In [1]:
```
import pandas as pd
import datetime as dt
import numpy as np
import yfinance as yf

pd.options
```

Out[1]:
```
<pandas._config.config.DictWrapper at 0x7fc2b844d910>
```

In [2]:
```
#https://www.learnpythonwithrune.org/calculate-the-market-sp-500-beta-with-python-for-any-stock/
#https://blog.devgenius.io/how-to-calculate-the-daily-returns-and-volatility-of-a-stock-with-python-d4e1de53e53b
#https://stackoverflow.com/questions/64506283/create-a-pandas-table
```

In [3]:
```
#Download all the stocks, benchmarks and define the period of download

stocks = ['MSFT', 'AAPL', 'AMZN', 'GOOG', 'NFLX', 'ACLS', 'TSLA']
benchmarks = [ 'SPY', 'IWM', 'DIA']
end_date = dt.datetime.today()
start_date = end_date - dt.timedelta(10*365)

adj_close = yf.download(stocks + benchmarks, start = start_date, end = end_date)['Adj Close']
```

```
[*********************100%***********************]  10 of 10 completed
```

In [4]:
```
#Create a Table
table_1 = pd.DataFrame(index = stocks)
```

In [5]:
```
#Calculate the Stock Returns Percentage Wise
stock_returns = adj_close.pct_change()
```

In [6]:
```
#Calculate the weights of the stocks in the portfolio
table_1['Weights'] = 1/len(stocks)
```

In [7]:
```
#Create New Column
table_1['Annualized Volatility'] = adj_close[-(21*3):].std()*np.sqrt(4)
```

In [8]:
```
table_1
```

Out[8]:

	Weights	Annualized Volatility
MSFT	0.142857	41.017917
AAPL	0.142857	21.039839
AMZN	0.142857	20.886994
GOOG	0.142857	16.497089
NFLX	0.142857	28.450138
ACLS	0.142857	14.921603
TSLA	0.142857	64.725308

In [9]:
```
#Calculate the Beta
beta = stock_returns[-252:].cov() / stock_returns[-252:].var()
```

In [10]:
```
#Create a For Loop
for bench in benchmarks:
    table_1[bench + '_Beta'] = beta[bench]
```

In [11]:
```
#Calculate the drawdowns

drawdown_max = adj_close[-252:].rolling(5).max()
drawdown_min = adj_close[-252:].rolling(5).min()
```

In [12]:
```
weekly_drawdown = drawdown_max - drawdown_min

table_1['AVG weekly drawdown'] = weekly_drawdown.mean()
table_1['Max weekly drawdown'] = weekly_drawdown.max()

table_1
```

Out[12]:

	Weights	Annualized Volatility	SPY_Beta	IWM_Beta	DIA_Beta	AVG weekly drawdown	Max weekly drawdown
MSFT	0.142857	41.017917	1.220078	0.858420	1.333755	12.346119	30.175964
AAPL	0.142857	21.039839	1.227404	0.893118	1.365886	7.049572	16.806656
AMZN	0.142857	20.886994	1.603952	1.216373	1.717253	8.496032	22.567993
GOOG	0.142857	16.497089	1.268150	0.913481	1.373195	5.669209	18.915497
NFLX	0.142857	28.450138	1.623277	1.296050	1.703970	25.850890	149.439972
ACLS	0.142857	14.921603	2.098224	1.847565	2.210747	5.883468	21.360001
TSLA	0.142857	64.725308	1.838506	1.548518	1.810769	26.461587	68.803345

In [13]:
```
#Calculate 10 year returns
table_1['10 Year Returns'] = adj_close.pct_change(len(adj_close)-1)[-1:].T *100
```

In [14]:
```
#Calculate the annualized return
table_1['Annualized 10 year Return'] = table_1['10 Year Returns'] ** (1 / np.sqrt(10))
```

In [15]:
```
table_1
```

Out[15]:

	Weights	Annualized Volatility	SPY_Beta	IWM_Beta	DIA_Beta	AVG weekly drawdown	Max weekly drawdown	10 Year Returns	Annualized 10 year Return
MSFT	0.142857	41.017917	1.220078	0.858420	1.333755	12.346119	30.175964	956.295738	8.760845
AAPL	0.142857	21.039839	1.227404	0.893118	1.365886	7.049572	16.806656	719.729283	8.007855
AMZN	0.142857	20.886994	1.603952	1.216373	1.717253	8.496032	22.567993	928.983657	8.680936
GOOG	0.142857	16.497089	1.268150	0.913481	1.373195	5.669209	18.915497	507.708084	7.171175
NFLX	0.142857	28.450138	1.623277	1.296050	1.703970	25.850890	149.439972	2395.141518	11.712196
ACLS	0.142857	14.921603	2.098224	1.847565	2.210747	5.883468	21.360001	1406.249981	9.897028
TSLA	0.142857	64.725308	1.838506	1.548518	1.810769	26.461587	68.803345	11164.666353	19.056416

In [16]:

```
#Transpose the table
table_1.T
```

Out[16]:

