PROJETO PRÁTICO – PROGRAMAÇÃO ORIENTADA A OBJETOS

Objetivos:

- Elaborar um programa que cumpra os pré-requisitos enumerados no enunciado.
- Comentários no código-fonte são encorajados e alvo de avaliação.
- Otimização é alvo de avaliação.
- **É obrigatório o programa executar** (sem erros de compilação).
- Deve ser usada linguagem Java para a resolução do presente projeto prático.
- Deve ser entregue na tarefa do Microsoft Teams. Pasta do Projeto comprimida (.zip) com o nome TP_POO_(nome_formando)

EXERCÍCIOS

Após ter terminado o curso de **Software Developer**, foi contactado por um estúdio de videojogos para desenvolver o novo Simulador de Corridas. Este jogo trata-se de pilotar um carro que deve percorrer uma pista com curvas e obstáculos. O objetivo final é bater o recorde da pista mais famosa do mundo e tornar-se um piloto lendário. Para isso:

- Crie uma classe abstrata "**Veiculo**" com os atributos:
    - marca (String)
    - modelo (String)
    - potenciaCV (int)
    - pesoKg (double)
    - desgaste (int)
    - preco (int)
  - Crie o método mostrarDetalhes que escreva na consola todos os detalhes do veículo.

- Crie as suas subclasses **Carro** e **Mota.**
- A classe **Carro** deverá ter atributos:
    - tipoCarro (TipoCarro)
    - ArrayList<Modificacao> kitCorrida

- A classe **Mota** deverá ter os atributos:
    - ArrayList<Habilidade> habilidadesMota

- Desenvolva a enumeração **TipoCarro**, podendo ter os seguintes valores:
    - F1
    - Rally
    - GT

- Desenvolva a classe abstrata **ItemCorrida** que tem como atributos:
    - nome (String)
    - preço em fichas de corrida (int)

- Desenvolva a subclasse de ItemCorrida: **Modificacao** que tem como atributos:
    - nome (String)
    - preço em fichas de corrida (int)
    - diminuicaoDesgaste(int)
    - diminuicaoPeso(double)
    - ArrayList<String> carrosPermitidos que irá guardar o tipo de carros que podem/sabem usar dada reparacao.
  - Desenvolva também o método mostrarDetalhes.

- Desenvolva a subclasse de ItemCorrida**: Habilidade** que tem como atributos:
    - aumentoPotencia(int)
  - Desenvolva também o método mostrarDetalhes.

- Crie a classe **Oficina** que terá atributos:
    - ArrayList<Veiculo> garagem, que representa os veículos que o Piloto poderá comprar durante o seu jogo. Sejam carros ou motas.
    - ArrayList<ItemCorrida> stock, que representa os itens que o Piloto poderá comprar durante o seu jogo. Sejam Modificações ou Habilidades.
  - Deve conter o método imprimirStock que imprime aleatoriamente 6 itens em stock, assim como as suas especificações. Mesmo que a oficina tenha um stock maior, apenas 6 devem ser mostrados de forma aleatória.
  - Deve conter o método imprimirGaragem que imprime aleatoriamente 12 veiculos em stock, assim como as suas especificações. Mesmo que a oficina tenha uma garagem maior, apenas 12 devem ser mostrados de forma aleatória.
  - Deve conter o método venderItem( ) que recebe o Piloto como parâmetro, e verifica se a compra pode ser efetuada, caso tal compra seja possível, deve acrescentar ao inventário do VeiculoAtual do Piloto o item, e decrementar as suas fichas de corrida.
  - Deve conter o método venderVeiculo( ) que recebe o Piloto como parâmetro, e verifica se a compra pode ser efetuada, caso tal compra seja possível, deve alterar o Veiculo do Piloto, e decrementar as suas fichas de corrida.

