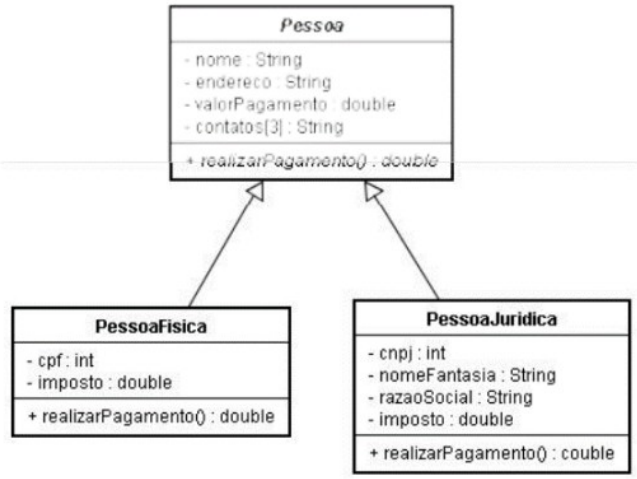
Data: 28/11/23 **Nome:** **E-mail:** **Vaga: Dev Python**

Exercício 1:

Implementar as classes do seguinte diagrama:



Detalhes:
• Atributo contatos[] é do tipo String e corresponde a uma lista de 3 posições.
• Atributo imposto corresponde a uma porcentagem que será diminuída sobre o valorPagamento (10% para PF e 20% para PJ: estes valores devem ser inicializados no construtor da classe).
• O método realizarPagamento retorna o valor do pagamento diminuído da porcentagem do imposto.

Exercício 2

Crie uma classe “Produto” que possua os atributos “nomeloja” e “preco”, crie os métodos sets e gets para estes atributos. Crie também o atributo “descrição” e seu método chamado “getDescrição” que retorna uma string com o simples conteúdo “Produto de informática”. Crie duas classes filhas de “Produto”, que serão “Mouse” com o atributo “tipo” e “Livro” com o atributo “autor”, no método construtor de cada uma dessas classes passe como argumento a descrição desse produto, por exemplo, Mouse (“Mouse ótico, Saída USB. 1.600 dpi”); Crie o método “getDescrição” que retorna a descrição que foi passada no argumento do construtor concatenada com o atributo que a classe tiver, “autor” no caso de livro e “tipo” no caso de mouse, esse método deve ter a mesma assinatura do método “getDescrição” da classe pai “Produto”. Crie um script “main” que simulará a compra de um cliente de vários mouses e livros, deve haver apenas uma lista no “Main” para armazenamento de todos os livros e mouses. Essa lista deve se chamar “carrinho” que simula o carrinho de compras de produtos variados de um cliente em um e-commerce. Insira nesse “carrinho” vários mouses e livros e depois chame o método “getDescrição” de todos os objetos presentes na lista. Para o usuário do carrinho saber as informações dos produtos em seu carrinho.

Exercício 3

Crie um sistema chamado Sistema de Gestão de Pessoas (SGP). Nele, haverá um script principal chamado main.py com um menu para gerenciar dados de colaboradores relacionados com a empresa. Escreva todo o programa documentando e testando restrições de entradas de dados e exceções (try-except no Python).

Crie uma Classe Pessoa, contendo os atributos encapsulados, com seus respectivos seletores (getters) e modificadores (setters), e ainda o construtor padrão. Atributos: nome, endereço, CPF, RG e telefone.

Considere, como subclasse da classe Pessoa, a classe Fornecedor. Considere que cada instância da classe Fornecedor tem, para além dos atributos que caracterizam a classe Pessoa, os atributos valorCredito (correspondente ao crédito máximo atribuído ao fornecedor) e valorDivida

(montante da dívida para com o fornecedor). Implemente na classe Fornecedor, para além dos usuais métodos seletores e modificadores, um método obterSaldo() que devolve a diferença entre os valores dos atributos valorCredito e valorDivida. Depois de implementada a classe Fornecedor, 2 altere o main para que você possa verificar o funcionamento dos métodos implementados na classe Fornecedor e os herdados da classe Pessoa.

Identifique a possibilidade de usar além dos métodos de objetos, métodos de classe (@classmethod) e métodos fora de contexto (@staticmethod) nos exercícios 4 e 5.

Exercício 4
Implemente uma classe chamada “Agenda” que represente uma agenda telefônica. Essa classe deve permitir adicionar, editar e remover contatos, além de buscar por contatos a partir de um nome ou número de telefone.

Exercício 5
Crie uma classe chamada “MáquinaDeVendas” que simule uma máquina de venda de produtos. Essa classe deve permitir cadastrar produtos, selecionar um produto para compra, inserir dinheiro, retornar o troco e exibir o estoque disponível.

Exercício 6
Crie uma classse Student e adicione três métodos à classe Student que compara dois objetos Student. Um método deve testar a igualdade. Um segundo método deve testar para menor que. O terceiro método deve testar para maior que ou igual a. Em cada caso, o método retorna o resultado da comparação dos nomes dos dois alunos. Inclua uma função main que testa todos os operadores de comparação. Em seguida, coloque vários objetos Student em uma lista e embaralhe. Em seguida, execute o método sort com esta lista e exiba todas as informações dos alunos.