Truth Table

We can also use operators that use Boolean values as inputs and output. Logical operators, also known as Boolean operators, can be used to create Boolean expressions.

Logical operators include:

> AND `&&`: Both expressions are evaluated and will return True *only* if both expressions evaluate to True. Otherwise, it will return False.
>
> OR `||`: Both expressions are evaluated and will return True if *at least one* of the expressions evaluates to True. Otherwise, it will return False.
>
> NOT `!`: An expression, no matter its logical value, evaluates to its opposite. What is True becomes False and what is False becomes True.

Let's look at some examples:

```
bool andExample = ((4 > 1) && (2 < 7));
// (True AND True) evaluates to True
```

In this example, both comparisons evaluate to True, so the overall expression evaluates to True.

```
bool orExample = ((8 > 6) || (3 > 6));
// (True OR False) evaluates to True
```

Here, only one comparison evaluates to True and the other evaluates to False, so the expression evaluates to True.

```
bool notExample = !(1 < 3);
// NOT (True) evaluates to False
```

In this last example, the comparison evaluates to True, so the expression evaluates to False.

A common way to visualize these relationships is using a diagram known as a *truth table*. Truth tables allow us to quickly see what the outcome is for different relationships between Boolean values. Handling two Boolean values is simple, but longer expressions can be very complex. It’s crucial that you are familiar with these fundamentals before going ahead.

☑Instructions

We can visualize the relationship of Boolean values and logical operators using a diagram known as a *truth table*. Truth tables allow us to quickly see what the outcome is for different relationships between boolean values.

For example, what is the result of False OR True?