Project 01

By Xingjian Yin, NetID as: xy179

##Problem 1:

A. A simple question using functions read_csv and calculate returns with .shift(1).

B. Same using np.log function.

```
Arithmetic Returns - Last 5 Rows:
                 SPY      AAPL      EQIX
Date
2024-12-27 -0.011492 -0.014678 -0.006966
2024-12-30 -0.012377 -0.014699 -0.008064
2024-12-31 -0.004603 -0.008493  0.006512
2025-01-02 -0.003422 -0.027671  0.000497
2025-01-03  0.011538 -0.003445  0.015745

Arithmetic Returns - Total Standard Deviation:
SPY     0.008077
AAPL    0.013483
EQIX    0.015361
dtype: float64

Log Returns - Last 5 Rows:
                 SPY      AAPL      EQIX
Date
2024-12-27 -0.011515 -0.014675 -0.006867
2024-12-30 -0.012410 -0.014696 -0.007972
2024-12-31 -0.004577 -0.008427  0.006602
2025-01-02 -0.003392 -0.027930  0.000613
2025-01-03  0.011494 -0.003356  0.015725

Log Returns - Total Standard Deviation:
SPY     0.008078
AAPL    0.013446
EQIX    0.015270

dtype: float64
```

##Problem 2:

A. Simple calculations where *shares.T @ portfolio.*

B. As below.

a. Exponentially weighted covariance sourced from Project01 (my own code).

After get the EWM using the previous function, simply use the new EWM matrix as the origin of stock variance and portfolio sigma.

```
            SPY      AAPL      EQIX
SPY   0.000072  0.000054  0.000052
AAPL  0.000054  0.000140  0.000038
EQIX  0.000052  0.000038  0.000153

                VaR  $  Expected Shortfall  $
Portfolio  3856.321669          4835.982950
SPY         827.848763          1038.155747
AAPL        946.076369          1186.417935
EQIX       2933.512216          3678.742668
```

---

b. Given a vector of SPY AAPL and EQIX returns, steps are as follows:
1) Fit the stocks each into T-distribution
2) Map the vector through the T-distribution CDF to (0,1)
3) Map the (0,1) using the normal quantile function
4) Using spearman rank correlation to get the correlation matrix
5) Using the correlation matrix to simulate.

```
              VaR 5% $  Expected Shortfall 5% $
Portfolio  4370.396952            6015.125106
SPY         776.069970           1029.625609
AAPL       1060.162802            1508.531772
EQIX       3394.449844            4774.627478
```

---

c. To simply take the historic data as an simulated result and get the lowest 5% percent from that result. Yet I was trying to make like 200 or 400 draws from the 500+ data as taught, but I don't know whether it is "using the full history", so answers may be slightly different.

	VaR 5% $	Expected Shortfall 5% $
Portfolio	4575.034060	6059.387076
SPY	872.403863	1080.104204
AAPL	1067.114956	1437.785272
EQIX	3635.077091	4714.893996

C. Method 1, EWM: It generates the least loss, maybe because it assumes a more ideal normal distribution condition. It calculates the covariance matrix using the EWM method, which is more sensitive to the recent data.

Method 2, Copulas: It generates a result similar to historical simulation.

Method 3, Historic Simulation: It simply uses history data without any modeling & parametres to estimate the var and risk.

##Problem 3

A. An easy problem using Black Scholes Merton. First to define the call-price function, then use the brentq function in scipy.optimization to approach the answer.

B. Still implement the formulas. And compare between the results from a vega * volatility_change and the actual calculation using BSM.

```
Implied volatility: 0.3351
Delta: 0.6659
Vega: 5.6407
Theta: -5.5446
Estimated price change: 0.0564
Actual price change: 0.0565
```

C. Of course it would hold. BSM naturally conform to Put-Call Parity.

```
Put-Call Parity Holds! (Diff: 0.000000)
```

D. Questions towards this problem: why implied volatility is hugely different from assumed annual volatility.

   d. I can't find exact formula on course slides. I scenario one case where there is only one underlying price: the stock price that could affect 3 assets. As for the theta decay, I simple add the theta of call option and put option as the portfolio theta, and multiply it by holding_days/trading_days.

   ```
   For Delta Normal VaR and ES
   VaR: 5.3951
   ES: 6.6030
   ```

   ---

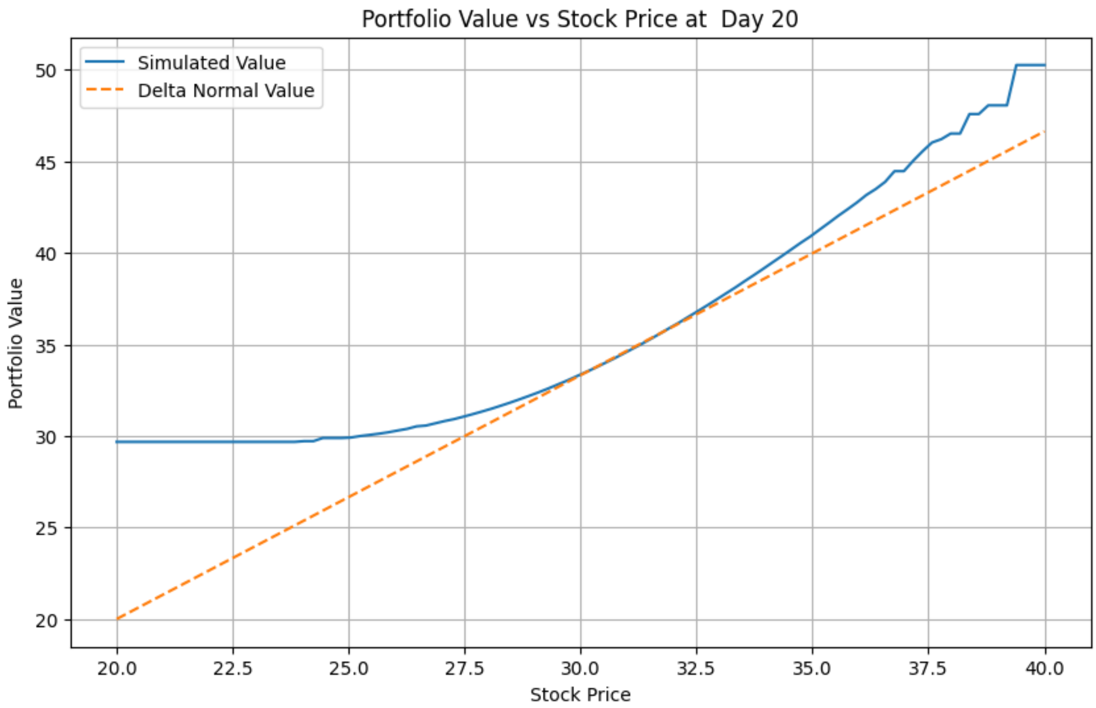
   e. I simulated a $(2000, 20)$ array of $(0, 20\%/\sqrt{255})$ normally-distributed data as the daily return. And then I calculate the portfolio value through the 20-day time, and conduct a 5% lowest among all these data to get the VaR and ES.

   ```
   For Historical Simulation VaR and ES

   VaR: 3.0989
   ```

```
ES: 3.7307
```

E. Comparison: Since there are time_laps in this question, so I think an appropriate method is to see the portfolio holding for a specific period. While using delta normal would arrive at a linear result, the Monte Carlo Simulation would arrive at an irregular curve that is due to random sampling.
For these two methods, I think the simulation is more acceptable because it is a dynamic way to modeling the portfolio value.



Codes are in relative ipynb files. Thanks for your reading.

Have a gooooood day !