Model 1 Conditional Operators

Boolean expressions, like `written > problem` and `teamwork < 75.0`, can be combined using the ***conditional operators***:

Operator	Meaning
`!`	not
`&&`	and
`||`	or

For example, `written > problem && teamwork < 75.0` is false, because `teamwork` is not less than `75.0`. (Both conditions need to be true in order for `&&` to be true.)

The following table summarizes the result of `&&`, `||`, and `!` for all possible inputs. The variables `p` and `q` represent conditions like `written > problem` and `teamwork < 75.0`.

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

Questions (20 min) Start time:

1. Consider the following variables:

```
double initiative = 74.2;
double analytical = 71.9;
double workEthic  = 70.8;
boolean hired = true;
boolean fired = false;
```

What are the results (true or false) of the following expressions?

Expression	Result
`!fired`	true
`!(workEthic < initiative)`	false
`workEthic < 71.0 && 71.0 < initiative`	true
`initiative < 70.0 || workEthic > 70.0`	true
`fired || workEthic < 50.0`	false
`analytical < initiative && fired`	false
`hired && !fired`	true

2. Write a boolean expression that . . .

a) uses `initiative`, `analytical`, and `!`, and evaluates to false. !(analytical < initiative)

b) uses `analytical`, `workEthic`, and `!`, and evaluates to true. !(workEthic > analytical)

c) uses any variable(s), and evaluates to false. fired

d) uses any variable(s), and evaluates to true. hired

3. Using your answers to the previous question, write a boolean expression "`p && q`" where `p` is your answer to part a) and `q` is your answer to part b).

a) Your expression: `!(analytical < initiative) && !(workEthic > analytical)`

b) Result of `p && q`: false (no matter what)

Relational operators (`<`, `>`, and `==`) are evaluated before conditional operators (`!`, `&&`, and `||`). When multiple conditional operators are used, Java evaluates `!` first, then `&&`, and finally `||`.

4. Show the intermediate result of each operator below. In other words, show your work as you evaluate the code in the same order that Java would.

```
!(initiative < analytical) && workEthic > analytical
```

	Operator	Expression	Result
1st	`<`	`initiative < analytical`	false
2nd	`!`	`!false`	true
3rd	`>`	`workEthic > analytical`	false
4th	`&&`	`true && false`	false

5. Change the parentheses in the original expression (from the previous question) so that the `&&` is evaluated before the `!`. Then remove any unnecessary parentheses.

a) Expression: `!(initiative < analytical && workEthic > analytical)`

b) New result: true

6. Review the table from Model 1 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?

You only need to look at `q` when `p` is true. If `p` is false, you know the expression will be false.

7. Review the table from Model 1 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?

You only need to look at `q` when `p` is false. If `p` is true, you know the expression will be true.

8. 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: `q` — if it's false, then `p` won't be evaluated

b) left of the `||` expression: `p` — if it's true, then `q` won't be evaluated

9. What is the result of the following expressions?

a) `1 + 0 > 0 && 1 / 0 > 0` java.lang.ArithmeticException: / by zero

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