Code Challenge: Ganho de Capital

Contexto

O objetivo deste exercício é implementar um programa de linha de comando (CLI) que calcula o imposto a ser pago sobre lucros ou prejuízos de operações no mercado financeiro de ações.

Por favor, leia as instruções abaixo e sinta-se à vontade para fazer perguntas, caso ache necessário.

Exemplo de uso do Ganho de Capital

Como o programa deve funcionar?

Entrada

Seu programa vai receber listas, uma por linha, de operações do mercado financeiro de ações em formato `json` através da entrada padrão (`stdin`). Cada operação desta lista contém os seguintes campos:

Nome	Significado
`operation`	Se a operação é uma operação de compra (`buy`) ou venda (`sell`)
`unit-cost`	Preço unitário da ação em uma moeda com duas casas decimais
`quantity`	Quantidade de ações negociadas

Este é um exemplo da entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":20.00, "quantity": 5000}]
[{"operation":"buy", "unit-cost":20.00, "quantity": 10000},
{"operation":"sell", "unit-cost":10.00, "quantity": 5000}]
```

As operações estarão na ordem em que elas ocorreram, ou seja, a segunda operação na lista aconteceu depois da primeira e assim por diante.

Cada linha é uma simulação independente, seu programa não deve manter o estado obtido em uma linha para as outras.

A última linha da entrada será uma linha vazia.

Saída

Para cada linha da entrada, o programa deve retornar uma lista contendo o imposto pago para cada operação recebida. Os elementos desta lista devem estar codificados em formato `json` e a saída deve ser retornada através da saída padrão (`stdout`). O retorno é composto pelo seguinte campo:

Nome	Significado
`tax`	O valor do imposto pago em uma operação

Este é um exemplo da saída:

```
[{"tax":0.00}, {"tax":10000.00}]
[{"tax":0.00}, {"tax":0.00}]
```

A lista retornada pelo programa deve ter o mesmo tamanho da lista de operações processadas na entrada. Por exemplo, se foram processadas três operações (buy, buy, sell), o retorno do programa deve ser uma lista com três valores que representam o imposto pago em cada operação.

Regras do Ganho de Capital

O programa deve lidar com dois tipos de operações (`buy` e `sell`) e ele deve seguir as seguintes regras:

- O percentual de imposto pago é de 20% sobre o lucro obtido na operação. Ou seja, o imposto vai ser pago quando há uma operação de venda cujo preço é superior ao **preço médio ponderado** de compra.
- Para determinar se a operação resultou em lucro ou prejuízo, você pode calcular o **preço médio ponderado**, então quando você compra ações você deve recalcular o preço médio ponderado utilizando essa fórmula: `nova-media-ponderada = ((quantidade-de-acoes-atual * media-ponderada-atual) + (quantidade-de-acoes-compradas * valor-de-compra)) / (quantidade-de-acoes-atual + quantidade-de-acoes-compradas)` . Por exemplo, se você comprou 10 ações por R$ 20,00, vendeu 5, depois comprou outras 5 por R$ 10,00, a média ponderada é `((5 x 20.00) + (5 x 10.00)) / (5 + 5) = 15.00` .
- Você deve usar o prejuízo passado para deduzir múltiplos lucros futuros, até que todo prejuízo seja deduzido.
- Prejuízos acontecem quando você vende ações a um valor menor do que o preço médio ponderado de compra. Neste caso, nenhum imposto deve ser pago e você deve subtrair o prejuízo dos lucros seguintes, antes de calcular o imposto.
- Você não paga nenhum imposto se o valor total da operação (custo unitário da ação x quantidade) for menor ou igual a R$ 20000,00. Use o valor total da operação e não o lucro obtido para determinar se o imposto deve ou não ser pago. E não se esqueça de deduzir o prejuízo dos lucros seguintes.
- Nenhum imposto é pago em operações de compra.

Você pode assumir que nenhuma operação vai vender mais ações do que você tem naquele momento.

Exemplos do Ganho de Capital

Caso #1

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	100	0	Comprar ações não paga imposto

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`sell`	15.00	50	0	Valor total menor do que R$ 20000
`sell`	15.00	50	0	Valor total menor do que R$ 20000

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 100},
{"operation":"sell", "unit-cost":15.00, "quantity": 50},
{"operation":"sell", "unit-cost":15.00, "quantity": 50}]
```

