CS 101
Boolean Expressions

Lecture 19
Boolean Variable

- We’ve seen and used boolean variables

```java
boolean skyIsBlue = True;
boolean UofARocks = True;
boolean ASURocks = False;
if (UofARocks) {
    // ...
}
```
Boolean Expressions

- A **boolean expression** is a logical statement that always evaluates to either **True** or **False**.
- You’ve actually been using these already
- For example: **if-statements**

```java
boolean UofARocks = True;
if (UofARocks) {
    // ...
}

if (mouseX > 100) {
    // ...
}

int xPos = 100;
int yPos = 200;
if (xPos == yPos) {
    // ...
}
```
Boolean Expressions

- A **boolean expression** is a logical statement that always evaluates to either **True** or **False**.
- You’ve actually been using these already
- For example: for-loops

```java
for (int i = 0; i < 100; i+= 10) {
    rect(i, 10, 5, 5);
}

for (int w = 0; w < mouseX; w += 50) {
    ellipse(w, w, 20, 20);
}
```
Boolean Operators

- So far, there are a few ways that we know how to make a boolean expression
  - Just write the name of a boolean variable (for example: `keyPressed` or `mousePressed`)
  - Compare two non-boolean values (numbers) with comparison operators
    
    `<  >  `<=  `>=  `==  `!=
  - Any others we’ve used in class?
Boolean Operators

- Two new Boolean operators:
  
  ```
  &&
  ||
  ```

- These operators work like the number comparison ones, where there must be some value to the left and right that they are operating on.

- However, the left and right-hand sides must be booleans!

  ```
  expr_1 && expr_2
  ```

- These compare boolean values.
Boolean Operators

- **&&** (logical and)
  - $X \&\& Y$ is true if both $X$ and $Y$ are true
  - If one is True or neither are true, it makes the whole expression false

- **||** (logical or)
  - $X \mid\mid Y$ is True if both $X$ and $Y$ are true, or if one of them is true
  - If neither are true, it makes the whole expression false
Truth tables for the boolean operators && and ||

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>&amp;&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

| X   | Y   | ||  |
|-----|-----|-----|
| True| True| True|
| True| False| True|
| False| True| False|

Example

What will be drawn when this code runs?

Why?

```java
boolean skyIsBlue = true;
boolean UofARocks = true;
boolean ASURocks = false;

if (UofARocks || ASURocks) {
    rect(100, 100, 50, 50);
}

if (skyIsBlue && false) {
    ellipse(200, 200, 70, 70);
} else if (UofARocks) {
    triangle(50, 50, 100, 100, 200, 200);
}
```
Boolean Operators

- The left and right-hand side of a boolean operator can be another boolean expression, not just a `true, false`, or a boolean variable.
- For example

```java
boolean skyIsBlue = true;
boolean UofARocks = true;
boolean ASURocks = false;

if (UofARocks || ASURocks && UofARocks) {
    rect(100, 100, 50, 50);
}
```
What will be drawn when this code runs?

Why?

```java
void draw()
    if (mousePressed || (keyPressed && key == 'a')){
        ellipse(mouseX, mouseY, 30, 30);
    } else {
        rect(mouseX-15, mouseY-15, 30, 30);
    }
```
Example

What is the difference?

```java
void draw()
    if (keyPressed && key == 'a') {
        ellipse(mouseX, mouseY, 30, 30);
    }
}

void draw()
    if (keyPressed || key == 'a') {
        ellipse(mouseX, mouseY, 30, 30);
    }
}

void draw()
    if (key == 'a') {
        ellipse(mouseX, mouseY, 30, 30);
    }
}
```
Boolean Operators

- Some of you probably did something like this in the Tag assignment
- How could you use boolean operators so that only one if-statement is used?

```cpp
int tagX = 0;
int tagY = 0;
int hideX = 0;
int hideY = 0;

if (tagX == hideX) {
    if (tagY == hideY) {
        text("Tag, you are it!", 10, 20);
    }
}
```
Boolean Operators

• How could you use boolean operators so that only one if-statement is used?

```java
int tagX = 0;
int tagY = 0;
int hideX = 0;
int hideY = 0;

if (tagX == hideX) {
    if (tagY == hideY) {
        if (mousePressed == true) {
            text("Tag, you are it!",10,20);
        }
    }
}
```
Using Boolean Expressions

- Download website-buttons.pde from the lectures page on the class website
- How can we reduce the number of if-statements needed in the program?
- (Focus on the if statements related to the Amazon button.)
Not (!)

- Another unary operator: `not ( ! )`
- Put this to the left of any boolean expression
- Turns value from `True` to `False`, or vice-versa
- `&&`, `||` and `!` can also all be used with parentheses to force the order of operation, just like with math operators
Example

What happens when this code runs?

Why?

```java
void draw()
    if (!keyPressed && key == 'a') {
        ellipse(mouseX, mouseY, 30, 30);
    } else if (!mousePressed && !keyPressed) {
        rect(mouseX-15, mouseX-15, 30, 30);
    }
}
```
What will the value of each variable be?

boolean iAmCool = True || False

boolean youAreCool = (False && False) || True

boolean weAreCool = iAmCool && youAreCool

boolean theyAreCool = (! weAreCool) || False
What will the value of each variable be?

boolean a = true || true && true

boolean b = false && false || false

boolean c = a || (! b)

boolean d = b && c && a
Materials

- Required Materials
  - Practice on your own!