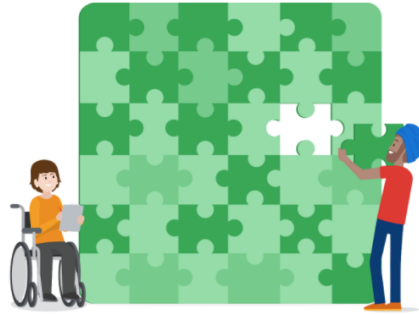


Logical operators and conditional statements

Earlier, you learned that an **operator** is a symbol that identifies the type of operation or calculation to be performed in a formula. In this reading, you'll learn about the main types of logical operators in R, and how they can be used to create conditional statements in R code.

Logical operators



Logical operators return a logical data type such as TRUE or FALSE.

There are three primary types of logical operators:

- AND (sometimes represented as & or && in R)
- OR (sometimes represented as | or || in R)
- NOT (!)

This table summarizes the logical operators:








AND operator "&"	OR operator " "	NOT operator "!"
<p>The AND operator takes two logical values. It returns TRUE only if <i>both</i> individual values are TRUE. This means that TRUE & TRUE evaluates to TRUE. However, FALSE & TRUE, TRUE & FALSE, and FALSE & FALSE all evaluate to FALSE.</p>	<p>The OR operator () works in a similar way to the AND operator (&). The main difference is that at least one of the values of the OR operation must be TRUE for the entire OR operation to evaluate to TRUE.</p> <p>This means that TRUE TRUE, TRUE FALSE, and FALSE TRUE all evaluate to TRUE. When both values are FALSE, the result is FALSE.</p>	<p>The NOT operator (!) simply negates the logical value it applies to. In other words, !TRUE evaluates to FALSE, and !FALSE evaluates to TRUE.</p>
<p>If you run the the corresponding code in R, you get the following results:</p> <pre>> TRUE & TRUE [1] TRUE > TRUE & FALSE [1] FALSE > FALSE & TRUE [1] FALSE > FALSE & FALSE [1] FALSE</pre>	<p>If you write out the code, you get the following results:</p> <pre>> TRUE TRUE [1] TRUE > TRUE FALSE [1] TRUE > FALSE TRUE [1] TRUE > FALSE FALSE [1] FALSE</pre>	<p>When you run the code, you get the following results:</p> <pre>> !TRUE [1] FALSE > !FALSE [1] TRUE</pre> <p>Just like the OR and AND operators, you can use the NOT operator in combination with logical operators. Zero is considered FALSE and non-zero numbers are taken</p>

<p>You can illustrate this using the results of our comparisons. Imagine you create a variable <code>x</code> that is equal to 10.</p> <pre>x <- 10</pre> <p>To check if “<code>x</code>” is greater than 3 but less than 12, you can use <code>x > 3</code> and <code>x < 12</code> as the values of an “AND” expression.</p> <pre>x > 3 & x < 12</pre> <p>When you run the function, R returns the result TRUE.</p> <pre>[1] TRUE</pre> <p>The first part, <code>x > 3</code> will evaluate to TRUE since 10 is greater than 3. The second part, <code>x < 12</code> will also evaluate to TRUE since 10 is less than 12. So, since <i>both</i> values are TRUE, the result of the AND expression is TRUE. The number 10 lies between the numbers 3 and 12.</p> <p>However, if you make “<code>x</code>” equal to 20, the expression <code>x > 3 & x < 12</code> will return a different result.</p>	<p>For example, suppose you create a variable <code>y</code> equal to 7. To check if <code>y</code> is less than 8 or greater than 16, you can use the following expression:</p> <pre>y <- 7 y < 8 y > 16</pre> <p>The comparison result is TRUE (7 is less than 8) FALSE (7 is not greater than 16). Since only one value of an OR expression needs to be TRUE for the entire expression to be TRUE, R returns a result of TRUE.</p> <pre>[1] TRUE</pre> <p>Now, suppose <code>y</code> is 12. The expression <code>y < 8 y > 16</code> now evaluates to FALSE (12 < 8) FALSE (12 > 16). Both comparisons are FALSE, so the result is FALSE.</p> <pre>y <- 12 y < 8 y > 16 [1] FALSE</pre>	<p>as TRUE. The NOT operator evaluates to the opposite logical value.</p> <p>Let's imagine you have a variable “<code>x</code>” that equals 2:</p> <pre>x <- 2</pre> <p>The NOT operation evaluates to FALSE because it takes the opposite logical value of a non-zero number (TRUE).</p> <pre>> !x [1] FALSE</pre>
---	---	---

