Time Travelling Stonks Man

Background

It is the year 2037.
After making several poor stocks decisions in volatile stocks, you've lost most of your life savings.

Out of desperation, you recall a family heirloom stashed in a dusty corner of the storeroom that has been untouched for several decades.

Thinking that you could sell it for some cash, you brush off the dust to discover that its actually a strange never-before-seen contraption. After further investigation you realize that its a time machine capable of travelling into the past.

Pressing a dust-covered button, the machine lights up and asks you to select a specific year to travel to. From the corner of your eye, you notice that there is a battery icon, indicating the limited amount of energy remaining in the time machine.

In the heat of the moment, you decide to go back in time to purchase stocks. With the power of google, you know the historical stock prices and aim to make the most amount of money through buying and selling at the most optimal time across the years.

Scrounging up the rest of your savings, armed with historical stock prices and keeping in mind the remaining energy in the machine, you swear to return back to the year 2037 while earning the highest profits.

Challenge description

Travel back in time to maximize the amount of profits from buying and selling stocks in different years before returning back to 2037.

Requirement

- Expose a POST endpoint /stonks on your application

Rules

- You must return back to the year 2037
- Once a stock in a given year is bought, it is permanently gone
- To travel 1 year, it will cost the time travel machine 1 energy, and this scales linearly
    - Traveling 10 years will cost 10 energy

Sample

Input json (from challenge server)

```
[{
 "energy":2,
 "capital":500,
 "timeline":{
   "2037":{
     "Apple":{
       "price":100,
       "qty":10
     }
   },
   "2036":{
     "Apple":{
       "price":10,
       "qty":50
      }
   }
 }
},
{
<test case 2>
},...
]
```

```
Array containing json objects, with each json object corresponding to a respective test case
```

Assumptions

- Energy will always be a positive integer greater than 1
  - 1 < energy
- Capital will always be a positive integer greater than 0
  - 0 < capital
- Year will always be a positive integer that is less than or equal to 2037
  - 0 < year <= 2037
- Price of stock will always be a positive integer greater than 0
  - 0 < price
- qty of stock will always be a positive integer greater than or equal to 0
  - 0 <= qty

Output json (response from your endpoint)

```
[["j-2037-2036","b-Apple-50","j-2036-2037","s-Apple-50"],[<output for test case 2>],...]
```

Format of output

- Nested array, with each sub-array corresponding to a specific test case
- Each sub-array contains a list of actions to execute in sequence
- Each action is made up of 3 segments
  1. Action Type:
     i. **j** - Jump
     ii. **b** - Buy
     iii. **s** - Sell
  2. Year-From | Stock Name
  3. Year-To | Quantity of Stock

Expected response content type: **application/json**

Sample calculation of profits

1. j-2037-2036
   - **Jump** from year **2037** to year **2036**
2. b-Apple-50
   - **Buy 50 Apple** stocks at $10 = $500
3. j-2036-2037
   - **Jump** from year **2036** to year **2037**
4. s-Apple-50
   - **Sell 50 Apple** stocks at $100 = $5000

Profit: $5000 - $500 = **$4500**

Challenge created by Dion & Moses

For any questions or queries, feel free to contact us in the time-travelling-stonks-man channel in the discord server.

For feedback or improvements, you can email us at dion.ang@credit-suisse.com and moses.yong@credit-suisse.com