Saída:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 0.00}]
```

Caso #2

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`sell`	20.00	5000	10000	Lucro de R$ 50000: 20% do lucro corresponde a R$ 10000 e não possui prejuízo anterior
`sell`	5.00	5000	0	Prejuízo de R$ 25000: não paga imposto

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":20.00, "quantity": 5000},
{"operation":"sell", "unit-cost":5.00, "quantity": 5000}]
```

Saída:

```
[{"tax": 0.00},{"tax": 10000.00},{"tax": 0.00}]
```

Case #1 + Case #2

Quando a aplicação recebe duas linhas, elas devem ser lidadas como duas simulações independentes. O programa não deve carregar o estado obtido do processamento da primeira entrada para as outras execuções.

Input:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 100},
{"operation":"sell", "unit-cost":15.00, "quantity": 50},
```

```
{"operation":"sell", "unit-cost":15.00, "quantity": 50}]
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":20.00, "quantity": 5000},
{"operation":"sell", "unit-cost":5.00, "quantity": 5000}]
```

Output:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 0.00}]
[{"tax": 0.00},{"tax": 10000.00},{"tax": 0.00}]
```

Caso #3

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`sell`	5.00	5000	0	Prejuízo de R$ 25000: não paga imposto
`sell`	20.00	3000	1000	Lucro de R$ 30000: Deve deduzir prejuízo de R$ 25000 e paga 20% de R$ 5000 em imposto (R$ 1000)

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":5.00, "quantity": 5000},
{"operation":"sell", "unit-cost":20.00, "quantity": 3000}]
```

Saída:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 1000.00}]
```

Caso #4

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`buy`	25.00	5000	0	Comprar ações não paga imposto
`sell`	15.00	10000	0	Considerando preço médio ponderado de R$ 15 ((10×10000 + 25×5000) ÷ 15000) não teve lucro nem prejuízo

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"buy", "unit-cost":25.00, "quantity": 5000},
{"operation":"sell", "unit-cost":15.00, "quantity": 10000}]
```

Saída:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 0.00}]
```

Caso #5

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`buy`	25.00	5000	0	Comprar ações não paga imposto
`sell`	15.00	10000	0	Considerando preço médio ponderado de R$ 15 não teve lucro nem prejuízo
`sell`	25.00	5000	10000	Considerando preço médio ponderado de R$ 15 lucro de R$ 50000: paga 20% de R$ 50000 em imposto (R$ 10000)

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"buy", "unit-cost":25.00, "quantity": 5000},
{"operation":"sell", "unit-cost":15.00, "quantity": 10000},
{"operation":"sell", "unit-cost":25.00, "quantity": 5000}]
```

Saída:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 0.00},{"tax": 10000.00}]
```

Caso #6

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`sell`	2.00	5000	0	Perda de R$ 40000: valor total é menor do que R$ 20000, mas devemos deduzir o prejuízo independe disso

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`sell`	20.00	2000	0	Lucro de R$ 20000: se deduzir o prejuízo, seu lucro é zero. Você ainda tem R$ 20000 de prejuízo para deduzir dos próximos lucros
`sell`	20.00	2000	0	Lucro de R$ 20000: se deduzir o prejuízo, seu lucro é zero. Agora não tem mais prejuízo para deduzir dos próximos lucros
`sell`	25.00	1000	3000	Lucro de R$ 15000 e sem prejuízos: paga 20% de R$ 15000 em imposto (R$ 3000)

Entrada:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":2.00, "quantity": 5000},
{"operation":"sell", "unit-cost":20.00, "quantity": 2000},
{"operation":"sell", "unit-cost":20.00, "quantity": 2000},
{"operation":"sell", "unit-cost":25.00, "quantity": 1000}]
```

Saída:

```
[{"tax": 0.00},{"tax": 0.00},{"tax": 0.00},{"tax": 0.00},{"tax": 3000.00}]
```

Case #7

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`sell`	2.00	5000	0	Prejuizo de R$ 40.000: valor total é menor que R$ 20000, mas deduzimos os prejuízos independente disso
`sell`	20.00	2000	0	Lucro de R$ 20000: se deduzir o prejuízo, seu lucro é zero. Você ainda tem R$ 20000 de prejuízo para deduzir dos próximos lucros
`sell`	20.00	2000	0	Lucro de R$ 20000: se deduzir o prejuízo, seu lucro é zero. Agora não tem mais prejuízo para deduzir dos próximos lucros
`sell`	25.00	1000	3000	Lucro de R$ 15000 e sem prejuízos: paga 20% de R$ 15000 em imposto (R$ 3000)

Operação	Custo unitário	Quantidade	Imposto Pago	Explicação
`buy`	20.00	10000	0	Todas as ações anteriores foram vendidas. Comprar novas ações muda a média ponderada para o valor pago por elas (R$ 20)
`sell`	15.00	5000	0	Prejuizo de R$ 25000
`sell`	30.00	4350	3700	Lucro de R$ 43500: se deduzir o prejuízo de R$ 25000, ficou restando R$ 18500 de lucro. Paga 20% de R$ 18500 em imposto (R$ 3700)
`sell`	30.00	650	0	Lucro de R$ 6500, sem prejuizo para deduzir, mas o valor total e menor que R$ 20000, então não paga imposto

Input:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":2.00, "quantity": 5000},
{"operation":"sell", "unit-cost":20.00, "quantity": 2000},
{"operation":"sell", "unit-cost":20.00, "quantity": 2000},
{"operation":"sell", "unit-cost":25.00, "quantity": 1000},
{"operation":"buy", "unit-cost":20.00, "quantity": 10000},
{"operation":"sell", "unit-cost":15.00, "quantity": 5000},
{"operation":"sell", "unit-cost":30.00, "quantity": 4350},
{"operation":"sell", "unit-cost":30.00, "quantity": 650}]
```

Output:

```
[{"tax":0.00}, {"tax":0.00}, {"tax":0.00}, {"tax":0.00}, {"tax":3000.00},
 {"tax":0.00}, {"tax":0.00}, {"tax":3700.00}, {"tax":0.00}]
```

Case #8

Operação	Custo unitário	Quantidade	Imposto	Explicação
`buy`	10.00	10000	0	Comprar ações não paga imposto
`sell`	50.00	10000	80000	Lucro de R$ 400.000: paga 20% de R$ 400.000 em imposto (R$ 80.000)
`buy`	20.00	10000	0	Comprar ações não paga imposto
`sell`	50.00	10000	60000	Lucro de R$ 300.000: paga 20% de R$ 300.000 em imposto (R$ 60.000)

Input:

```
[{"operation":"buy", "unit-cost":10.00, "quantity": 10000},
{"operation":"sell", "unit-cost":50.00, "quantity": 10000},
{"operation":"buy", "unit-cost":20.00, "quantity": 10000},
{"operation":"sell", "unit-cost":50.00, "quantity": 10000}]
```

Output:

```
[{"tax":0.00},{"tax":80000.00},{"tax":0.00},{"tax":60000.00}]
```

Estado da aplicação

O programa **não deve depender** de nenhum banco de dados externo e o estado interno da aplicação deve ser gerenciado em memória explicitamente por alguma estrutura que achar adequada. O estado da aplicação deve estar vazio sempre que a aplicação for inicializada.

Arredondando Decimais

Para numeros decimais, o programa deve arredondar os valores para a segunda casa decimal. Por exemplo:

Se houver a compra de 10 ações por R$ 20,00 e 5 ações por R$ 10,00, o preço médio ponderado é (10 x 20,00 + 5 x 10,00) / 15 = 16.67.

Lidando com erros

Você pode assumir que não ocorrerão erros na conversão do `json` de entrada. Na avaliação da sua solução nós não vamos utilizar entradas que contenham erros, estejam mal formatadas, ou que quebrem o contrato.

Nossas Expectativas

Nós no Nubank valorizamos as seguintes qualidades:

- **Simplicidade**: espera-se da solução um projeto pequeno e de fácil entendimento;
- **Elegância**: espera-se da solução facilidade de manutenção, uma separação clara das responsabilidades e uma estrutura de código bem organizada;
- **Operacional**: espera-se da solução a resolução do problema, seus casos de borda ou extremos e a capacidade de extensão para futuras decisões de design.