<pre>x <- 20 x > 3 & x < 12 [1] FALSE</pre> <p>Although $x > 3$ is TRUE ($20 > 3$), $x < 12$ is FALSE ($20 < 12$). If one part of an AND expression is FALSE, the entire expression is FALSE (TRUE & FALSE = FALSE). So, R returns the result FALSE.</p>		
---	--	--

Let's check out an example of how you might use logical operators to analyze data. Imagine you are working with the "airquality" dataset that is preloaded in RStudio. It contains data on daily air quality measurements in New York from May to September of 1973.

The data frame has six columns: Ozone (the ozone measurement), Solar.R (the solar measurement), Wind (the wind measurement), Temp (the temperature in Fahrenheit), and the Month and Day of these measurements (each row represents a specific month and day combination).

	 Ozone 	Solar.R 	Wind 	Temp 	Month 	Day 
1	41	190	7.4	67	5	1
2	36	118	8.0	72	5	2
3	12	149	12.6	74	5	3
4	18	313	11.5	62	5	4

Let's go through how the AND, OR, and NOT operators might be helpful in this situation.

AND example

Imagine you want to specify rows that are extremely sunny and windy, which you define as having a Solar measurement of over 150 *and* a Wind measurement of over 10.

In R, you can express this logical statement as `Solar.R > 150 & Wind > 10`.

Only the rows where *both* of these conditions are true fulfill the criteria:

	Ozone	Solar.R	Wind	Temp	Month	Day
1	18	313	11.5	62	5	4

OR example

Next, imagine you want to specify rows where it's extremely sunny or it's extremely windy, which you define as having a Solar measurement of over 150 *or* a Wind measurement of over 10.

In R, you can express this logical statement as `Solar.R > 150 | Wind > 10`.

All the rows where *either* of these conditions are true fulfill the criteria:

	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	12	149	12.6	74	5	3
3	18	313	11.5	62	5	4

NOT example

Now, imagine you just want to focus on the weather measurements for days that are *not* the first day of the month.

In R, you can express this logical statement as `Day != 1`.

The rows where this condition is true fulfill the criteria:

	Ozone	Solar.R	Wind	Temp	Month	Day
1	36	118	8.0	72	5	2
2	12	149	12.6	74	5	3
3	18	313	11.5	62	5	4

Finally, imagine you want to focus on scenarios that are not extremely sunny and not extremely windy, based on your previous definitions of extremely sunny and extremely windy. In other words, the following statement should *not* be true: either a Solar measurement greater than 150 or a Wind measurement greater than 10.

Notice that this statement is the opposite of the OR statement used above. To express this statement in R, you can put an exclamation point (!) in front of the previous OR statement: `!(Solar.R > 150 | Wind > 10)`. R will apply the NOT operator to everything within the parentheses.

In this case, only one row fulfills the criteria:

	Ozone	Solar.R	Wind	Temp	Month	Day
1	36	118	8.0	72	5	2

Resources

To learn more about logical operators and conditional statements, check out the tutorial on [“Conditionals and Control Flow in R” on the DataCamp](#) website. DataCamp is a popular resource for people learning about computer programming. The tutorial is filled with useful examples of coding applications for logical operators and conditional statements (and relational operators), and offers a helpful overview of each topic and the connections between them.