09 Prove Milestone: Text Files

Purpose

Prove that you can write a Python program that reads CSV files and creates, populates, and uses dictionaries.

Problem Statement

A local grocery store subscribes to an online service that enables its customers to order groceries online. After a customer completes an order, the online service sends a CSV file that contains the customer's requests to the grocery store. The store needs you to write a program that reads the CSV file and prints to the terminal window a receipt that lists the purchased items and shows the subtotal, the sales tax amount, and the total.

Assignment

During this milestone, you will write half of this Python program. Specifically, by the end of this milestone, your program will read and process these two CSV files:

- The `products.csv` file is a catalog of all the products that the grocery store sells.
- The `request.csv` file contains the items ordered by a customer.

Helpful Documentation

- This article explains how to setup VS Code so that your Python program can read from files.
- The prepare content for this lesson shows how to read the contents of a CSV file into a dictionary and how to read and process a CSV file without storing it in a dictionary.
- The prepare content for lesson 8 explains how to find an item in a dictionary.
- This video shows a BYU-Idaho faculty member solving a problem that is similar to this prove assignment.

Steps

Do the following:

1. Download both of these files: `products.csv` and `request.csv` and save them into the same folder where you will save your Python program.
2. Open the `products.csv` file in VS Code and examine it. Notice that each row in this file contains three values separated by commas: a product number, a product name, and a retail price. Also, notice that each product number in the `products.csv` file is unique. This means that your program can read the `products.csv` file into a dictionary and use the product numbers as keys in the dictionary.
3. In VS Code, create a new file and save it as `receipt.py` in the same folder where you saved the `products.csv` and `request.csv` files.
4. In `receipt.py`, write a function named `read_products` that will open the `products.csv` file for reading and use a `csv.reader` to read each row and populate a dictionary named *products* with the contents of the `products.csv` file.

Recall that each item in a dictionary has a key and a value. Each item in the *products* dictionary must have a product number as the key and a list with the product name and price as the value as shown in the following table.

Products

Key	**Value**
"D150"	["1 gallon milk", 2.85]
"D083"	["1 cup yogurt", 0.75]
"P143"	["1 lb baby carrots", 1.39]
"W231"	["32 oz granola", 3.21]
"W112"	["wheat bread", 2.55]
"C013"	["twix candy bar", 0.85]
⋮	⋮

5. Open the `request.csv` file in VS Code and examine it. Notice that each row contains only two values, a product number and a quantity. Notice also that product number D083 appears twice in the file. It appears twice because the customer who created the order in the `request.csv` file added four yogurts to his order and then later added three more yogurts to his order. Because product numbers may appear multiple times in the `request.csv` file, your program must not read the contents of `request.csv` into a dictionary.

6. In your `receipt.py` program, write another function named `main` that does the following:

    a. Calls the `read_products` function and stores the products dictionary in a variable named *products*.
    b. Prints the *products* dictionary.
    c. Opens the `request.csv` file for reading.
    d. Contains a loop that reads and processes each row from the `request.csv` file. Within the body of the loop, your program must do the following for each row:
        i. Use the requested product number to find the corresponding item in the *products* dictionary.
        ii. Print the product name, requested quantity, and product price.

    Because product number D083 appears twice in the `request.csv` file, your program must not read the `request.csv` file into a dictionary. Recall that each key in a dictionary is unique. If your program reads the `request.csv` file into a dictionary, when your program reads line 3 of the `request.csv` file, your program will put a request for four yogurts into the dictionary. Then when your program reads line 6 of the `request.csv` file, your program will replace the request for four yogurts with a request for three yogurts. In other words, if your program reads the `request.csv` file into a dictionary, your program will think that the customer ordered only three yogurts instead of the seven (4 + 3) that he ordered. Therefore, your program must not read the `request.csv` file into a dictionary but should instead read and process each row similar to example 3 in the prepare content for this lesson.

7. At the bottom of your `receipt.py` file, add a call to the `main` function. Be certain to protect the call to `main` with an `if` statement as taught in the prepare content for lesson 5.

Testing Procedure

Verify that your program works correctly by following each step in this testing procedure:

1. Download the `test_products.py` file and save it in the same folder where you saved your `receipt.py` program. Run the `test_products.py` file and ensure that the `test_read_products` function passes. If it doesn't pass, there is a mistake in your `read_products` function. Read the output from `pytest`, fix the mistake, and run the `test_products.py` file again until the test function passes.

```
> python test_products.py
====================== test session starts =======================
platform win32--Python 3.8.6, pytest-6.1.2, py-1.9.0, pluggy-0.13.
rootdir: C:\Users\cse111\lesson09
collected 1 item

test_products.py::test_read_products PASSED                 [100%]

======================= 1 passed in 0.12s ========================
```

2. Run your program and verify that it prints the *products* dictionary and requested items as shown in the sample output below.

```
> python receipt.py

Products
D150 ['1 gallon milk', 2.85]
D083 ['1 cup yogurt', 0.75]
D215 ['1 lb cheddar cheese', 3.35]
P019 ['iceberg lettuce', 1.15]
P020 ['green leaf lettuce', 1.79]
P021 ['butterhead lettuce', 1.83]
P025 ['8 oz arugula', 2.19]
P143 ['1 lb baby carrots', 1.39]
W231 ['32 oz granola', 3.21]
W112 ['wheat bread', 2.55]
C013 ['twix candy bar', 0.85]
H001 ['8 rolls toilet tissue', 6.45]
H014 ['facial tissue', 2.49]
H020 ['aluminum foil', 2.39]
H021 ['12 oz dish soap', 3.19]
H025 ['toilet cleaner', 4.5]

Requested Items
wheat bread: 2 @ 2.55
1 cup yogurt: 4 @ 0.75
32 oz granola: 1 @ 3.21
twix candy bar: 2 @ 0.85
1 cup yogurt: 3 @ 0.75
```

Submission

On or before the due date, return to I-Learn and report your progress on this milestone.