ATIVIDADE PRÁTICA – POO

Modelagem de Classes e Construção do CRUD

Instruções gerais:

Esta Lista de Exercícios é Opcional e não será entregue no Canvas

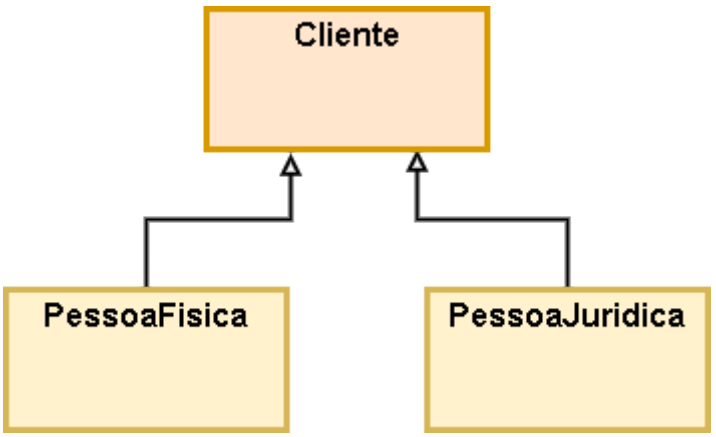
EXERCÍCIO

Boas práticas:

1. Leia o enunciado do exercício com atenção
2. Observe as indicações de Entrada e Saída esperadas em cada exercício
3. Observe com atenção os desenhos e diagramas inseridos nos exercícios para facilitar a compreensão
4. Utilize o Cookbook, os Vídeos da Plataforma e os Códigos guia como referências para a resolução do exercício
5. Caso ainda fique alguma dúvida, consulte os instrutores da sua turma pelo Discord

Atividade 01

Dado o Diagrama de Classes abaixo, construa uma aplicação que contenha um Menu principal, que forneça todas as operações do CRUD para um sistema de controle de Clientes:



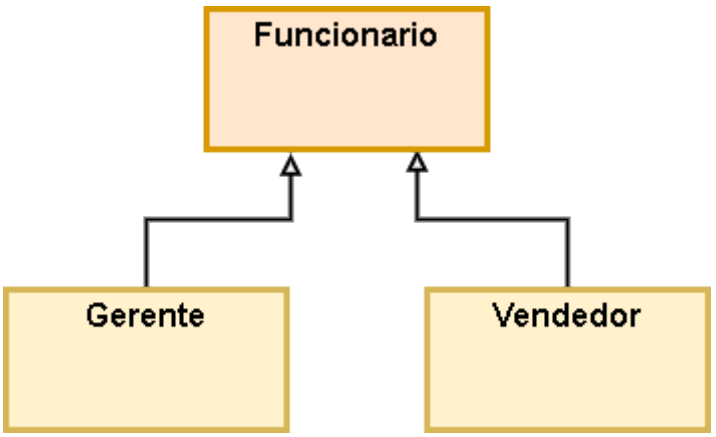
Boas Práticas:

1. Construa o projeto utilizando a Linguagem TypeScript.
2. Utilize as tipagens do TypeScript na construção das variáveis, objetos e métodos
3. Crie o Menu principal na Classe Menu
4. Construa as 3 Classes Model, conforme o Diagrama de Classes acima
5. A Classe Cliente será a Super Classe da aplicação e será definida como Abstrata.
6. Defina pelo menos 4 atributos além do id do cliente.
7. As Classes PessoaFisica e PessoaJuridica serão Sub Classes da Classe Cliente (Heranças)
8. Defina pelo menos 1 atributo relevante em cada Sub Classe.
9. Construa a Interface ClienteRepository contendo os Métodos do CRUD:
    a. Criar()
    b. Listar Todos()
    c. Consultar por Id()
    d. Atualizar()
    e. Deletar()
10. Construa a Classe ClienteController, implementando os Métodos da Interface ClienteRepository

11. Crie as respectivas entradas de dados para os Métodos da Classe ClienteController no Menu
12. Os Clientes deverão ser armazenados em uma Collection Array
13. Envie o projeto para um repositório no seu GitHub.
14. **Desafio:** *Crie um Método adicional para calcular a idade do cliente com base na data de nascimento.*