- Crie uma classe "**Pista**" com os atributos:
    - nome (String)
    - tempo (double)
    - tempoRecordeSeg (double)
    - distanciaVoltaM (double)
    - quantidadeVoltas (int)
    - ArrayList<Momento> momentosPista

- Crie uma classe "**Momento**" com os atributos:
    - nome (String)
    - atrasoPeso (double)
    - atrasoPotencia (double)

  Estes atributos de atraso vão representar um atraso que o Veiculo irá sofrer no determinado momento da pista, por exemplo, Curva em U, Curva em S, gravilha na pista, borracha na pista…
  Alguns momentos vão favorecer Veículos mais leves, outros momentos vão favorecer veículos mais potentes.

- Crie uma classe "**Piloto**" com os atributos:
    - nome (String)
    - fichasCorrida (int)
    - veiculoAtual (Veiculo)
    - vitorias (int)

- Deve ter o método usarItem( ) que imprime o inventário de itens do VeiculoAtual e pergunta qual quer usar, seguidamente aplica os efeitos no Veiculo do Piloto.
- Deve implementar o método corrida( ) que recebe como parâmetro uma pista e retorna o tempo total que o piloto demorou a percorrer a pista no seu Veiculo, de acordo com as seguintes regras:
    - Calcula o tempo de cada volta de acordo com a seguinte fórmula:
        - distanciaVolta/((1.6*potencia))-(0.2*peso)-(0.5*desgaste))
        - A este tempo, somar o tempo que os Momentos atrasaram. Sendo que a cada volta, todos os momentos são repetidos de novo. Para calcular o atraso usar:
          ((peso*atrasoPeso)+(potencia*atrasoPotencia))/100
          A cada volta, o desgaste do veículo aumenta em 20.

- Elabore uma Classe **Jogo** com um método **criarPiloto** que permita criar um piloto através de feedback da consola (pode criar métodos auxiliares nas respetivas classes).
    - A seguir é também atribuído fichas de corrida à personagem, se a dificuldade for fácil tem direito a 2000 moedas de ouro, se for difícil apenas a 1500 moedas de ouro. (exemplo demonstrativo, pode alterar os valores e ainda acrescentar outros graus de dificuldade).

Na Classe Jogo crie o método (nome do jogo), como por exemplo, corridasRapidas( ), simuladorCorridas( ) ou aventuraNaPista( ). Recebe o Piloto criado como parâmetro. Este será o método de "jogo", ou seja, terá o percurso a percorrer pelo piloto.

- Assim, voltando ao método (nome do jogo): Instancie, no mínimo, 12 itens e 16 veiculos, os quais serão adicionados a uma instância de Oficina.
- Deve instanciar, sem conhecimento do utilizador, as pistas e os seus momentos.
- Devem também ser instanciadas e adicionadas ao Array os momentos. E adicione os respetivos momentos a cada pista.
- Seguidamente na oficinaInicial do início de campeonato, terá veículos à venda, a primeira ação é obrigar o jogador a comprar um Veículo.

- Posteriormente, no final de cada corrida a oficina será de uma segunda instância e deve ter itensCorrida e Veiculos. Sendo que quando o piloto acaba a corrida, pode comprar itens e veiculos.

- O campeonato deve ter, pelo menos 4 pistas e, no mínimo, 2 momentos em cada pista, diferentes.

- No final de cada corrida, o piloto pode também usar itensCorrida num máximo de 2.

- Se o piloto bater o recorde da pista, recebe uma quantidade de fichas de corrida, caso não bata o recorde, recebe metade dessas fichas.

- O objetivo do jogo é, no final, apresentar a derradeira pista, que o piloto tem obrigatoriamente de bater o recorde. Sendo assim o objetivo do jogo é gerir fichas de corrida e comprar veiculos e itensCorrida melhores para poder competir com os melhores nesta última pista que deve ter, no mínimo, 5000m. por volta, 5 voltas e 4 momentos.
- Pode implementar métodos ao seu critério, para além dos acima mencionados.
- Ponto de Valorização: Carregar o ArrayList da Oficina através de ficheiro.
- Deve gerar o **JAVADOC**

Bom trabalho! ☺