Section 1: Operators

Relational operators are:

Equal to	Not equal to	Less than	Less than or equal to	Greater than	Greater than or equal to
==	!=	<	<=	>	>=

These can be used to ask questions about variables, like "is x equal to y?", or "is n less than 10?".

Logical operators are:

Not (negation)	AND	OR
!	&&	||

These are used to combine questions together, such as "is x greater than 0 and is x less than 10?". Both "x > 0" and "x < 10" each result in true or false, and by combining them together we can ask if both statements are true at the same time, or not.

Exercise 1 — 25%

Translate the following questions into expressions using variables and operators.

(a) Is age less than 13?

(b) Does name1 and name2 have the same value?

(c) Does word1 come before word2 in the dictionary?
(Hint: a computer considers "a" to be less than "b"!)

(d) Is age less than 21 and is age greater than 18?

(e) Is x between 5 and 10?
(Hint: These have to be written as two separate expressions and combined with a logical operator)

(f) Is color red, or is color blue?

Section 2: Boolean Expressions

Boolean Expressions are statements that result in either true or false. These can be questions like "Is x less than 10?", or "Is the printer on?", or "Is age between 18 and 21?". These are essentially what you wrote in Exercise 1.

Boolean expression are what if statements and while loops operate on, in order to figure out whether to execute their internal code blocks!

A boolean expression can be created from a simple boolean variable:

```
bool savedDocument = true;
if ( savedDocument ) { ... }
```

Or from an expression using relational operators:

```
if ( score != 100 ) { ... }
if ( score > 69 ) { ... }
```

And we can combine multiple boolean expressions together using logical operators:

```
if ( name == "admin" || name == "administrator" ) { ... }
```

Exercise 2 **25%**

Figure out whether the following boolean expressions result in **true** or **false**.

(a) isDone is true	`( isDone )`	True	False
(b) isDone is true	`( !isDone )`	True	False
(c) age is 20	`( age >= 18 && age < 21 )`	True	False
(d) x is 0	`( x > 0 && x < 10 )`	True	False
(e) color is "cyan"	`( color != "green" )`	True	False

Exercise 3 **25%**

Suppose that we're tossing coins and recording the result as a boolean: isHeads. Therefore, when we flip a coin, if it lands on heads, then `pennyHeads = true`,
and if it lands on tails, then `pennyHeads = false`.

(a) We are just tossing one penny. List out all possible outcomes. How many total outcomes are there?

(b) We are tossing one penny and one dime. The variables will be `pennyHeads` and `dimeHeads`. List out all possible outcomes. How many total outcomes are there?

(c) We are tossing one penny, one dime, and one nickel. The variables will be `pennyHeads`, `dimeHeads`, and `nickelHeads`. List all possible outcomes. How many total outcomes are there?

(d) Given the following boolean expression:

```
( pennyHeads && dimeHeads && nickelHeads )
```

There is only one way to get a value of **true**. What do the values of each of the variables need to be?

pennyHeads	dimeHeads	nickelHeads

(e) Given the same boolean expression:

```
( pennyHeads && dimeHeads && nickelHeads )
```

There are four different ways for the result to be **false**. List them all.

pennyHeads	dimeHeads	nickelHeads

(f) Given the following boolean expression:

```
( pennyHeads || dimeHeads )
```

There are three ways we can get a result of **true**. List them all.

pennyHeads	DimeHeads

(g) Given the following boolean expression:

```
( pennyHeads || dimeHeads )
```

There is only one way to get a value of **false**. What do the values of each of the variables need to be?

pennyHeads	DimeHeads

Section 3: Truth Tables

We can actually model the result of a boolean expression with truth tables. Truth tables list out all possible combinations of variable values, and all possible results for those values.

For example, if we have two variables, *p* and *q*, and we're asking *"p and q?"*, we will get the following table:

p	*q*		*p* && *q*
True	**True**		**True**
True	False		False
False	True		False
False	False		False

And for the question *"p or q?"*, we will get this table:

p	*q*		*p* || *q*
True	**True**		**True**
True	**False**		**True**
False	**True**		**True**
False	False		False

This will always be the case for && and || statements…:

- **If we're asking *p && q && r && … && z*, in order for the expression to be true, all boolean variables must be true.**
- **If we're asking *p && q && r && … && z*, in order for the expression to be false, at least one boolean variable must be false.**
- **If we're asking *p || q || r || … || z*, in order for the expression to be true, at least one boolean variable must be true.**
- **If we're asking *p || q || r || … || z*, in order for the expression to be false, all boolean variables must be false.**

Exercise 4 **25%**

(a) The variable *g* stands for "x is greater than 0", and the variable *l* stands for "x is less than 10". The result of each of these questions is either true or false. Fill out the following truth-table to find all possible outcomes of asking "*is x > 0 and x < 10?*"

`x > 0` `g`	`x < 10` `l`		`Is x > 0 and x < 10?` `g && l`
True	True		
True	False		
False	True		
False	False		

(b) The variable *k* stands for "person is a kid: age is less than 13", and the variable *s* stands for "person is a senior: age is greater than or equal to 60." In order to give a discounted ticket price, the person needs to either be a kid or a senior. Fill out the following truth-table to find all possible outcomes of asking, "*is the person's age < 13, or is their age >= 60?*"

`age < 13` `k`	`age >= 60` `s`		`age < 13 or age >= 60?` `k || s`
True	True		
True	False		
False	True		
False	False		