Final exam
Shanglin Li
sl803
Hashed Name: 8afd6878a54824b8adaba8caf6c1f8bc7414708d
(Code file is a separate file than this)

Problem 1

a. First, we read the data in problem1.csv and check its format.
Then, we use return_calculate function in the library to calculate the retuns.
Add parameter 'LOG'to the function so that it would return log return.
We perform the log return by LOG(Pt/P(t-1))
Here is a head() look of the log return data:

	Price1	Price2	Price3
1	-0.035169	-0.000252	0.018486
2	-0.002239	-0.001236	0.000946
3	NaN	NaN	-0.026082
4	NaN	NaN	0.007248
5	0.049850	-0.003192	0.009794

b. We use the built in function in python to calculate the pairwise covariance matrix:

	Price1	Price2	Price3
Price1	0.000972	0.000029	0.000085
Price2	0.000029	0.000065	0.000018
Price3	0.000085	0.000018	0.000138

c. To find out if this matrix is PSD, we would like to find out the eigen values of the covariance matrix and if all eigen values are positive, we can say the matrix is PSD. We call our function is_psd() in our library with that algorithm. And it shows that the output is True, which means all eigen values are positive, and thus PSD.

d. For a data with missing values, we can
1. Omit it and calculate the pairwise data directly. But it may result in some imprecision when there are too much of them missing.
2. Fill the missing data with randomly generated numbers with certain algorithm like mean value between the missing values or using random normal generate with data's mean and standard deviation.

Problem 2

We can get the Underlying price, strike price of the call option, risk free rate, Time to maturity, dividend rate, implied volatility from problem2.csv, and Time to maturity need to be converted to a yearly unit (divide by 255). After plug in the parameters to the function we can get:

```
Call option price: 8.750112481033398
Delta: 0.6316672417005217
Gamma: 0.02236809662810111
Vega: 28.879866718413588
Rho: 33.000458844581445
```

f, g:

We Assume the Normal distribution of arithmetic returns where the implied volatility is the annual volatility and 0 mean, so we can generate a random normal series with mean=0 and sigma = implied volatility we have. Then we call our function in the library to calculate the VaR and ES with a normal assumption. Then we reached a result:

```
VaR: 0.3487798734998971
ES: 0.4433183588065814
```

h:

This portfolio performs longing a stock and short a call option, that it generally working like a covered call. The only difference is it is short a call but not writing a call option (Selling a call). It generate some profits back even if the underlying assets drop.

Problem 3

a. To calculate the optimal weights for max Sharpe ratio, we need to perform a minimize function in python:
    a) First construct the portfolio vol and portfolio return with the expected return, and covariance matrix.
    b) Construct function of negative Sharpe ratio with the formula (r-rf)/sigma
    c) Restrict the weights sum to be 1, we also need to add a bound for weight to make it greater or equal to 0.
    d) Using minimize function in scipy.stats to find out the optimal weight

```
Max sharpe ratio: 0.5218568887570968
Max Sharpe portfolio weight: [0.3507 0.2856 0.3638]
```

b. For this step, we will also apply the minimize function:
    a) Construct three function that calculate portfolio volatility, portfolio component Standard deviation (CSD), and sum of squares of CSD.
    b) Restrict the weights sum to be 1, make sure it is greater or equal to 0.
    c) Using minimize function to solve the risk parity portfolio

```
Risk parity portfolio: [0.3499 0.2857 0.3644]
```

c. We can see little difference of the weights between Sharpe and Risk parity. After we find out the risk contribution of each portfolio, we the attribution difference is tiny. So we can nearly say the optimal Sharpe portfolio is the risk parity portfolio.

Problem 4

a. The weights of assets will change after each change in returns
   a) The length of each returns is 20, so we will have a 20x3. Matrix record the weights of each time. (We have 20 different weights other than the initial weights)
   b) Each weight is a multiplication of (1+ return) of that row

```
Each weights:
[0.37322451 0.36249127 0.26428422]
[0.37146542 0.35617318 0.2723614 ]
[0.36805836 0.35487591 0.27706573]
[0.37158255 0.34868393 0.27973351]
[0.36486543 0.36154515 0.27358942]
[0.36314882 0.36273412 0.27411706]
[0.35964966 0.37607531 0.26427503]
[0.32247843 0.40632223 0.27119934]
[0.30039854 0.43209018 0.26751128]
[0.34501985 0.41201558 0.24296458]
[0.32642169 0.42464402 0.24893429]
[0.2962571  0.41730728 0.28643562]
[0.27838056 0.4334545  0.28816495]
[0.28523851 0.44108855 0.27367294]
[0.28239936 0.44396344 0.2736372 ]
[0.25841339 0.45866093 0.28292569]
[0.25227395 0.45940523 0.28832083]
[0.25122589 0.4725095  0.27626461]
[0.25576158 0.46172779 0.28251063]
[0.25290569 0.47663443 0.27045987]
```

b. We stored the portfolio returns of each iteration and then calculate the total return.
   Calculate the Carino K and scale it, then calculate the return attribution.

c. Continuing from last step, we calculate stock returns scaled by their weight at each time. Set up X with the portfolio return, calculate the Beta and discard the intercept. Finally, calculate component SD, which is Beta times the standard Deviation of the portfolio.

	Value	Asset1	Asset2	Asset3	Portfolio
0	TotalReturn	-0.203172	0.561752	0.243720	0.190415
1	Return Attribution	-0.078657	0.202951	0.066121	0.190415
2	Vol Attribution	0.009023	0.010447	0.005675	0.025145

Problem 5

1. We read the data from problem5.csv, use return_calculate function in the library to calculate the discrete returns for each asset
2. Construct the portfolio for price1 + price2 and price3 + price4, and the all assets in the portfolio. Add them as columns in the return Dataframe.
3. For each column, we perform a fitting of a generalized T distribution, we fit by the method of scipy.stats.t with MLE
4. Return the result of VAR at alpha=0.05 with VAR function in the library. And we get:

```
The VaR(5%) of Price1 is :0.0004793566590829447
The VaR(5%) of Price2 is :0.0005934543611220 74
The VaR(5%) of Price3 is :0.0007097287591203007
The VaR(5%) of Price4 is :0.0006221085515889815
The VaR(5%) of P1+2 is :0.0008771863553208672
The VaR(5%) of P3+4 is :0.0010904667355758422
The VaR(5%) of ALL is :0.0016366522425689315
```