1 Overview

For homework 5, we will create a basic online storefront. We will utilize methods, loops, conditionals, and dictionaries. Comments in the starter code indicate where you can add your own code. You can also add additional code elsewhere.

2 Methods

Your homework submission must have exactly 7 functions as stated below. **Note:** any modifications made to a dictionary that was passed to a method are reflected back where the function was called from, therefore, the dictionary does not have to be returned.

1. **price_check**
    1. **Parameters**:
        - A string that represents the item to price check
        - A dictionary that represents the entire inventory
    2. **Return**: One float representing the price of the item
    3. **Description**: This function takes the name of an item and the dictionary that represents the inventory. If the item name is not in the dictionary, the function should return -1.0, otherwise, return the value of the item.
2. **checkout**
    1. **Parameters**:
        - One dictionary representing the user's cart.
        - One dictionary representing the inventory.
    2. **Returns**: One string representing a receipt for the user.
    3. **Description**: Given the user's cart and the dictionary that represents the price information, return a receipt style string for the user. It should be formatted such that the user is shown one type of item per line where each line starts with the quantity of the item, the item name, the cost per item, then finally the total price. The last two lines should be the subtotal for all items and a short statement of thanks.

```
7 Eggs at $2.07.....$14.49
5 Milk at $3.42.....$17.10
4 Cheese at $4.98.....$19.92
Your total is $51.51.
Thank you for shopping with us!
```

3. **show_qty**
    1. **Parameters**: One dictionary representing the store's inventory.
    2. **Returns:** none
    3. **Description:** This function will print the quantity of each item in the structure. This will print one item per line with the number of items followed by the name of the item. Each line should be prefaced with a tab.

```
7 of Eggs
5 of Milk
4 of Cheese
```

4. **show_prices**
    1. **Parameters**: One dictionary representing the store's inventory.

2. **Returns:** none
3. **Description:** This function will print the price of each item. This will print one item per line with the item name followed by its price. Each line should be prefaced with a tab.

```
Eggs.....$2.07
Milk.....$3.42
Cheese.....$4.98
```

5. **show_cart**
    1. **Parameters**: One dictionary representing the shopping cart.
    2. **Returns:** none
    3. **Description:** This function will print the name and quantity of each item in the cart. This will print one item per line with the item name followed by the number of that item in the cart (ex: **Eggs x2**). Each line should be prefaced with a tab.
6. **add_to_cart**
    1. **Parameters**:
        - One string representing the item to add to the cart
        - A dictionary that represents the shopping cart
        - A dictionary representing the store's inventory.
    2. **Return**: A Boolean value indicating whether the item was added to the cart
    3. **Description:** This function will take a string representing an item and the dictionary that we are trying to add it to. If the item is not currently in the cart, add it and set its initial value to 1, otherwise, increment its value by 1. Once the item has been added to the cart, the stores inventory should be updated to reflect that the item has been removed (i.e. decrease its quantity by one). Note that before adding the item to the cart, you should make sure that there is still some left in the inventory, if not, the method should return false.
7. **remove_from_cart**
    1. **Parameters**:
        - One string representing the item to remove from the cart
        - A dictionary that represents the shopping cart
        - A dictionary representing the store's inventory.
    2. **Return**: A Boolean value indicating whether the item was removed from the cart
    3. **Description:** This function will attempt to remove the given item from the shopping cart and update the store inventory accordingly. The method should update the number of the given item in the shopping cart. If the number of that item in the shopping cart is reduced to 0, then that item should be deleted from the cart completely. The method should return true if the item is successfully removed from the cart. Note that before adding the item to the cart, you should make sure that it exists in the cart, if not, the method should return false.
8. **main**
    1. **Parameters:** none
    2. **Returns:** none
    3. **Description:** This function controls the flow of our program. All of the options are discussed below

3 Functionality

There is one primary state in our program: shopping. Before the user begins shopping, the current prices of the inventory should be shown.

```
Here are our current prices:
    Eggs.....$2.07
    Milk.....$3.42
    Cheese.....$4.98
    Bread.....$2.72
    Ketchup.....$3.98
    Mustard.....$2.72
    Turkey.....$34.58
    Chicken.....$14.92
```

