Boolean Logic

The primitive data type `boolean` has two values: `true` and `false`. Boolean expressions are built using ***relational operators*** and ***conditional operators***.

Content Learning Objectives

After completing this activity, students should be able to:

- Recognize the value of developing process skills.
- Evaluate boolean expressions with relational operators (<, >, <=, >=, ==, !=).
- Explain the difference between assignment (=) and equality (==) operators.
- Evaluate boolean expressions that involve comparisons with &&, ||, and !.

Process Skill Goals

During the activity, students should make progress toward:

- Evaluating complex logic expressions based on operator precedence. (Critical Thinking)

Model 1 Relational Operators

Some IDEs (including DrJava) have an "Interactions" feature that allows you to enter a single line of code and see what it does. If you type an ***expression*** like 1 + 2 and leave off the semicolon, it will display the resulting ***value*** (i.e., 3). In the table below, predict what values will be displayed and identify the relational operator. The first four rows are completed for you.

Interactions	Value displayed	Relational operator
`int three = 3`	none	none
`int four = 4`	none	none
`System.out.println(four)`	4	none
`three > four`	false	>
`boolean isLarger = three > four`		
`System.out.println(isLarger)`		
`three == four`		
`three < four`		
`three <= four`		
`three = four`		
`three == four`		

Questions (15 min) Start time: _____

1. List the four unique ***boolean expressions*** used in Model 1.

2. Examine the fifth line of Java code in the above model.

   a) What three actions are performed in this single line of code?

   b) Write two lines of code, ending with semicolons, that would perform these same actions (but in two lines instead of a single line).

3. The `!=` operator means "not equals". Give an example of a boolean expression that uses `!=` and evaluates to false.

4. Explain why the same boolean expression `three == four` resulted with two different values in the table.

5. What is the difference between = and == in Java?

6. List the six relational operators that can be used in a boolean expression. (Five have been used so far, but you should be able to guess the sixth.) Explain briefly what each one means.

Model 2 Conditional Operators

Boolean expressions use conditional operators to implement basic logic. If all three operators appear in the same expression, Java will evaluate `!` first, then `&&`, and finally `||`. If there are multiples of the same operator, they are evaluated from left to right. Relational operators are evaluated before `&&` and `||`, so there is generally no need for parentheses.

Operator	Meaning
!	not
&&	and
||	or

Example Variables:

```
int a = 3;
int b = 4;
int c = 5;
boolean funny = true;
boolean weird = false;
```

Example Expressions:

```
a < b && funny
a < b && b < c
c < a || b < a
funny && a < c
!funny || weird
```

Questions (20 min)

Start time: _____

7. What are the values (true or false) of the example expressions?

8. Give different examples of boolean expressions that:

   a) uses a, b, and !, and evaluates to false

   b) uses b, c, and !, and evaluates to true

   c) uses any variables, but evaluates to false

   d) uses any variables, but evaluates to true

9. Using your answers from the previous question, write the boolean expression `p && q` where `p` is your answer to step a) and `q` is your answer to step b).

   a) Your expression:

   b) Result of `p && q`:

10. Complete the following table:

p	q	p && q	p || q	!p
false	false			
false	true			
true	false			
true	true			

11. Using the values in Model 2, give the result of each operator in the following expression. In other words, show your work as you evaluate the code in the same order that Java would.

```
!(a > c) && b > c
```

	Operator	Expression	Result
1st	>	`a > c`	false
2nd			
3rd			
4th			

12. Add parentheses to the boolean expression from the previous question so that the `&&` is evaluated before the `!`. Then remove any unnecessary parentheses.

a) Expression:

b) New result:

13. Review the table from #10 for evaluating `&&` and `||`. Looking only at the `p` and `&&` columns, when is it necessary to examine `q` to determine how `p && q` should be evaluated?

14. Review the table from #10 for evaluating `&&` and `||`. Looking only at the `p` and `||` columns, when is it necessary to examine `q` to determine how `p || q` should be evaluated?

15. In Java, `&&` and `||` are ***short circuit*** operators, meaning they evaluate only what is necessary. If the expression `p` is more likely to be true than the expression `q`, which one should you place on the left of each operator to avoid doing extra work?

a) left of the `&&` expression:

b) left of the `||` expression:

16. What is the result of the following expressions?

a) `1 + 0 > 0 && 1 / 0 > 0`

b) `1 + 0 > 0 || 1 / 0 > 0`