UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA

VIVIAN MARTINS MOURA

RELATÓRIO PBL

FEIRA DE SANTANA

2025

1. INTRODUÇÃO

1.1 Resumo do problema

A proposta é criar um sistema de atendimento para um quiosque de chás personalizados, mais conhecido como Bubble tea, e esse sistema deverá ser capaz de armazenar o cadastro de todos os clientes, incluindo o nome da pessoa, a sua categoria (estudante ou funcionário), e o valor do cashback que a pessoa possui. Criar esse sistema ajudará à loja a ter uma melhor organização e a ter um atendimento mais rápido e eficiente.

1.2 Descrição breve da solução

Então implementei esse sistema utilizando um código em python na versão mais atual, a versão 3.13.2, no Visual Studio Code e implementei tudo que foi solicitado, que detalharei melhor na metodologia.

2. METODOLOGIA

2.1 Processo de construção de conhecimento nas sessões tutoriais

Na primeira sessão mesmo lendo o problema eu não havia entendido muito bem como eu faria o programa armazenar o cadastro dos clientes sem perder esse cadastro assim que fechasse o programa, pois ainda não havia estudado sobre a persistência de dados, porém após essa sessão eu estudei sobre e entendi melhor como esse programa teria que funcionar. Além disso, as sessões no geral me ajudaram a perceber que haviam várias maneiras diferentes de criar o sistema, pois muitos dos meus colegas me explicaram como fizeram e cada um pensou em uma solução diferente, alguns por exemplo utilizaram arquivo texto e outros utilizaram arquivo binário, e os dois tipos de arquivo deram certo.

2.2 Definição de requisitos

O sistema deve permitir o cadastro e identificação de clientes, aplicando descontos conforme a categoria (25% para estudantes e R$1,00 para funcionários). O cliente escolhe uma base de chá e até cinco complementos, com os preços já definidos. Após o pedido, o sistema calcula o valor com descontos e oferece a opção de uso do cashback acumulado caso o cliente tenha algum valor de cashback guardado de pedidos anteriores. A cada compra, 10% do valor final é retornado como cashback. O sistema deve exibir um resumo completo do pedido e persistir os dados dos clientes em um arquivo. Entradas inválidas devem ser tratadas, e o fluxo permite o atendimento de múltiplos clientes consecutivamente, até a atendente decidir encerrar os atendimentos.

2.3 Descrição de alto nível

O programa foi desenvolvido utilizando a linguagem Python e aplicando o conceito de modularização. Ele é dividido em dois arquivos: um contendo a parte principal do código e outro com as classes e com as funções auxiliares. Essa separação ajuda a organizar melhor o

código. O programa simula um sistema de atendimento personalizado para uma loja de bebidas, permitindo:

- Cadastrar clientes e manter histórico com uso de arquivos.
- Personalizar pedidos com escolha de bases e complementos.
- Aplicar políticas de desconto e cashback.
- Armazenar e recuperar dados de clientes com persistência em arquivo binário.

Persistência de Dados: Utiliza o módulo pickle para armazenar e recuperar os dados dos clientes em um arquivo chamado 'registro.dat'. Isso garante que informações como nome, categoria e saldo de cashback sejam preservadas entre execuções.

Definição de Estruturas de Dados:

- Classe Cliente: representa um consumidor, com nome, categoria (estudante ou funcionário) e cashback.
- Classe bebida: usada para armazenar tanto as bases (ex: leite, manga) quanto os complementos (ex: boba, taro), com nome e preço.
- Dicionários bases e complementos: armazenam as opções de personalização disponíveis.

Verificação de Existência de Cliente: A função ver_se_existe(nome) é responsável por verificar se um cliente já possui cadastro no sistema. Para isso, ela tenta abrir o arquivo de registros (registro.dat) e procurar o nome informado. Se o nome for encontrado, significa que o cliente já está registrado, e suas informações (como categoria e saldo de cashback) podem ser recuperadas para personalizar o atendimento. Caso contrário, o sistema informa que o cliente ainda não está cadastrado, permitindo que um novo registro seja criado. Essa verificação garante que cada cliente tenha um histórico individual de compras e cashback acumulado.

Interface de Atendimento:

- O programa inicia com um loop perguntando ao atendente se deseja iniciar um atendimento.
- Caso afirmativo, o cliente é identificado (ou cadastrado, se for um novo cliente).
- Em seguida, o cliente personaliza sua bebida escolhendo uma base e até cinco complementos.

Cálculo de Preço e Benefícios:

- Aplica descontos com base na categoria do cliente:
- Estudantes: 25% de desconto.
- Funcionários: R$ 1,00 de desconto fixo.
- Oferece a possibilidade de usar o saldo de cashback acumulado em compras anteriores caso o cliente não seja novo.