3.1 Shopping

This state allows our user to shop from the store. This state will continue to present the user options for using the store until the user checks out. Each of the options must correctly utilize the functions discussed in the previous section. Our user will have the following options:

```
Do you want to
    Check (P)rice
    (V)iew inventory
    (S)how cart
    (G)et item
    (R)eturn an item from your cart? or
    (C)heckout
Enter Choice:
```

- **Check Price:** The user enters a P or p in order to check the price of a specific item. When this choice is selected, the user will be prompted to enter the name of the item to check. If the item exists, the price is given.
- **View Inventory:** The user enters a V or v in order to view the number of each item in the store

```
Here is our current inventory:
    7 of Eggs
    5 of Milk
    Cheese: out of stock.
    7 of Bread
    5 of Ketchup
    2 of Mustard
    3 of Turkey
    4 of Chicken
```

- **Show Cart:** The user enters S or s in order to view what is in their shopping cart. This should show each item along with the number of that item in the user's cart. If nothing is in the cart, a message should be printed to indicate so.
    - **Before adding to the cart**

```
Enter Choice: s
There is nothing in your cart...
```

- **After adding to the cart**

```
Enter Choice: s

Here is your current cart:
    Mustard.....1
    Ketchup.....2
```

- **Get item:** The user enters G or g to select an item to put into their shopping cart. This should prompt the user to indicate which item they want. If that item exists and is in stock, it should be added to the cart and the store's inventory should be updated. Whether the action was successful or not, a message should be displayed to update the user.
- **Return item:** The user enters R or r to select an item from their cart to put back into the store's inventory. This should prompt the user to indicate which item they want to remove from their cart. If that item is in their cart, then it should be removed. If there is only one of these items in the cart, the item should be deleted completely from the cart; otherwise, only one of the items should be removed. Once removed from the cart, the store's inventory should be updated. Whether the action was successful or not, a message should be displayed to update the user.
- **Checkout:** The user enters C or c to check out and buy the items in their cart. A summary of each item and the total cost for them (as well as the total for the entire cart) should be printed and the user should stop shopping (i.e. the program should exit).

```
Enter Choice: c

    1 Eggs at $2.07.....$2.07
    2 Milk at $3.42.....$6.84
    1 Bread at $2.72.....$2.72
    3 Turkey at $34.58.....$103.74
Your total is $115.37.
Thank you for shopping with us!
```

4 Additional Guidelines

- Variables in your program must use the snake_case notation.
- You must use while loops and dictionaries in meaningful ways.
- Your program may not make use of any variable declared outside of a function. Likewise, the use of the "global" keyword is not allowed.
- Your program should have the structure:

```
def function_one():
    …
def function_two():
    …
…
def main():
```

```
…

main()
```

Calling ‘main()’ to start the program should be your last line

- Make sure you explain your code using comments (at minimum, each function should be documented). Also, make sure to have your name, section, and an overall program description in comments at the top of the file.
- Please be sure to make sure your program produces correct output, as well as including your name in the file name) before submitting to Canvas. **Programs that do not interpret with Python 3 will receive a grade of ZERO. NO EXCEPTIONS.**
    - By beginning this homework, you are agreeing to all syllabus policies governing this course, including this policy as well as the academic honesty policy (all work completed should be your own). **If you get assistance from a TA or professor, please indicate in comments in your code who helped you and what portion they assisted you with.**
- Name your python file lastName_firstName_sticks.py.

8pts Extra Credit: Add the capability to update the store inventory and the capability to update the product prices. If you attempt the extra credit, a loop should be added before the shopping loop (this will be referred to as the inventorying state) to allow the user to do the following:

1. During the inventorying state, the user can input ‘M’ or ‘m’ to modify the quantity of an item in the inventory.
2. During the inventorying state, the user can input ‘U’ or ‘u’ to update the price of an item.

5 Diagram of Program Flow (extra credit not shown here)