**Engenharia Elétrica – Programação Orientada Objetos**

**Trabalho 1**

## Informações Gerais

- **Data de Entrega**: 18/10/2023 (23:59)
- **Método de Entrega**: Um arquivo zip contendo os códigos dos trabalhos devem ser enviados até a data limite para o e-mail filipe.mutz@ufes.br . O título do e-mail deve ser "EE – POO – TRAB 1 – [nomes completos dos integrantes]" e o corpo do e-mail deve conter os nomes e números de matrículas dos alunos.
- Os trabalhos deverão ser feitos em duplas.
- **Entregas além da data limite ou entregues em formato diferente do especificado acima não serão corrigidos**.
- **Trabalhos que resultarem em erro de execução receberão nota zero. Se um trecho de código não estiver funcionando até a data de entrega, comente o trecho ou remova. Só receberão pontuação as funcionalidades que estiverem funcionando.**

## Descrição

O trabalho consiste em criar uma loja virtual em que pessoas podem comprar itens usando programação orientada a objetos em Python. Os usuários do sistema podem gerenciam os produtos vendidos, realizar compras e ver relatórios. Devem ser criadas as seguintes classes de domínio:

### Pessoa

- Atributos:
  - Nome: str
  - Email: str
  - Cpf: str
- Métodos:
  - Construtor

### Produto

- Atributos:
  - Nome: str
  - Preco: float (privado; > 0)
  - Quantidade em estoque: int (inicialmente 0; privado; >= 0)
  - Desconto percentual: float (inicialmente 0, privado; >= 0 e < 1)
  - Categoria: str
- Métodos:
  - construtor que receba como argumentos o nome, o preço e a categoria do produto. Os demais campos devem ser inicializados com valores padrão.
  - preco: retorna o preco aplicando o desconto percetual.

    - Registrar aquisicao: usado para incrementar a quantidade de itens em estoque quando estes são adquiridos junto ao fornecedor. O método recebe como argumento a quantidade de itens comprados.
    - Registrar venda: usado para decrementar a quantidade em estoque quando um cliente realiza uma compra. Recebe como argumento o número de itens comprados. A quantidade em estoque não pode se tornar negativa.
    - Quantidade em estoque: retorna a quantidade de produtos em estoque.
    - Atualizar desconto: recebe o novo valor do desconto garantindo que o novo valor é válido.
    - Atualizar preço: recebe o novo valor do preço garantindo que o novo valor é válido.

**Item de Compra: Representa um item do carrinho de compras.**

- Atributos
    - Produto sendo adquirido: Produto
    - Número de cópias do item: int (> 0)
- Métodos
    - Construtor
    - Custo: retorna o valor total dos itens. Por exemplo, se o item é um livro que custa 10 reais e estão sendo compradas 5 cópias, o método deve retornar 50.

**Compra: Representa o carrinho de compras**

- Atributos
    - Cliente: Usuario
    - Itens: lista de itens de compra (inicialmente vazio).
- Métodos:
    - Construtor. Recebe como entrada um usuário representando a pessoa que está fazendo a compra.
    - Custo: retorna o valor total da compra.
    - Adicionar produto: Recebe o produto e a quantidade sendo adquirida, cria um objeto do tipo Item de Compra e adiciona na lista de itens.
    - Remover produto: recebe o índice do item na lista de itens e remove-o do carrinho.
    - Atualizar quantidade: recebe o índice do item na lista de itens e a nova quantidade.
    - Visualizar compra: Mostra na tela os produtos que pertencem à compra, o valor unitário, o valor total relativo ao produto e, ao final, o valor total da compra. O índice do produto na lista de itens de compra deve ser exibido para permitir que o usuário remova ou atualize o item da compra usando este índice.

**Loja**

- Atributos:
    - Produtos: lista de objetos do tipo Produto (inicialmente vazia)
    - Compras: lista de objetos do tipo compra representando as compras que já foram concluídas.
    - Compra aberta: Compra (inicialmente None; representa a compra em andamento).
- Métodos

- Construtor. Não recebe nenhum argumento e inicializa os atributos com seus valores padrão.
- Iniciar compra: cria um objeto do tipo Compra e o define como a compra aberta. Se uma compra aberta já existir, exiba uma mensagem de erro informando que a pessoa deve finalizar ou cancelar a compra antes de criar uma nova.
- Buscar produto: recebe como entrada o código do produto, busca o objeto na lista de produtos e retorna o objeto. Se ele não existir, retorne None.
- Cancelar compra: define a compra aberta como None.
- Finalizar compra: adiciona a compra na lista de compras finalizadas e define a compra aberta como None. Em um sistema real, neste ponto, o usuário deveria realizar o pagamento e definir o endereço de entrega, mas não faremos isso. Por fim, para cada item na compra, desconte a quantidade comprada da quantidade em estoque do produto.

