Here’s the logical expression used to represent Julia’s weekend plans:

**var** colt = "not busy";

**var** weather = "nice";

**if** (colt === "not busy" && weather === "nice") {

console.log("go to the park");

}

***Prints:****"go to the park"*

Notice the && in the code above.

The && symbol is the logical AND operator, and it is used to combine two logical expressions into one larger logical expression. If **both** smaller expressions are *true*, then the entire expression evaluates to *true*. If **either one** of the smaller expressions is *false*, then the whole logical expression is *false*.

Another way to think about it is when the && operator is placed between the two statements, the code literally reads, "if Colt is not busy *AND* the weather is nice, then go to the park".

**Logical expressions**

**Logical expressions** are similar to mathematical expressions, except logical expressions evaluate to either *true* or *false*.

11 != 12

***Returns:****true*

You’ve already seen logical expressions when you write comparisons. A comparison is just a simple logical expression.

Similar to mathematical expressions that use +, -, \*, / and %, there are logical operators &&, || and ! that you can use to create more complex logical expressions.

**Logical operators**

**Logical operators** can be used in conjunction with boolean values (true and false) to create complex logical expressions.

By combining two boolean values together with a logical operator, you create a *logical expression* that returns another boolean value. Here’s a table describing the different logical operators:

| **Operator** | **Meaning** | **Example** | **How it works** |
| --- | --- | --- | --- |
| && | Logical AND | value1 && value2 | Returns true if **both** value1 **and** value2 evaluate to true. |
| || | Logical OR | value1 || value2 | Returns true if **either** value1 **or** value2 (**or even both!**) evaluates to true. |
| ! | Logical NOT | !value1 | Returns the **opposite** of value1. If value1 is true, then !value1 is false. |

By using logical operators, you can create more complex conditionals like Julia’s weekend example.

Note - This video does not have an audio. It was created as a visual to aid learning.

Logical operators can be used to combine multiple conditional statements into a single statement.

***TIP:****Logical expressions are evaluated from left to right. Similar to mathematical expressions, logical expressions can also use parentheses to signify parts of the expression that should be evaluated first.*

**QUESTION 1 OF 3**

What value of [BLANK] would make the following expression evaluate to false. Notice the ! right at the beginning!

!([BLANK] === 4) && "STRing" === "STRing"

***Returns:****false*

* 

-4

* 4
* 

"4"

* 

"-4"

SUBMIT

**QUESTION 2 OF 3**

Select the operator that would make the following expression evaluate to true.

3 < -10 [BLANK] "James" !== "james"

***Returns:****true*

* 

!

* ||
* 

&&

SUBMIT

**QUESTION 3 OF 3**

Evaluate the following logical expressions. Check the ones that evaluate to true.

* true || false
* 

false && false

* 

!true

* (13 > -7) || (false == 0)
* 

(10 === "10") && (1 <= 2)

* (3 != 6 % 3) && !(24 > 45) && (!false)

SUBMIT

NEXT