```
!pip install -U finance-datareader
import FinanceDataReader as fdr
df_krx = fdr.StockListing('KRX')


kospi = fdr.DataReader('KS11', '2015-01-01', '2019-11-25')

삼성 = fdr.DataReader('005930', '2015-01-01', '2019-11-25')
한전 = fdr.DataReader('015760', '2015-01-01', '2019-11-25')
SKT = fdr.DataReader('017670', '2015-01-01', '2019-11-25')
포스코 = fdr.DataReader('005490', '2015-01-01', '2019-11-25')
현대차 = fdr.DataReader('005380', '2015-01-01', '2019-11-25')



import pandas as pd
df = pd.concat([kospi['Close'], 삼성['Close'], 한전['Close'],
                SKT['Close'], 포스코['Close'], 현대차['Close']], axis=1)
df.columns = ['KOSPI','SAMSUNG', 'KEPCO', 'SKT', 'POSCO', 'HMC']
df.head(3)
```

Date	KOSPI	SAMSUNG	KEPCO	SKT	POSCO	HMC
2015-01-02	1926.44	26600	42700	272500	283500	169000
2015-01-05	1915.75	26660	42000	276500	279000	168000
2015-01-06	1882.45	25900	41850	274500	275000	164500

```
df_rev = df.diff(axis = 0, periods = 1)/df.shift(periods=1)
df_rev.dropna(inplace=True)
df_rev.head(3)
```

Date	KOSPI	SAMSUNG	KEPCO	SKT	POSCO	HMC
2015-01-05	-0.005549	0.002256	-0.016393	0.014679	-0.015873	-0.005917
2015-01-06	-0.017382	-0.028507	-0.003571	-0.007233	-0.014337	-0.020833
2015-01-07	0.000733	0.009266	0.031063	-0.018215	0.016364	0.033435

과업 5개 기업의 포트폴리오 기대수익을 구하시오.

```
#평균수익률_행렬이용
import numpy as np
ds=np.array(df_rev.iloc[:,1:6])
ds.shape #(1202, 5)
exp = np.repeat(1,ds.shape[0])@ds/ds.shape[0]
exp
```

```
array([ 6.83988024e-04, -2.23216312e-04,  1.60454096e-05,  1.09404310e-05,
       -9.27239651e-05])
```

```
#평균수익률_함수이용(값이 같은 것을 확인하시오)
df=df_rev.iloc[:,1:6]
df.mean()
```

```
SAMSUNG    0.000684
KEPCO     -0.000223
SKT        0.000016
POSCO      0.000011
HMC       -0.000093
dtype: float64
```

```
#포트폴리오 정보
import numpy as np
pf = np.array([[0.2,0.2,0.2,0.2,0.2],
               [0.4, 0.1, 0.4, 0.1, 0],
```

```
              [0.1, 0.2, 0.1, 0.4, 0.2]])
pf@exp #각 포트폴리오별 기대 수익
```

```
array([7.90067175e-05, 2.58785785e-04, 1.11914604e-05])
```

과업 5개 기업의 포트폴리오 위험(분산)을 구하시오.

```
exp_repeat = np.tile(exp,(ds.shape[0],1))
exp_repeat.shape #(1205, 5)
```

```
(1202, 5)
```

```
#공분산_행렬이용
#공분산 Cov(X,Y) = E[(x-mu_x)*(y-mu_y)]
#                = E[A * B] 라 하자.
A = ds.T - exp_repeat.T # (5, 1205)
B = ds - exp_repeat # (1205, 5)
(A @ B) #(5, 5)
cov_matrix = (A @ B)/ds.shape[0]
cov_matrix
```

```
array([[ 2.58590486e-04, -7.08970300e-06,  7.17807881e-06,
          5.72825221e-05,  2.13345668e-05],
       [-7.08970300e-06,  2.65356728e-04,  3.97204310e-05,
          3.76114304e-05,  2.66223165e-05],
       [ 7.17807881e-06,  3.97204310e-05,  2.02542041e-04,
          1.58444762e-05,  1.28952550e-05],
       [ 5.72825221e-05,  3.76114304e-05,  1.58444762e-05,
          3.85671221e-04,  6.96873178e-05],
       [ 2.13345668e-05,  2.66223165e-05,  1.28952550e-05,
          6.96873178e-05,  3.16850895e-04]])
```

```
#공분산_함수이용(값이 같은 것을 확인하시오)
df.cov()
```

	SAMSUNG	KEPCO	SKT	POSCO	HMC
SAMSUNG	0.000259	-0.000007	0.000007	0.000057	0.000021
KEPCO	-0.000007	0.000266	0.000040	0.000038	0.000027
SKT	0.000007	0.000040	0.000203	0.000016	0.000013
POSCO	0.000057	0.000038	0.000016	0.000386	0.000070
HMC	0.000021	0.000027	0.000013	0.000070	0.000317

```
#포트폴리오 분산_행렬이용
pf[0]@cov_matrix@pf[0].T #p1의 분산
pf[1]@cov_matrix@pf[1].T #p2의 분산
pf[2]@cov_matrix@pf[2].T #p3의 분산

pf@cov_matrix@pf.T
```

```
array([[7.96473902e-05, 6.78044620e-05, 8.99817442e-05],
       [6.78044620e-05, 9.18013158e-05, 6.20431503e-05],
       [8.99817442e-05, 6.20431503e-05, 1.17572754e-04]])
```

```
#포트폴리오 분산_함수이용
ab = np.matmul(pf,np.array(df.cov()))
abc = np.matmul(ab,pf.T)
abc
```

```
array([[7.97137077e-05, 6.78609187e-05, 9.00566666e-05],
       [6.78609187e-05, 9.18777532e-05, 6.20948098e-05],
       [9.00566666e-05, 6.20948098e-05, 1.17670650e-04]])
```

/