grab\_asset\_data.daily\_HLOCV(tick\_symbol, start, end):

tick\_symbol: the tick symbol string. e.g. enter ‘AAPL’ to grab apple stock data.

start: datetime object

end: datetime object

potfolio.Portfolio():

A class that represents a portfolio

self.stock\_symbols = stock\_symbols

self.share\_numbers = share\_numbers

self.portfolio\_value = 0

self.stock\_weights = []

\_\_init\_\_(self, stock\_symbols, share\_numbers):

stock\_symbols: an array of stock tick symbols(array of strings)

share\_numbers: an array of number of shares of all the stocks(array of integers)

portfolio\_value: the value of the portfolio(at a specified date)

portfolio\_beta: the beta of the portfolio(at a specified date)

portfolio\_sigma: the sigma of the portfolio(during a specified period)

stock\_values: an array of the values of all the stocks

stock\_betas: an array of the betas of all the stocks

stock\_sigmas: an array of sigmas of all the stocks

def calc\_portfolio\_value\_on\_date(self,date):

date: a datetime object

def calc\_portfolio\_value\_between(self,start,end):

**not implemented yet**

def calc\_portfolio\_beta(self, benchmark):

benchmark: the ticker symbol of the benchmark index

def value\_at\_risk(self):

**not implemented yet**

def visualize(self):

renders a pie chart showing the portfolio composition in terms of weight on each stock

stock.stock\_beta(stock\_symbol):

stock\_symbol: the ticker symbol of the stock for which you want to calculate the beta

benchmark: the ticker symbol of the benchmark index(‘^GSPC’ for S&P500)

stock.stock\_sigma(stock\_symbol, start,end):

stock\_symbol: the ticker symbol of the stock for which you want to calculate the beta