CS256 Advanced Programming – Assign6

This assignment 6 is also the project 1.

Submit a compressed tar file **XXX_Assign6.tar.gz**, which include the following files:

1) **eval_postfix.py**
2) **capital_gain.py**

In each Python file, please include the following:

- function(s) you developed to solve the problem
- test code for using your function(s)

P-6.34 (50 points) Implement a program that can input an expression in postfix notation and **output** its value.

Postfix notation is an unambiguous way of writing an arithmetic expression without parentheses. It is defined so that if "(exp1)op(exp2)" is a normal, fully parenthesized expression whose operation is op, the postfix version of this is "pexp1 pexp2 op", where pexp1 is the postfix version of exp1 and pexp2 is the postfix version of exp2. The postfix version of a single number or variable is just that number or variable. For example, the postfix version of "((5+2) * (8-3))/4" is "5 2 + 8 3 - * 4 /".

Hint You will need to use a stack. Refer to the example of evaluating infix arithmetic expression in course slide of Section 6.1.

Note: Implement a function that accepts an input string with postfix expression, and **returns** a value of evaluating the postfix expression. If the input string is not a valid postfix string, raise an error. In your test code, asking users to input a postfix expression, in which operands and operators are separated by space. If the input is an **invalid postfix expression**, print some error message.

P-6.36 (50 point) When a share of common stock of some company is sold, the ***capital gain*** (or, sometimes, loss) is the difference between the share's selling price and the price originally paid to buy it. This rule is easy to understand for a single share, but if we sell multiple shares of stock bought over a long period of time, then we must identify the shares actually being sold. A standard accounting principle for identifying which shares of a stock were sold in such a case is to use a FIFO protocol—the shares sold are the ones that have been held the longest (indeed, this is the default method built into several personal finance software packages).

For example, suppose we buy 100 shares at $20 each on day 1, 20 shares at $24 on day 2, 200 shares at $36 on day 3, and then sell 150 shares on day 4 at $30 each. Then applying the FIFO protocol means that of the 150 shares sold, 100 were bought on day 1, 20 were bought on day 2, and 30 were bought on day 3. The capital gain in this case would therefore be 100*10+20*6+30*(−6), or $940.

Write a **program** that takes as input a sequence of transactions of the form "buy *x* share(s) at *y* each" or "sell *x* share(s) at *y* each," assuming that the transactions occur on consecutive days and the values *x* and *y* are integers.

Given this input sequence, the **output** should be the total capital gain (or loss) for the entire sequence, using the FIFO protocol to identify shares.
Hint Keep information about the purchase shares and prices in a queue, and then match those against sales. Care must be taken if only part of a purchase block is sold.

Note: Think about what type of queue you might need.
For this problem, you should provide a function that accepts a sequence of transactions. For each transaction, perform the corresponding operation (sell or buy). The function returns the capital gain after performing all the transactions. If the selling is larger than the current all shares purchased, just sell all that you have, and print a message "sell XXX<the number specified in the current sell transaction> is larger than all shares XXX<number of all shares>". Each time a sell is performed, update the capital gain.
In your test code, ask user to input a sequence of transactions, and print out the final capital gain. Use <Enter> to separate transactions, and use <Ctrl>+D to indicate the end of the input transactions.