APME 2018 – Assignment 01: Stock Manager

An investor buys and sells shares of stock in various companies listed in the stock market. What if the investor sells 120 shares of a company for $90 each at a time when the investor owns 150 shares, of which 50 shares were bought in January at $60 each, 50 more in February at $70 each, and 50 more in March at $80 each? Which of the shares have to be sold?

Tax laws require the investor to compute the profit (or loss) on a LIFO basis, i.e, the most recently purchased shares are sold first. So the profit on the sale is $500 on the 50 shares purchased in March, $1000 on the 50 shares purchased in February, and $600 (20 shares at $30 each profit) on shares purchased in January.

Therefore, an appropriate data structure for the investor is one stack of stock transactions for each of the companies that the investor owns stock in.

Your assignment is to implement a software “Stock Manager” for the investor. In doing so, you have to complete the class “StockManager” provided in the PyCharm code skeleton under the package “assignment01”.

The following requirements should be satisfied by your implementation:

- The investor buys and sells stock of 3 companies: “Google”, “Apple”, and “BMW”

Your “StockManager” allows the investor do to the following:

- buy_shares(company, number, buy_price) : buy shares of a company at a certain price
- sell_shares(company, number, sell_price) : sell shares of a company at a certain price
- sell_multiple(company_list, number_list, sell_price_list) : sell shares of different companies in one go, e.g., sell_multiple([‘Google’, ‘Apple’], [20, 10], [30, 67]) sells 20 shares of Google at $30 and 10 shares of Apple at $67.
- buy_multiple(company_list, number_list, sell_price_list) : buy shares of different companies in one go (parameters are set similarly to sell_multiple())
- All “sell” methods should return the profit/loss made by the investor in the current transaction(s)
- Note that you cannot sell stocks that you do not own!
- The software should keep track of the cumulative profit (or loss) made by the investor since the initialisation of the “StockManager”. This information can be queried at any time calling the method get_profit()

EXAMPLE

```
SM = StockManager()                    # Initialises new Stock Manager
SM.buy_shares(‘Google’, 10, 50)
SM.buy_shares(‘Apple’, 20, 30)
SM.buy_shares(‘Google’, 20, 40)
SM.buy_shares(‘Apple’, 20, 35)
print(SM.sell_shares(‘Apple’, 5, 30))   # Output: -25
SM.get_profit()                        # Output: Loss of $25 from stock manager initialisation
print(SM.sell_shares(‘Google’, 25, 60)) # Output: 450    (= 20*20 + 5*10)
SM.get_profit()                        # Output: Profit of $452 from stock manager initialisation
```

Instructions:

- You have to conduct this project **individually**

- You have to **submit the code** that you develop **on blackboard** by the **deadline of Thursday October 11 at 10pm**.
  Please zip the folder "assignment01" on your computer and upload the zip file.
  The submission area will be available in due time.

- You have to **give a demo** of your code. The exact date of the demo will be communicated later on.

  <u>You are not allowed to change your code between the submission deadline and the demo</u> (we will check!).
  "Giving a demo" means to show the functionality of your software. So, you have to develop appropriate code in the main() of your application to showcase the implemented functionality. Failing to demonstrates the implementation of (some of the) functionality will lead to point deductions.