

Assignment 4. Object Oriented Programming

Nerea Salamero Labara

March 7, 2025

1 Code

```
# *****
# Title:      Assignments 4
# Author:     Nerea Salamero Labara
# Date:       29/01/2025
# File:       assignment_4.py
# Subject:    Object Oriented Programming
# Description: This ShoppingList class has methods for adding items, removing items,
#              getting the count of unique items, getting the total units, and
#              displaying the current shopping list etc. Create a ShoppingList class
#              which has several methods item_count, add_item, unit_count etc. Here
#              is partially created shopping list class.
# *****

class ShoppingList:
    def __init__(self):
        self.items = {}          # Dictionary to store items and their quantities

    # Get the item name by using the index value
    def item(self, i: int):
        if len(self.items) > i:
            return list(self.items.keys())[i]

    # Add an item to the shopping list
    def add_item(self, item: str, unit: int):
        if item in self.items:
            self.items[item] += unit
        else:
            self.items[item] = unit

    # Remove a specified quantity of an item from the shopping list
    def remove_item(self, item, quantity: int):
        if item in self.items:
            self.items[item] -= quantity
            if self.items[item] <= 0:
                self.items.pop(item)

    # Get the total count of unique items on the shopping list
    def item_count(self):
        return len(self.items)

    # Get the total count of all units (quantities) of items on the shopping list
    def unit_count(self):
        return sum(self.items.values())
```

```

    # Display the current shopping list
    def display_list(self):
        print("Shopping List:")
        for item, unit in self.items.items():
            print(f"- {item}: {unit}")

# Test
shopping_list = ShoppingList()
shopping_list.add_item('Apple', 3)
shopping_list.add_item('Banana', 2)
shopping_list.add_item('Orange', 4)
shopping_list.display_list()

print(f"Total unique items: {shopping_list.item_count()}")    # Output: 3
print(f"Total units: {shopping_list.unit_count()}")          # Output: 9

shopping_list.remove_item('Banana', 1)
shopping_list.display_list()

# Display one item (Banana) by using the index
print(shopping_list.item(1))    # Output: Banana

```

```

36 Codium: Refactor | Explain
37 class ShoppingList:
38     Codium: Refactor | Explain | Generate Docstring | X
39     def __init__(self):
40         self.items = {} # Dictionary to store items and their quantities
41
42     # Get the item name by using the index value
43     Codium: Refactor | Explain | Generate Docstring | X
44     def item(self, i: int):
45         if len(self.items) > i:
46             return list(self.items.keys())[i]
47
48     # Add an item to the shopping list
49     Codium: Refactor | Explain | Generate Docstring | X
50     def add_item(self, item: str, unit: int):
51         if item in self.items:
52             self.items[item] += unit
53         else:
54             self.items[item] = unit
55
56     # Remove a specified quantity of an item from the shopping list
57     Codium: Refactor | Explain | Generate Docstring | X
58     def remove_item(self, item, quantity: int):
59         if item in self.items:
60             self.items[item] -= quantity
61             if self.items[item] <= 0:
62                 self.items.pop(item)
63
64     # Get the total count of unique items on the shopping list
65     Codium: Refactor | Explain | Generate Docstring | X
66     def item_count(self):
67         return len(self.items)
68
69     # Get the total count of all units (quantities) of items on the shopping list
70     Codium: Refactor | Explain | Generate Docstring | X
71     def unit_count(self):
72         return sum(self.items.values())
73
74     # Display the current shopping list
75     Codium: Refactor | Explain | Generate Docstring | X
76     def display_list(self):
77         print("Shopping List:")
78         for item, unit in self.items.items():
79             print(f"- {item}: {unit}")
80
81 # Test
82 shopping_list = ShoppingList()
83 shopping_list.add_item('Apple', 3)
84 shopping_list.add_item('Banana', 2)
85 shopping_list.add_item('Orange', 4)
86 shopping_list.display_list()
87
88 print(f"Total unique items: {shopping_list.item_count()}") # Output: 3
89 print(f"Total units: {shopping_list.unit_count()}") # Output: 9
90
91 shopping_list.remove_item('Banana', 1)
92 shopping_list.display_list()
93
94 # Display one item (Banana) by using the index
95 print(shopping_list.item(1)) # Output: Banana

```

Figure 1: Code

2 Output

After executing, the output obtained is the following one:

```
ea\Desktop\SAVONIA UAS\ObjectOrientedProgramming> & C:/Python312/python.exe "c:/Users/nerea/Desktop/SAVONIA UAS/ObjectOrientedProgramming/202502_assignment3y4/assignment3_4.py"
Shopping List:
- Apple: 3
- Banana: 2
- Orange: 4
Total unique items: 3
Total units: 9
Shopping List:
- Apple: 3
- Banana: 1
- Orange: 4
Banana
PS C:\Users\nerea\Desktop\SAVONIA UAS\ObjectOrientedProgramming>
```

Figure 2: Output obtained