Carleton University
Department of Systems and Computer Engineering
ECOR 1041 - Fundamentals of Engineering I

Lab 8 - Boolean Logic Puzzles

Objectives

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

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 will need 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 must write another version of the test function with different argument types (example: test_float)

Begin by creating a new file within Wing 101. **Save it as lab8.py.**

Exercise 1

Toronto zoo needs a program for its admission booths. When visitors arrive at the booth to purchase their tickets, the worker uses this program to figure out how much to charge them. You will write this program.

- During the weekdays, the ticket price is $30.00.
- On the weekends, the ticket price is $40.00.
- Senior citizens (age ≥ 65) are given a 50% discount.
- The young visitors (age <= 12) are given 30% discount.

Use the function design recipe to develop a function named **ticket_price**. The function has two input parameters. The first one is the person's age. The second one is **True** if it is the weekend and **False** if the day is a weekday. The function returns the **ticket price** after applying the discount.

Exercise 2

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** as input parameters. It returns **True** if either value is 42, or if their sum or difference is 42. Otherwise, it returns **False**.

Exercise 3

Use the function design recipe to develop a function named `sort_integers`. This function takes 3 integers as input parameters. The function returns a string with the integers sorted in ascending order. For example, if the function is called with 19, 11, and 15, the function returns `'11, 15, 19’`.

Exercise 4

Use the function design recipe to develop a function named `gross_earnings`. This function computes the gross earnings of an employee. If the employee has worked more than 40 hours during their workweek, they should be paid 1.5 times their hourly wage for all extra hours.

The function has two input parameters: (1) the number of hours the employee has worked and (2) the hourly wage. The function returns the gross earnings of the employee.

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. To begin, in this lab, we are asking you to perform a code review of your own code.

Minimally, you should ensure that your code meets the posted marking rubrics for the labs and it does automated testing using the functions you defined: `test_int()`, `test_float()`, etc.. 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 30, 40, 50%, and 30%), 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.

- Make sure that you included your name and student number
- Check proper use constants (UPPER_CASE) and variables (lower_case) (There is a 10/100 deduction for misuse of UPPPER & lower case)
- Check the indents of the function bodies. (There is a 10/100 deduction for misuse of indentation)
- Check file organization: (1) imports, (2) CONSTANTS, (3) all function definitions; (4) Main Script (There is a 10/100 deduction for not organizing the file according to the instructions)
- Confirm that your filename matches exactly.
- Confirm that your .py script runs properly, otherwise, the TA will also assign a zero.
- Submit the file on Brightspace.

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

Last edited: January 31, 2022