Atividade 02

Dado o Diagrama de Classes abaixo, construa uma aplicação que contenha um Menu principal, que forneça todas as operações do CRUD para um sistema de controle de RH:



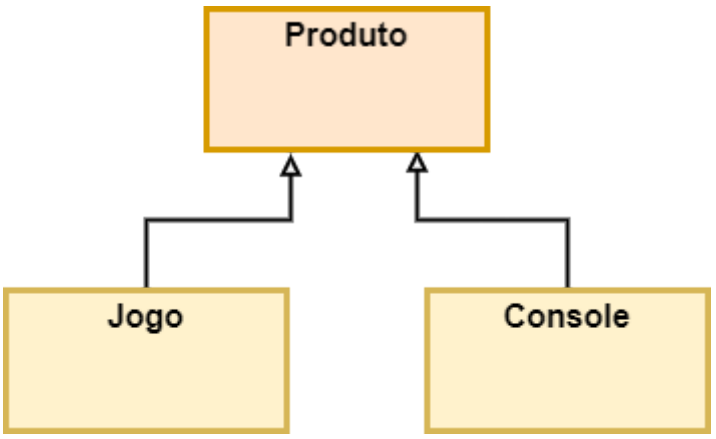
Boas práticas:

1. Construa o projeto utilizando a Linguagem TypeScript.
2. Utilize as tipagens do TypeScript na construção das variáveis, objetos e métodos
3. Crie o Menu principal na Classe Menu
4. Construa as 3 Classes Model, conforme o Diagrama de Classes acima
5. A Classe Funcionario será a Super Classe da aplicação e será definida como Abstrata.
6. Defina pelo menos 4 atributos além do id do funcionario.
7. As Classes Gerente e Vendedor serão Sub Classes da Classe Funcionario (Heranças)
8. Defina pelo menos 1 atributo relevante em cada Sub Classe.
9. Construa a Interface FuncionarioRepository contendo os Métodos do CRUD:
    a. Criar()
    b. Listar Todos()
    c. Consultar por Id()
    d. Atualizar()
    e. Deletar()
10. Construa a Classe FuncionarioController, implementando os Métodos da Interface FuncionarioRepository

11. Crie as respectivas entradas de dados para os Métodos da Classe FuncionarioController no Menu
12. Os funcionários deverão ser armazenados em uma Collection Array
13. Envie o projeto para um repositório no seu GitHub.
14. **Desafio:** *Crie um Método adicional para calcular o salário do colaborador com base nas bonificações e metas, conforme a função do colaborador.*

Atividade 03

Dado o Diagrama de Classes abaixo, construa uma aplicação que contenha um Menu principal, que forneça todas as operações do CRUD para um sistema de uma Loja de Games:



Boas práticas:

1. Construa o projeto utilizando a Linguagem TypeScript.
2. Utilize as tipagens do TypeScript na construção das variáveis, objetos e métodos
3. Crie o Menu principal na Classe Menu
4. Construa as 3 Classes Model, conforme o Diagrama de Classes acima
5. A Classe Produto será a Super Classe da aplicação e será definida como Abstrata.
6. Defina pelo menos 4 atributos além do id do produto.
7. As Classes Jogo e Console serão Sub Classes da Classe Produto (Heranças)
8. Defina pelo menos 1 atributo relevante em cada Sub Classe.
9. Construa a Interface ProdutoRepository contendo os Métodos do CRUD:
    a. Criar()
    b. Listar Todos()
    c. Consultar por Id()
    d. Atualizar()
    e. Deletar()
10. Construa a Classe ProdutoController, implementando os Métodos da Interface ProdutoRepository
11. Crie as respectivas entradas de dados para os Métodos da Classe ProdutoController no Menu

12. Os Produtos deverão ser armazenados em uma Collection Array
13. Envie o projeto para um repositório no seu GitHub.
14. **Desafio:** *Crie um Método adicional para calcular o valor do produto, com base nos descontos e no frete.*