressed()` is an event function
  - it's called once when the mouse button is pressed
- `mouseIsPressed` is a built-in boolean variable
  - it's true when the mouse button is pressed, false otherwise
- Same for `keyIsPressed` and `keyPressed()`
**draw (extra demo)**

*draw when mouse button is held down*
*erase when any key is held down*
*erase whole drawing with “x” key*

`pmouseX`
`pmouseY`

`mouseIsPressed`
`keyIsPressed`
`keyPressed()`

Reference: https://p5js.org/reference/

Starter: https://editor.p5js.org/cs105/sketches/vZKWgLEmp

https://editor.p5js.org/cs105/sketches/gU0ur1yce
Numerical Representation

- A civil engineer doesn't care about the difference between 10 meters and 10.0001 meters
- 0.0001 meters is a huge difference for a microchip designer, but all measurements will be less than about 0.01 meters
- A physicist needs to use the speed of light (about 300000000) and Newton's gravitational constant (0.00000000000667) in the same equation
Numerical Precision

- How many numbers are there between 0 and 1?
- How many decimal points in one-third (1/3)?
- Computers can not always do exact math
- This has implications for equality testing ...
let y = 0;

function setup() {
    createCanvas(50, 200);
    background(220);
}

function draw() {
    if (y === 1) {
        // this code block may never be executed due to numerical precision
        stroke(255, 0, 0);
        print("y is exactly 3");
    }
    line(0, y * 100, width, y * 100);
}

function keyPressed() {
    y = y + 0.1;
    print(y);
}

https://editor.p5js.org/cs105/sketches/XiSjxkZip
Revisit: A2 Exquisite Corpse

- I have written up a program for you guys to play with it: https://sanghosuh.github.io/blog/coding/cs105-exquisite-corpse
- It shows how modular design can scale

Assignment 2

Due: Friday, January 17 at 11:59 PM

You will write a JavaScript P5 sketch using shapes and colours to draw a “body” (a robot, a person, an animal, an alien, etc.). Your drawing will be constrained so it can be divided up for a class-wide “exquisite corpse”.

Exquisite Corpse

An exquisite corpse is a way to generate random text or pictures. We’ll be creating a variation where separate images are combined to create a random body (see below).