COSC 101 C Homework 6: Spring 2025

Introduction

In this assignment you will build a system that presents customers with a menu, takes orders from customers and prints the total bill at the end. This assignment is designed to give you practice with the following topics:

- List iteration
- List mutability

In addition, you will also get additional practice with accumulator patterns, conditionals and definite/indefinite iteration.

Important Tips

- Remember, computer science is a science. Always write code with a prediction in mind. While you can sparingly code to try and see output, you should focus on thinking through what you want your code to do, what you expect as a result, and compare what your code actually does to your result.
- Before you type, trace the execution of the starter code already provided
- Sometimes sub-tasks go together, like computation followed by printing. You should consider them simultaneously if that helps you follow the correctness of your work.
- Use extra steps if needed for your thought process, printing results to test your code. Just remember to remove these extra prints once you know your code is correct.
- Match your output to the output given. Precision is important in computer science.
- Don't change file names. You're generally given `.py` files to work in. Don't change the filenames of those files.

Your assignment

Your task is to complete following steps:

1. Download the `hw6.zip` file and open it. You will see one python file: `hw6_orders.py` in the unzipped folder.
2. Complete all of the functions in `hw5_orders.py`. Make sure to build on your bottom-up programming skills and test the functions as you go!
3. Submit your completed program.

Required functionality

- The main function in `hw6_orders.py` creates two lists called `menu` which has all the items available, and `prices` which has the corresponding prices for each item on the menu. The price for item at index `i` on the menu can be found at index `i` in prices.
- The program should prompt the user for their order, and dynamically maintain a list with their order. At the very end, it should print their entire order and the total price.
- At any given point, the user has four actions they can take: print the menu, print the current order, modify their existing order, finalize the order. The code should validate the user input such that the user is continuall prompted until only one of these actions is entered.
- If the user selects print menu or print order, these get printed to the screen.
- If the user selects modify order, they are faced with the following additional choices:
  - Whether to add or delete from the order.
  - The item to add or delete.
  - The quantity of the item to add or delete.
- If the user selects the modify order option but then changes their mind and wants to not make any changes, the program gives them a way to cancel the modification.
- If the user selects the finalize option, the program prints the entire order along with the total bill.

Example outputs

Example output 1

```
Welcome! Here is our menu
1. Pasta: $15
2. Rice bowl: $17
3. Tacos: $9
4. Pizza: $12
5. Soda: $3
6. Iced tea: $4

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: +
Enter item: 2
Enter quantity: 5
Added to order

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: po
Here is your current order!
* Rice bowl x 5

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: 2
Enter an action: 3
Enter an action: f
Enter an action: +
Enter item: 6
Enter quantity: 4
Added to order
```

```
What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: -
Enter item: 6
Enter quantity: 2
Deleted from order

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: f

Here is your final order
* Rice bowl x 5
* Iced tea x 2

Your total bill is: $93
```

Example output 2

```
Welcome! Here is our menu
1. Pasta: $15
2. Rice bowl: $17
3. Tacos: $9
4. Pizza: $12
5. Soda: $3
6. Iced tea: $4

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: pm
Here is the menu!
1. Pasta: $15
2. Rice bowl: $17
3. Tacos: $9
4. Pizza: $12
5. Soda: $3
6. Iced tea: $4

What would you like to do?
```

```
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: 1
Enter an action: asad
Enter an action: po
Here is your current order!

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: c

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: f

Here is your final order

Your total bill is: $0
```

Example output 3

```
Welcome! Here is our menu
1. Pasta: $15
2. Rice bowl: $17
3. Tacos: $9
4. Pizza: $12
5. Soda: $3
6. Iced tea: $4

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification
```

```
Enter an action: -
Enter item: 3
Enter quantity: 5
Deleted from order

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: +
Enter item: 4
Enter quantity: 1
Added to order

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: po
Here is your current order!
* Pizza x 1

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: m
How would you like to modify your order?
 * +: Add something
 * -: Delete something
 * c: No modification

Enter an action: -
Enter item: 4
Enter quantity: 2
Deleted from order

What would you like to do?
 * pm: Print the menu
 * po: Print your current order
 * m:  Modify your order
 * f:  Finalize your order

Enter an action: f
```

```
Here is your final order

Your total bill is: $0
```

Your tasks

Task A: Implement the building blocks

Using bottom up programming, implement the following core components of the wordle program.

- `print_menu`
- `print_order`
- `get_action`
- `get_modification`
- `get_item`
- `get_quantity`
- `modify_order`
- `print_total`

Make sure to test each of these functions as you write them!

Task B: Plan how to compose the functions together

Sketch out a plan for how you will compose the functions together. You can either draw this out on paper and submit an image, or type it up and submit a pdf.

Task C: Compose the core components

Using your sketch, put the core components you implemented together in the `main()` function.

Submission Instructions

Submit `hw6_orders.py` and your sketch (either as an image or a pdf) to the platform indicated in your class section. Remember to complete the questions at the top of the file before submitting!