Carleton University
Department of Systems and Computer Engineering
ECOR 1051 - Fundamentals of Engineering I

Lab 7 - Boolean Logic Puzzles

Objectives

- To gain experience developing short Python functions that use `if` statements and Boolean operators (`and`, `or` and `not`).

Learning outcomes: 2, 4, 5; Graduate attributes: 1.3, 5.3 (see the course outline)

Getting Started

Launch Wing 101.

As always, functions must be written with full docStrings and type annotations. The main script must now demonstrate automated testing for each of the functions. You may wish to copy-paste your `test_int()` function from the previous lab. It will be useful for automated testing of any function you write that returns an integer. For functions that return other data types, you can choose to:

1. Write another version of the test function with different argument types (example: test_float)
2. Simply write out the testing code repetitively (without any function).

Begin by creating a new file within Wing 101. **Save it as lab7.py.**

Exercise 1

When bakers get together for a party, they like to eat pastries. A bakers' party is successful when the number of pastries is between 40 and 60, inclusive. Unless it is the weekend, in which case there is no upper bound on the number of pastries. Use the function design recipe to develop a function named `bakers_party`. The function takes two arguments. The first argument is the number of pastries (an integer). The second argument is `True` if it's the weekend, `False` if the day is a weekday. The function returns `True` if a party with the given arguments is successful, otherwise it will return `False`.

Exercise 2

The squirrels in Palo Alto spend most of the day playing. In particular, they play if the temperature is between 60 and 90 degrees F (inclusive). Unless it is summer, then the upper limit is 100 degrees instead of 90. Use the function design recipe to develop a function named `squirrel_play`. This function takes two arguments. The first argument is a temperature (an integer). The second argument is a boolean value that specifies the season (`True` if it's summer, otherwise `False`.) The function returns `True` if the squirrels are playing, given the temperature and the season.; otherwise it returns `False`.

Exercise 3

Any fan of the late author Douglas Adams knows that 42 is a truly great number. Use the function design recipe to develop a function named `great_42`. This function takes two integer values, `a` and `b`. It returns `True` if either value is 42, or if their sum or difference is 42. Hint 1: the built-in function `abs(x)` computes the absolute value of `x`.

Exercise 4

Use the function design recipe to develop a function named `blackjack`. This function takes two positive integer values, `a` and `b`. The function returns whichever value is closest to 21 without being over 21. It returns 0 if both values are over 21. For example, if the values of `a` and `b` are 19 and 20, the function returns 20. If the values of `a` and `b` are 19 and 21, the function returns 21. If the values of `a` and `b` are 17 and 22, the function returns 17.

Exercise 5

Use the function design recipe to develop a function named `sum_uniques`. This function takes three integer values, `a`, `b` and c, and returns their sum. However, if one of the values is the same as another of the values, that value is not used when the sum is calculated. For example, if the values of `a`, `b` and c are 3, 2 and 3, respectively, the sum is 2 (the two 3's aren't used). If the values of `a`, `b` and c are 3, 3 and 3, the sum is 0 (the three 3's aren't used).

Final Exercise

In English class, you have likely gone through a draft-revise process – You write your essay; then you sit back, maybe even print out your essay on paper, and you review your essay, looking for run-on sentences, spelling mistakes, etc.

In coding, you should develop the same process – We call it a code review. In this lab, we are asking you to perform a code review of your own code; later, you will be reviewing the work of your peers.

Minimally, you should ensure that your code meets the posted marking rubrics for the labs. As well, for this lab, you are required to make (at least) the following two improvements.

1. For any of the values that you hard-coded (e.g. in Exercise 1, the numbers 40 and 60), replace these hard-coded values with named CONSTANTS.
2. Look at the bodies of all of your functions. Do any of them look like this?

   ```
   def some_function( … ):
        if  some_boolean_expression :
             return true
        else:
             return false
   ```

   The body contains only one if-statement – the if-portion returns true (or false) and the else-portion returns the opposite. This style of programming is correct but can be simplified and shortened so that the return statement returns the calculated Boolean value directly.

```
def some_function( … ):
     return some_boolean_expression :
```

Wrap Up

Ensure that your code meets the posted marking rubrics for the labs.

Submit file lab7.py.

You are required to keep a backup copy of (all) your work for the duration of the term.

Last edited: April 23, 2020