Richard Hayes Crowley

03/25/2021

CSC\_157\_Lab\_016\_QA

**(1) This particular project’s starter code uses the datetime library. With a function call having your own arguments, test the following user - defined function.**

**from datetime import datetime**

**def daysBetween(day1, day2) :**

**day1 = datetime.strptime(day1, "%m/%d/%Y")**

**day2 = datetime.strptime(day2, "%m/%d/%Y")**

**return abs((day2 - day1).days)**

Executing the following:

`print(daysBetween("03/25/2021", "04/15/2021"))`

Returns the correct number of days between today and three weeks from now: 21

**(2) Explain what is accomplished by the following code segment.**

**import datetime**

**today = datetime.date.today()**

**targetDay = datetime.date(2019, 12, 25)**

**diff = abs(today - targetDay)**

**print (diff.days)**

This code segment assigns two different datetime objects, Today, and TargetDay, and calculates the difference in days between the two. The function abs(today – targetDay) works because of the arithmetic methods defined in the datetime class. So, even though `today` and `targetDay` would read “yyyy-mm-dd” if you printed them as strings, mathematical operations can still be performed on these because of class methods defined within the datetime class.

**(3) Execute the following program segment, using today’s date.**

**import datetime**

**print ("Today's Date:", datetime.datetime.today())**

**date\_today = datetime.date.today()**

**print (date\_today)**

**print ("Current Year:", date\_today.year)**

**print ("Current Month:", date\_today.month)**

**print ("Month Name:", date\_today.strftime("%B"))**

**print ("Month\'s Day :", date\_today.day) print ("Weekday Name:", date\_today.strftime("%A"))**The code segment above returns:   
  
Today's Date: 2021-03-25 20:03:10.651776 2021-03-25 Current Year: 2021 Current Month: 3 Month Name: March Month's Day : 25 Weekday Name: Thursday

**(4) Explain how business intelligence is used in this project.**

Business intelligence is used in this project to analyze sales trends and present a prediction of future sales figures. I would not trust it, though, to make important company-wide decisions, since all I’m doing is finding the average between three sales periods and mulitplying it by 1.1.

**(5) What have you learned from performing and coding this lab assignment?**

While I could have created a procedural or purely functional program for this assignment, I decided to try and make an OOP app with a SalesPeriod class. I found that once I built out my classes properties and methods, I could easily create three different sales periods. I incorporated logic within the class’ initialization method to accumulate total sales (previous periods accumulated sales + this periods sales), all using getters that are agnostic of their instances.   
  
I believe my classes are overengineered for this assignment (do I really need enapsulation for all of my classes properties if I’m not mutating those properties after initializing, anyway?) I did some stack overflow searching and found some lively discussions around the “pythonic” way of writing classes. So, I really got some good pracitce in OOP in this lab.