	MSFT	AAPL	AMZN	GOOG	NFLX	ACLS	TSLA
Weights	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
Annualized Volatility	41.017917	21.039839	20.886994	16.497089	28.450138	14.921603	64.725308
SPY_Beta	1.220078	1.227404	1.603952	1.268150	1.623277	2.098224	1.838506
IWM_Beta	0.858420	0.893118	1.216373	0.913481	1.296050	1.847565	1.548518
DIA_Beta	1.333755	1.365886	1.717253	1.373195	1.703970	2.210747	1.810769
AVG weekly drawdown	12.346119	7.049572	8.496032	5.669209	25.850890	5.883468	26.461587
Max weekly drawdown	30.175964	16.806656	22.567993	18.915497	149.439972	21.360001	68.803345
10 Year Returns	956.295738	719.729283	928.983657	507.708084	2395.141518	1406.249981	11164.666353
Annualized 10 year Return	8.760845	8.007855	8.680936	7.171175	11.712196	9.897028	19.056416

In [17]:
```
#Calculate the Equally Weighted Portfolio
adj_close['Equal Weight Port'] = adj_close[stocks].mean(axis=1)
stock_returns['Equal Weight Port'] = stock_returns[stocks].mean(axis=1)
```

In [18]:
```
stock_returns[benchmarks + ['Equal Weight Port']]
```

Out[18]:

	SPY	IWM	DIA	Equal Weight Port
Date				
2012-10-31	NaN	NaN	NaN	NaN
2012-11-01	0.010470	0.010535	0.010566	0.013821
2012-11-02	-0.008892	-0.015759	-0.009926	-0.006144
2012-11-05	0.002049	0.006651	0.001531	0.020920
2012-11-06	0.007825	0.007464	0.009016	-0.002615
...	...	...	...	...
2022-10-18	0.011750	0.011851	0.011289	0.000636
2022-10-19	-0.007086	-0.016995	-0.003666	0.016364
2022-10-20	-0.008385	-0.012733	-0.003417	-0.005821
2022-10-21	0.024301	0.021712	0.025573	0.036069
2022-10-24	0.012236	0.004285	0.013413	0.002381

2513 rows × 4 columns

In [19]:
```
#Create a New Table
table_2 = pd.DataFrame(index = benchmarks + ['Equal Weight Port'])
```

In [20]:
```
#Create Correlation column
table_2['Correlation'] = stock_returns[-252:][benchmarks + ['Equal Weight Port']].corr()['Equal Weight Port']
```

In [21]:
```
#Create Covariance Column
table_2['Covariance'] = (stock_returns[-252:][benchmarks + ['Equal Weight Port']]*100).cov()['Equal Weight Port']
```

In [22]:
```
#Tracking Error
table_2['Tracking Error'] = 0
for bench in benchmarks:
    table_2.loc[bench, 'Tracking Error'] = (stock_returns[bench] - stock_returns['Equal Weight Port']).std() * 10
```

In [23]:
```
#Sharpe Ratio
rf_rate = 0.0275
table_2['Sharpe Ratio'] = (((stock_returns[-252:][benchmarks + ['Equal Weight Port']]).mean()) - (rf_rate)) / (st
```

In [24]:
```
#Annualized Volatility Spread
table_2['Annual Vol Spread'] = (stock_returns[-252:]['Equal Weight Port']).std()-(stock_returns[-252:][benchmarks
```

In [25]:
```
#Transpose the table
table_2.T
```

Out[25]:

	SPY	IWM	DIA	Equal Weight Port
Correlation	0.894209	0.845653	0.786402	1.000000
Covariance	3.235825	3.672355	2.364415	6.289581
Tracking Error	1.063100	1.206519	1.219125	0.000000
Sharpe Ratio	-1.943755	-1.638635	-2.322723	-1.118261
Annual Vol Spread	0.010650	0.007763	0.013090	0.000000

In [26]:
```
#Correlation Matrix
corr_data = stock_returns[stocks+ benchmarks + ['Equal Weight Port']][1:].corr(method='pearson')

corr_data
```

Out[26]:

	MSFT	AAPL	AMZN	GOOG	NFLX	ACLS	TSLA	SPY	IWM	DIA	Equal Weight Port
MSFT	1.000000	0.613260	0.588699	0.684108	0.388515	0.442454	0.375217	0.770161	0.599787	0.708453	0.757477
AAPL	0.613260	1.000000	0.500533	0.559332	0.301247	0.403077	0.363871	0.699662	0.561934	0.638048	0.693905
AMZN	0.588699	0.500533	1.000000	0.625232	0.472452	0.366392	0.370340	0.605871	0.489099	0.506407	0.742729
GOOG	0.684108	0.559332	0.625232	1.000000	0.428707	0.441779	0.359611	0.727634	0.599434	0.650677	0.760439
NFLX	0.388515	0.301247	0.472452	0.428707	1.000000	0.295938	0.310704	0.418856	0.358826	0.350162	0.663522
ACLS	0.442454	0.403077	0.366392	0.441779	0.295938	1.000000	0.322200	0.549556	0.588614	0.495611	0.688429
TSLA	0.375217	0.363871	0.370340	0.359611	0.310704	0.322200	1.000000	0.433003	0.429081	0.369663	0.683718
SPY	0.770161	0.699662	0.605871	0.727634	0.418856	0.549556	0.433003	1.000000	0.880914	0.964822	0.801268
IWM	0.599787	0.561934	0.489099	0.599434	0.358826	0.588614	0.429081	0.880914	1.000000	0.853110	0.713236
DIA	0.708453	0.638048	0.506407	0.650677	0.350162	0.495611	0.369663	0.964822	0.853110	1.000000	0.705002
Equal Weight Port	0.757477	0.693905	0.742729	0.760439	0.663522	0.688429	0.683718	0.801268	0.713236	0.705002	1.000000