Desta forma, procuraremos avaliar:

- Uso adequado de trasparência referencial quando aplicável;
- Testes de unidade e integração de qualidade;
- Documentação onde for necessário;
- Instruções sobre como executar o código.

Por último, porém não menos importante:

- Você pode utilizar bibliotecas de código aberto (open source) que acredite serem adequadas para ajudar na solução do desafio, por exemplo analisadores de json; Por favor tente limitar o uso de frameworks e boilerplate code desnecessários.
- O desafio espera uma aplicação de linhas de comando **independente**; Por favor, evite adicionar infraestrutura desnecessária e/ou dependências externas. É esperado que você seja capaz de identificar as ferramentas necessárias para resolver o problema apresentado sem adicionar camadas extras de complexidade.

Notas gerais

- Esse desafio poderá ser estendido por você e por outra pessoa engenheira do Nubank durante uma outra etapa do processo;
- O Ganho de Capital deve receber as operações através da entrada padrão (`stdin`) e retornar o resultado do processamento através da saída padrão (`stdout`), ao invés de uma API REST.

Preparando seu desafio para envio

- Você deve entregar o código fonte de sua solução para nós em um arquivo comprimido (zip) contendo o código e toda documentação possível. Favor não incluir arquivos desnecessários como binários compilados, bibliotecas, etc.;
- Não faça o upload da sua solução em nenhum repositório público como GitHub, BitBucket, etc.;
- Se estiver builds conteinerizados, não faça o upload da sua imagem no em hubs públicos como DockerHub, Sloppy.io, etc.

Remova informações pessoais

⚠ **IMPORTANTE:** Por favor remova toda informação que possa lhe identificar nos arquivos do desafio antes de enviar a solução. Atenção especial para os seguintes pontos:

- Arquivos da solução como código, testes, namespaces, binários, comentários, e nomes dos arquivos;
- Comentários automáticos que seu editor de código pode ter adicionado aos arquivos;
- Documentação do código como annotations, metadata, e README.MD;
- Informações de autoria do código e configuração do versionador de código.

Se você planeja utilizar git como sistema de controle de versões, execute o seguinte comando na raíz do repositório para exportar a solução anonimizada:

```
git archive --format=zip --output=./capital-gains.zip HEAD
```

Adicione um README

Sua solução deve conter um arquivo de README com:

- Uma explicação sobre as decisões técnicas e arquiteturais do seu desafio;
- Uma justificativa para o uso de frameworks ou bibliotecas (caso sejam usadas);
- Instruções sobre como compilar e executar o projeto;

- Instruções sobre como executar os testes da solução;
- Notas adicionais que você considere importantes para a avaliação.

Ambiente de execução

O processo de build e execução da aplicação deve ser possível num sistema operacional Unix ou Mac. Builds conteinerizadas são bem vindas.

FAQ

> P: Como eu leio da entrada padrão (stdin)? Ela precisa estar em um arquivo tipo "input.txt"? Eu preciso pedir para o usuário colocar o nome do arquivo no terminal?

R: Esse é geralmente o tipo de leitura mais simples em qualquer aplicação de linha de comando, por exemplo Console.ReadLine() em C# ou input() em Python. Sua solução deve esperar que o usuário escreva cada linha no terminal e pressione 'enter'. Isso também permite que um arquivo seja passado à aplicação por Input Redirection. Por examplo:

```
./capital-gains < input.txt
```

Adicionalmente, nós não esperamos que a sua solução imprima qualquer explicação ao usuário de qual input esperado do tipo "Agora insira as operações de entrada:". Você pode assumir que o usuário sabe qual entrada seu programa espera e em qual ordem. A única saída esperada é o JSON com as respostas dos impostos.

> P: Pode haver um evento de compra após eventos de venda? Nesse caso, o preço médio de compra deve ser recalculado usando a nova compra?

R: Sim, o preço médio ponderado de compra deve sempre considerar todos os eventos de compra anteriores, até o evento de venda atual. Por favor, veja o Caso #7 para um exemplo prático.