Lab 04: Booleans, predicates, conditionals

Create a separate file for each question. Keep them in your "Labs" folder, with the name **`l`**`ij`**`q`**`k` for **`Lab`** `ij`, **`Question`** `k`.

Download the headers for each function from the file labinterface04`.rkt`.

After you have completed a question (except class exercises), including creating tests for it, you can obtain feedback by submitting it and requesting a public test. Follow the instructions given in the Style Guide.

Language level: Beginning Student

1. *[Class exercise with lab instructor assistance]*

Create a function *two-multiples* that consumes three numbers, *target*, *candidate1*, and *candidate2*, and determines whether *target* is a multiple of both candidates. Your function should produce `"both"` if it is multiples of both, `"neither"` if it is a multiple of neither, and the value of the candidate if it is the multiple of one of the two. If any of the three numbers is a non-integer, your function should produce `false`.

For example:

```
(two-multiples 10 2 5) => "both"
(two-multiples 10 2 7) => 2
(two-multiples 10 9 7) => "neither"
(two-multiples 88 0.5 8) => false
```

2. Create the following 3 functions (i) *in-subset-1?*, (ii) *in-subset-2?*, and (iii) *in-subset-3?* respectively, that consume a number, x, and produce true if x is in the subset and false if it is outside the subset:

   i) $(3 < x \leq 7)$

   ii) the union of $(1 < x < 3)$ and $(9 < x < 11)$

   iii) the range of numbers outside of $(1 \leq x \leq 3)$

   For example:

```
(in-subset-1? 5) => true
(in-subset-2? 10) => true
(in-subset-3? 10) => true
```

3. Consider an auction where the rules are such that each new bid must be at least 5% higher than the current high bid. For example, if the current high bid is $100, then the next bid must be at least $105. Create a function *acceptable-bid?* that consumes two positive numbers (*current-high* and *next-bid*) and produces `true` if *next-bid* includes an increase of at least 5% when compared to *current-high*, and `false` otherwise. Do not use conditional statements.

   For example:
   ```
   (acceptable-bid? 100 104) => false
   (acceptable-bid? 3 4) => true
   ```

4. Create a function *new-string* that consumes three strings, *original*, *add-on*, and *position*, and produces the string *original* followed by *add-on* if *position* is `"after"`, and the string *add-on* followed by *original* if *position* is `"before"`, and produces *original* for any other value of *position*.

   For example:
   ```
   (new-string "bat" "man" "after") => "batman"
   (new-string "bat" "man" "before") => "manbat"
   ```

5. Create the predicate *connect?* that consumes two non-empty strings, *string1* and *string2*, and determines if the last letter in *string1* is the same as the first letter in *string2*.

   For example:
   ```
   (connect? "abc" "c") => true
   (connect? "hi" "hello") => false
   ```

Helpful tips

Switching between `true` and `#true`

When choosing your language, select "Show Details", and choose the preferred "Constant Style". You will need to repeat this every time you change a language level in the future.

Highlighted unused code

After you have run your program, any unused part of the code will be highlighted. This either means that you have parts of the code that are not needed (and should be removed) or that you need to add more tests.