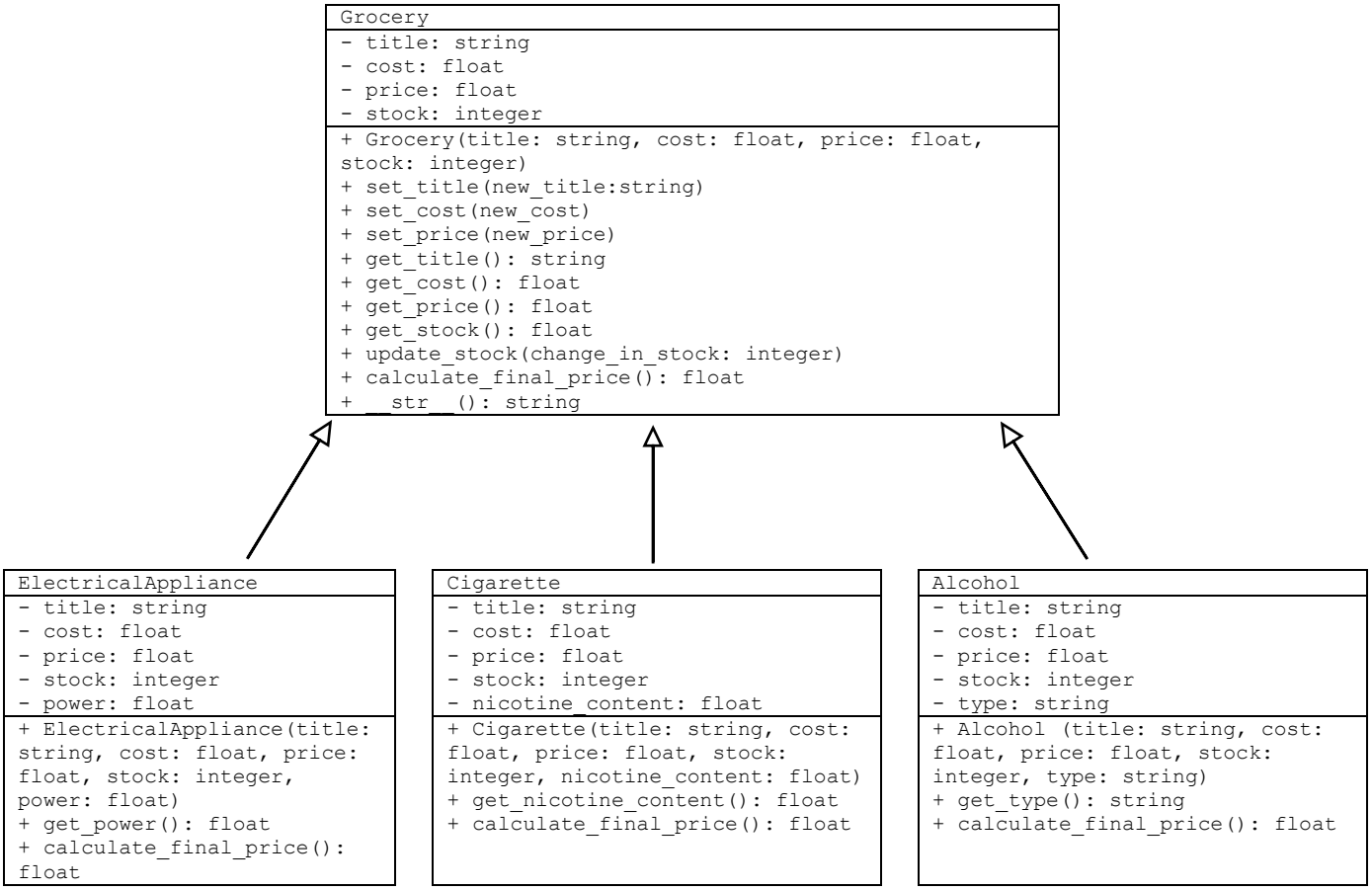
Grocery Store Manager

The grocery shop in the neighborhood ask your help to create an application to manage the grocery store.

First, you are tasked to work on an object-oriented solution to store all the grocery details. The `title` of the grocery item, `cost`, `price` and `stock` of each grocery is recorded. Besides the normal groceries, the shop identifies three unique types of grocery too, namely:

- Electrical Appliance: there is a need to indicate the `power` of the product to understand its energy consumption rate.
- Cigarette: it is important to track the `nicotine content` of various kinds of cigarette.
- Alcohol: there are distinct `types` of alcohols such as wine or beer.

Below is an UML class diagram for your reference.



Task 5.1
Implement the classes of `Grocery`, `ElectricalAppliance`, `Cigarette` and `Alcohol` with object-oriented programming based on the above UML class diagram.

Task 5.2
In country S, purchasing all groceries will incur a 7% Goods and Services Tax(GST).

To promote healthy living, additional tax have been imposed to cigarettes and alcohols:

- Cigarette: additional 60% tax
- Wines: additional 50% tax
- Beers: additional 20% tax

For example: the price of one packet of "Yun Yan" cigarette is $23.00, the final price can be calculated by: $23.00 x 160% x 107% = $39.38

In addition, to support the "save energy movement", all electrical appliance with a power less than or equals to 10Watt was set to be sold at 80% of its original price.

Implement the function `calculate_final_price()` which includes the above mentioned tax and promotion into consideration.

Implement the `__str__()` function which returns a string in the following format (You may refer to the `test_function_5_1()` to understand the formatting):

```
Title                |    Cost |   Price |  Stock |   Final Price
```

For example, Yun Yan's cost is $16.50, price is set at $23.00, the current stock is 4 and final price is $39.38. The __str__() function should return the following string:

```
Yun Yan              | $ 16.50 | $ 23.00 |      4 | $       39.38
```

Task 5.3
Implement a class `StoreManager` which keep track of a list of grocery items, `curr_item_list`. The `StoreManager` should have the following class functions:

Function	Description
`sell_item(sold_item)`	`sold_item` is a tuple containing 2 elements: the `title` of the item and the `quantity` sold. You may assume the title of item is always valid and the quantity sold is always smaller than the current stock. The function should decrease the current stock of the sold items. Upon completion, it should print out a string containing the following information: `Title | Unit Price | Quantity Sold | Subtotal` The function should return a float containing the `sub_total` value.
`sell_items(sold_item_list)`	`sold_item_list` is a list of tuples; each tuple containing the item `title` and `quantity` sold. The function should print out a table displaying information for all sold items in the following format: `Title | Unit Price | Quantity Sold | Subtotal` The summary should end with a line indicating the overall total value of items sold in this transaction.
`stock_check()`	When this function is called, it should check the list of all grocery items and print out a summary of items with current stock value below 5. This indicates the need for stocking up these items soon. A summary table should be printed in the following format: `Title | Unit Cost | Quantity Left`