Plano de Execução em 4 Dias - Desafio Ganho de Capital (Java)

Dia 1: Leitura de entrada e estrutura de dados

Objetivo: Preparar a base da aplicação CLI.

Tarefas:

- Criar o projeto Java (pode ser um projeto simples com javac ou com Maven/Gradle).
- Criar uma classe Operation com os campos: operation, unitCost, quantity.
- Criar função para ler da entrada padrão (System.in) e transformar cada linha em uma List<Operation> usando uma biblioteca JSON (como Jackson ou Gson).
- Garantir que múltiplas linhas possam ser lidas até encontrar uma linha vazia.
- Escrever testes unitários para o parser de JSON.

Resultado esperado: Você consegue ler várias linhas da entrada padrão, parsear cada uma em uma lista de operações.

Dia 2: Lógica de cálculo de imposto

Objetivo: Implementar a regra de cálculo de imposto de acordo com as regras do desafio.

Tarefas:

- Criar uma classe TaxCalculator com métodos:
  - Calcular preço médio ponderado de compra.
  - Calcular imposto baseado nas regras (lucro/prejuízo, limite de R$ 20.000, compensação de prejuízo).
- Manter estado interno por linha:
  - Quantidade total de ações.
  - Preço médio.
  - Prejuízo acumulado.
- Arredondar imposto para duas casas decimais com BigDecimal.

Resultado esperado: Dada uma List<Operation>, você consegue retornar uma lista com o imposto calculado para cada operação.

Plano de Execução em 4 Dias - Desafio Ganho de Capital (Java)

Dia 3: Saída em formato JSON e integração completa

Objetivo: Conectar leitura + cálculo + saída da aplicação.

Tarefas:

- Para cada linha da entrada:
  - Processar as operações.
  - Calcular os impostos.
  - Imprimir lista de { "tax": valor } em JSON no System.out.
- Usar uma biblioteca JSON para construir a saída (como Gson ou Jackson).
- Validar a saída com os exemplos do PDF.

Resultado esperado: O programa recebe múltiplas linhas de operações e imprime corretamente a lista de impostos em JSON.

Dia 4: Testes automatizados e empacotamento

Objetivo: Garantir qualidade e entregar uma solução profissional.

Tarefas:

- Criar testes unitários para TaxCalculator.
- Criar testes de integração simulando a entrada e checando a saída.
- Escrever o README.md com:
  - Instruções de execução.
  - Decisões técnicas.
  - Como rodar os testes.
- Criar script de build (ex: build.sh) ou Makefile, se desejar.
- Usar git archive para gerar capital-gains.zip.

Resultado esperado: Solução completa, testada, com README e pronta para ser enviada.