BMGT 404 Homework 4 Python

What to turn in

Turn in your Python codes through ELMS by **10:00AM on Monday, March 27th**.

Learning objectives:

- File operations
- String operations
- Lists
- Dictionaries
- Conditional and loop statements
- Functions

You will write your Python codes to answer the following questions.

1) Create your own module (20 points)

Create one Python module: arithmetic which has two functions: add(x, y) and get_length(x). The description of two functions are below:

```
def add(x, y):
        # add two numbers

def get_length(x):
        # return the length of a given string
```

Then make another Python file: test.py where two functions defined in arithmetic will be imported and called using real arguments.

Note:

- You need to submit two separate Python files: arithmetic.py and test.py for this question.

2) Word count (30 points)

Use Python programming to count all unique words for all files in a given directory. The output must be words and their corresponding frequency.

Note:

- To get all files under a directory, use os.listdir(directory name)
- To check if a character is an alpha character, you can use the following example expression (3 is not a word character):
    - >>> str = “hello3hello”
    - >>> str = str.lower() # to be all lowercase first
    - >>> str[5] > ‘a’ and str[5] < ‘z’
    - >>> False
- To debug your code, please create a folder with at least 3 text files. You don’t need to submit your example folder and text files.

3) Top-selling products (30 points)

Use Python programming to find most popular single products and co-purchased products from the large transaction data: retail.txt. Each row in the file is one purchase transaction from a customer, including a set of product ids separated by spaces.

Note:

- Co-purchased products: a pair of products purchased in the same transaction.
- To find co-purchased product in each transaction, you might use a nested loop.
- Write top 10 single products and top 10 co-purchased products into a new file: output.txt

4) Try to well document your codes using comments (10 points)

In Python, there are two ways to annotate your code.

1) The first is to include comments that detail or indicate what a section of code – or snippet – does.
2) The second makes use of multi-line comments or paragraphs that serve as documentation for others reading your code.

Single-line comments are created simply by starting a line with a hash (#) character. It is automatically terminated by the end of line.
For example:

```
# This would be a comment in Python
```

Comments that span multiple lines – used to explain things in more detail. Simply use 3 single quotes before and after the part you want commented.
For example:

```
'''
 This would be a multiline comment in Python that spans several lines.
It describes your code, your day, or anything you want it to …

'''
```