Flow Control

Overview

In this lab, you'll write some Python code to perform conditional logic and iteration. The scenario will be based on processing day, month, and year values for a date.

Source folders

Student folder : `C:\PythonDev\Student\03-FlowControl`

Solution folder: `C:\PythonDev\Solutions\03-FlowControl`

Roadmap

There are 4 exercises in this lab, of which the last exercise is "if time permits". Here is a brief summary of the tasks you will perform in each exercise; more detailed instructions follow later:

1. Performing boolean operations
2. Using conditional logic
3. Using loops
4. Additional suggestions (if time permits)

Exercise 1: Performing boolean operations

In the *Student* folder, open the `dateprocessing.py` module in the editor. The module contains some simple starter code, to ask the user to enter a day, month, and year. You will add various bits of code in this lab, to manipulate the date values.

To start off, add some code to determine if the year is a leap year. A leap year is:
(evenly divisible by 4 *and not* evenly divisible by 100) *or*
(evenly divisible by 400)
Use the remainder operator (%) to help you out here.

Print a message to indicate whether the year is a leap year. Then run the program several times, to test that your "is-leap-year" algorithm works for various years.

Exercise 2: Using conditional logic

Validate the day, month, and year values. Output a single message saying whether the date is valid or not. Output it in the format dd/mm /yyyy.

Suggestions and requirements:

- The day must be 1…*daysInMonth*, where *daysInMonth* depends on the month and possibly the year. For example, there are 31 days in January, 28 or 29 days in February, and so on.
- The month must be 1…12.
- The year must be 0…2099 (let's say).

Exercise 3: Using loops

Add some code to display all the dates for a specific month (taking into account the number of days in that month, and whether or not it’s February in a leap year). For example, here’s the output for February in 2016 (which was a leap year):

- 1/2/2016
- 2/2/2016
- 3/2/2016
- 4/2/2016
- …
- 28/2/2016
- 29/2/2016

Exercise 4: Additional suggestions (if time permits)

- Improve your code from exercise 3, so that it outputs dates in a format such as 1 February 2016.

- When you're outputting a day, add a suffix such as *st*, *nd*, *rd*, or *th*. For example, 1st February 2016.