| |
|---|
| As operações de inserir, atualizar e remover produtos da compra em andamento podem ser feitas usando diretamente os métodos da classe Compra existentes no atributo “Compra aberta”. |

- Métodos de Relatório:
  - Número de produtos: retorna o número de produtos vendidos na loja.
  - Número de vendas: retorna o número de compras concluídas.
  - Valor total vendido: retorna a soma dos valores totais das compras concluídas.
  - Valor médio das compras: retorna o valor total médio das compras concluídas.
  - Número de usuários: retorna o número de pessoas diferentes que já fizeram compras na loja.
  - Usuário que mais fez compras: retorna um objeto do tipo Pessoa representando a pessoa que mais finalizou compras na loja.
  - 5 produtos mais caros: Retorne uma lista contendo os 5 produtos mais caros da loja. Não é válido mostrar os dados na tela diretamente no método.
  - 5 produtos mais vendidos e o valor total que a loja recebeu com as vendas do produto. O método pode exibir os dados diretamente e não é necessário retorná-los como, por exemplo, uma lista. Os dados devem ser exibidos de forma ordenada, do produto que levou ao maior montante para o que levou para o menor.
  - Para cada pessoa, some o valor total das compras feitas pela pessoa. Mostre na tela os nomes das pessoas e seguido pelo CPF e o valor total das compras. Os dados devem ser ordenados pelo nome da pessoa (ordem alfabética). Não é necessário retornar os dados como uma lista.
- Salvar: Salva os dados de produtos e compras de forma que as informações persistam mesmo que o programa seja encerrado. A compra aberta não deve ser salva.
- Carregar: Lê os dados de produtos e compras do arquivo e usa estas informações para preencher as respectivas listas.

**App**

- Atributos
    - loja: objeto do tipo loja
- Métodos
    - Construtor: Não deve receber argumentos e deve inicializar o atributo loja com um novo objeto do tipo Loja.
    - Menu: deve exibir um menu com as opções listadas na próxima seção e retornar a opção escolhida pelo usuário.
    - Executar: deve repetidamente mostrar o menu para o usuário e realizar a opção escolhida até que seja selecionada a opção de sair.

**Opções do Menu**

1. Cadastrar produto: Solicite que o usuário digite os dados do produto, cria um objeto do tipo Produto e o adiciona na lista de produtos da loja.
2. Ver lista de produtos: Mostra na tela o código, o nome e o valor (com desconto) de todos os produtos da loja, um por linha. Para cada produto, deve ser exibido o índice do produto na lista.
3. Ver detalhes de produto: Solicita que o usuário digite o código de um produto e mostre todas as suas informações, incluindo o valor de desconto e o valor bruto (sem desconto).
4. Atualizar desconto de produto: Solicita que o usuário digite o código de um produto e um novo valor de desconto e atualiza o desconto do produto.
5. Registrar aquisição de produto: Solicita que o usuário digite o código de um produto e a quantidade comprada junto ao fornecedor e incrementa a quantidade em estoque do produto.
6. Remover produto: Solicita que o usuário digite o código de um produto e remove o produto da lista.
7. Iniciar compra: Ao iniciar a compra, solicite que o usuário digite os dados do cliente que está fazendo a compra. Use estes dados para criar um objeto do tipo Pessoa que será passado para o construtor da nova Compra.
8. Cancelar compra
9. Finalizar compra
10. Adicionar item na compra: Solicita que o usuário digite o código do produto e a quantidade sendo comprada e adiciona o item na compra. Se o produto já existir compra, deve ser exibida uma mensagem de erro e ele não deve ser adicionado novamente.
11. Visualizar compra
12. Remover item da compra: Solicita que o usuário digite o índice do item na lista da compra e remove o respectivo item.
13. Atualizar quantidade de item na compra: Solicita que o usuário digite o índice do item na lista da compra e a nova quantidade e atualiza o item.
14. Relatório - Número de produtos.
15. Relatório - Número de vendas.
16. Relatório - Valor total vendido.
17. Relatório - Valor médio das compras.
18. Relatório - Número de usuários.
19. Relatório - Usuário que mais fez compras.
20. Relatório - 5 produtos mais caros.

21. Relatório - 5 produtos mais vendidos.
22. Relatório – Montante por pessoa.

**Restrições**

- Cada classe deve ser armazenada em um arquivo .py separado.
- Os arquivos .py devem importar apenas classes e métodos que forem utilizados no arquivo.
- O nome de atributos e métodos privados devem iniciar com "_" e estes não devem ser utilizados diretamente fora da classe em que eles foram definidos.
- Deve ser utilizada uma função main definida em um arquivo chamado main.py.
- Todos os atributos e métodos devem possuir *type hints*.
- Deve ser feito tratamento de erros. Por exemplo, na opção 4 do menu, se o produto não existir ou o desconto apresentar um valor inválido, devem ser exibidas mensagens informativas e o valor não deve ser atualizado.