Advanced: PSET 1

Stocks

In this problem, you will be implementing a feature to determine pairs and triplets of stock positions you can buy given a budget to be maximised.

A. Starting Code

prices10.csv

```
Company Name,Price
"STR Holdings, Inc Common Stock",9
Chevron Corporation Common Stock,11
"Checkpoint Systms, Inc. Common Stock",2
Nuveen Intermediate Duration Municipal Term Fund Common Shares of Beneficial
Interest,6
"Juniper Networks, Inc. Common Stock",1
"Skilled Healthcare Group, Inc. Common Stock",12
Valero Energy Corporation Common Stock,7
"Summit Hotel Properties, Inc. Pfd Ser B",15
"Enbridge Energy, L.P. Class A Common Units",4
Eclipse Resources Corporation Common Stock,18
```

data taken from https://datahub.io/core/nyse-other-listings

Inside of the **prices** folder, you are given several csv files. These csv files contains rows of:

- **Company Name**: name of company
- **Price**: price of a single stock of that company

main

In **stocks.py**, the main function has been implemented for you to help you test your code. In this function, we:

1. Ensure that we take in a csv filename as command line argument 2 and budget as command line argument 3
2. We read from the csv file specified and populate our companies and prices list. Companies contain the names of companies and prices contain the price of that company's stock
3. We pass these arguments into the doubles & triples functions, which you will implement, and then print the result
4. To test the companies and prices from a particular csv with a specific budget, we may run something similar to : **`python stocks.py prices10.csv 20`**

B. Your Task

Implement the following functions:

`doubles`

```
def doubles(companies, prices, budget):
    # TO IMPLEMENT
    pass
```

1. `doubles` takes in:
   - `companies`: list of company names as **strings**
   - `prices`: list of prices of company stocks as **integers**
   - `budget`: budget given to maximise spending
   - All prices & company names are distinct

2. This function should:
   - Return a 2D list where each inner list contains **pairs of company names** whose stock prices sum up perfectly to **budget**
   - Make use of hash tables (python dicts)
   - Have worst case time complexity of **O(N)**, assuming hash table insertion and search has guaranteed **O(1)** complexity

Example:

```
companies = ["Apple", "Microsoft", "Amazon", "Netflix", "Instagram",
"Spotify"]
prices = [200, 175, 100, 125, 130, 170]
```

```
doubles(companies, prices, 300) ->
[["Apple", "Amazon"], ["Microsoft", "Netflix"], ["Instagram", "Spotify"]]
```

`triples`

```
def triples(companies, prices, budget):
    # TO IMPLEMENT
    pass
```

1. `triples` takes in:
   - `companies`: list of company names as **strings**
   - `prices`: list of prices of company stocks as **integers**
   - `budget`: budget given to maximise spending
   - All prices & company names are distinct

2. This function should:
   - Return a 2D list where each inner list contains **triplets of company names** whose stock prices sum up perfectly to **budget**
   - Make use of hash tables (python dicts)

- Have worst case time complexity of **O(N^2)**, assuming hash table insertion and search has guaranteed **O(1)** complexity

Hint: If you have implemented doubles in O(N) time complexity, think about how you may use your solution to doubles in triples to solve a sub problem!

Example:

```
companies = ["Apple", "Microsoft", "Amazon", "Netflix", "Instagram",
"Spotify"]
prices = [200, 175, 100, 125, 130, 170]
```

```
triples(companies, prices, 500) ->
[["Microsoft", "Netflix", "Apple"], ["Instagram", "Spotify", "Apple"]]
```

C. Sample Output

```
python stocks.py prices/prices10.csv 20
doubles result: [['STR Holdings, Inc Common Stock', 'Chevron Corporation
Common Stock'], ['Checkpoint Systms, Inc. Common Stock', 'Eclipse Resources
Corporation Common Stock']]

triples result: [['Valero Energy Corporation Common Stock', 'Enbridge Energy,
L.P. Class A Common Units', 'STR Holdings, Inc Common Stock'], ['Checkpoint
Systms, Inc. Common Stock', 'Valero Energy Corporation Common Stock',
'Chevron Corporation Common Stock'], ['Nuveen Intermediate Duration Municipal
Term Fund Common Shares of Beneficial Interest', 'Skilled Healthcare Group,
Inc. Common Stock', 'Checkpoint Systms, Inc. Common Stock'], ['Skilled
Healthcare Group, Inc. Common Stock', 'Valero Energy Corporation Common
Stock', 'Juniper Networks, Inc. Common Stock'], ['Summit Hotel Properties,
Inc. Pfd Ser B', 'Enbridge Energy, L.P. Class A Common Units', 'Juniper
Networks, Inc. Common Stock']]
```

D. Submission

To test your code, run the following command:

```
python utils/test.py
```

To submit your code, run the following command:

```
python utils/submit.py
```

A report.txt should be generated for you to view your results

Note: You may submit your work more than once. However, do take note that each new submission will overwrite the previous one, regardless of score. Therefore, do test your code before submitting!