# Review Logic and if Statements

Hide tutorial navigation

*This is a review of what we covered in this tutorial on logic and if statements.*

We often want to be able to "conditionally" do things in our programs - we want to be able to say "if this thing is true, then do X but if this other thing is true, then do Y." It's like when we wake up in the morning - "if its raining outside, then I take an umbrella, but if its sunny, I wear sunglasses." We can do things conditionally in our programs using ***if statements*** and ***if/else statements*** combined with ***conditional expressions***.

An ***if statement*** tells the program to execute a block of code, as long as a condition is true. In the code below, we output a message only if `x` is greater than 0:

```
var x = 5;

if (x > 0) {
  text('x is a positive number!', 200, 200);
}
```

Since x is 5, which is greater than 0, we would see the message on the canvas. If we changed x to -1, we wouldn't see the message show up at all, since x wouldn't be greater than 0.

The `x > 0` is what we call a ***conditional expression*** - which means it's an expression that evaluates to either `true` or `false`. When a value is either `true` or `false`, we call it a ***boolean*** value (as opposed to a number or a string). For example, we could just display the conditional expression:

```
text(x > 0, 200, 200); // Displays "true"
```

We could also store it into a variable and then display it:

```
var isPositive = x > 0;
text(isPositive, 200, 200);
```

We would then say that `isPositive` is storing a boolean value, because it's either `true` or `false`, depending on what we set x to.

We have many ways of creating conditional expressions that will evaluate to `true` or `false`, because we have many comparison operators. Here are the most popular:

Assuming the following variable, here are the most common comparison operators and expressions that would be true with them:

```
var myAge = 28;
```

| Operator | Meaning | True expressions |
|---|---|---|
| `===` | Strict equality | `myAge === 28` |
| `!==` | Strict inequality | `myAge !== 29`<br>`28 !== 29` |

| Operator | Meaning | True expressions |
|---|---|---|
| > | Greater than | `myAge > 25`<br>`28 > 25` |
| >= | Greater than or equal | `myAge >= 28`<br>`28 >= 28` |
| < | Less than | `myAge < 30`<br>`28 < 30` |
| <= | Less than or equal | `myAge <= 28`<br>`28 <= 28` |

It is a very common mistake to confuse the assignment operator (=) with the equality operator (===), because they both use equal signs, but they are quite different. The assignment operator will actually change the value of the variable, whereas the equality operator will just read the value of the variable and see if it's equal to something. For example:

```
var x = 2 + 2; // Sets it equal to 4

if (x === 4) { // Asks the question, "does this equal 4?"
   text("yep, 2 + 2 = 4!", 200, 200);
}
```

We sometimes want to combine multiple comparisons together in a conditional expression, and that's why we have ***logical operators***. These operators let us say things like "if both X and Y are true" or "if either X or Y are true" in our programs.

If we want to require that two conditions are both true, we can use the `&&` operator ("and"):

```
var degreesOutside = 70;
var numberOfClouds = 50;
if (degreesOutside > 70 && numberOfClouds < 5) {
  text("Wear sun screen!", 200, 200);
}
```

We often use that in our programs here for checking that a user's mouse position is inside a rectangle (to mimic a button), in which case we need multiple `&&` operators:

```
rect(100, 50, 100, 100);

mousePressed = function() {
    if (mouseX > 100 && mouseX < 200 && mouseY > 50 && mouseY < 150) {
      text("You pressed it!", 80, 75);
    }
};
```

If we only care that at least one condition is true, then we can use the `||` operator ("or"):

```
var degreesOutside = 70;
var numberOfClouds = 50;
if (degreesOutside < 70 || numberOfClouds < 5) {
  text("Wear sun screen, even if it's not hot out!", 200, 200);
}
```

We can use both `&&` and `||` in the same expression, if we have some very complex condition to check. Just use parentheses to group expressions, so that the program isn't confused about what order to evaluate them:

```
var myAge = 28;
if ((myAge >= 0 && myAge < 3)  || myAge > 90) {
  println('You\'re not quite in your peak.');
}
```

Now let's return to ***if statements***. We often want to execute some block of code in the case that the if condition wasn't true - in that case, we add an ***else statement***.

```
var age = 28;
if (age > 16) {
  println('Yay, you can drive!');
} else {
  println('Sorry, but you have ' + (16 - age) + ' years until you can
drive.');
}
```

Sometimes we want to check multiple conditions, and do different things based on each, in which case we can use `else if`:

```
var age = 20;
if (age >= 35) {
  println('You can vote AND hold any place in government!');
} else if (age >= 30) {
  println('You can vote AND run for the Senate!');
} else if (age >= 18) {
  println('You can vote!');
} else {
  println('You have no voice in government!');
}
```

You can keep checking conditions as much as you want - just make sure that each of your conditions is actually reachable. Test your code by changing variables until you make it inside each of the blocks of code, so that you know it all works.

As you can hopefully now see, conditionals are an important part of programming and let us create much more powerful and flexible programs.