- Um novo valor de cashback é calculado (10% do valor final) e salvo no registro.

Finalização e Armazenamento:

- Ao final do pedido, o sistema apresenta um resumo ao usuário, como solicitado no problema.
- As alterações no objeto Cliente (como o novo cashback) são armazenadas novamente no arquivo registro.dat com pickle.

2.4 Ordem de codificação

Realmente peguei o código para fazer apenas na segunda semana, porém já na primeira semana comecei estudando mais sobre persistência de dados e classes e objetos a partir de vídeos no youtube, pois ainda não tinha conhecimento desses assuntos o suficiente para começar a criar o programa. Na segunda semana que comecei a ter ideias e colocá-las em prática para ver se funcionavam, e consegui fazer o código praticamente todo, porém tive muita dificuldade na parte da persistência de dados, que foi o que eu mais demorei para implementar no código pois não estava dando certo no início, mas após muitas tentativas eu consegui fazer funcionar. Então, na terceira semana eu já estava com o código pronto, só ficou faltando comentar o código, adicionar a mensagem de 'não plágio' e fazer a verificação das entradas, que eu fiz na última semana, assim terminando meu código.

3. RESULTADOS E DISCUSSÕES

3.1 Manual de uso

3.1.1 Como utilizar o seu programa?

Ao iniciar, o sistema perguntará se o atendente deseja realizar o atendimento de um cliente. Caso a resposta seja afirmativa (digitando 1), o próximo passo será informar o nome do cliente, que será convertido automaticamente para letras maiúsculas. Em seguida, o sistema verifica se esse cliente já possui cadastro. Se estiver registrado, suas informações anteriores, como categoria (estudante ou funcionário) e saldo de cashback, serão exibidas. Caso contrário, o sistema solicitará que o atendente informe a categoria do novo cliente, digitando 1 para estudante ou 2 para funcionário, e o cadastro será realizado. Após a identificação ou cadastro do cliente, ele poderá montar seu pedido. Primeiro, o sistema solicitará a escolha de uma base para a bebida, dentre quatro opções numeradas de 1 a 4. Em seguida, o cliente informará quantos complementos deseja adicionar, sendo possível escolher até cinco. Para cada complemento, ele deverá digitar o número correspondente ao item desejado. A cada escolha, os valores serão acumulados para cálculo do preço final. Finalizado o pedido, o programa apresenta um resumo contendo o nome do cliente, a base escolhida, os complementos selecionados e o valor total. Caso o cliente possua saldo de cashback acumulado, será perguntado se deseja utilizá-lo para obter um abatimento adicional. Se optar por usá-lo, o valor será subtraído do total. Caso contrário, o valor do cashback será mantido e somado ao novo valor que será gerado a partir da compra atual. Por fim, os dados atualizados do cliente são armazenados automaticamente em

um arquivo de dados chamado ‘registro.dat’, garantindo que as informações sejam preservadas para atendimentos futuros. O programa então volta ao início, pronto para realizar novos atendimentos, até que o atendente opte por encerrar digitando a opção correspondente.

3.1.2 Qual é o conjunto de dados de entrada válido para o correto funcionamento do programa?

- Opção de atendimento: número 1 (sim), caso queira atender um cliente, ou 2 (não), caso não queira.
- Nome do cliente: texto (sem restrições).
- Categoria: número 1 para estudante, 2 para funcionário.
- Base: número de 1 a 4, representando as opções disponíveis.
- Número de complementos: de 0 a 5.
- Complementos: número de 1 a 5 para cada um.
- Uso do cashback: número 1 (sim), caso queira usar, ou 2 (não), caso não queira usar.

3.1.3 Quais são as saídas do programa? Como são geradas as saídas (processamento para obtê-las)?

Saídas do programa:

- Exibição de mensagens de boas-vindas e instruções.
- Informações do cliente (categoria e cashback, se já cadastrado).
- Resumo do pedido (base, complementos, valor com desconto).
- Mensagem informando se o cashback foi utilizado ou acumulado.
- Valor total final da compra.
- Novo saldo de cashback.
- Geração das saídas: Essas saídas são produzidas com base nas escolhas feitas pelo cliente durante a execução, usando condicionais (if) e laços (while/for) pra tratar a lógica de desconto, cashback e composição do pedido.

3.1.4 Quais os testes efetuados (durante e ao final do desenvolvimento) e quais os resultados obtidos? Quais erros ocorreram nos testes? Em que situação o seu programa não funcionaria?

Durante o desenvolvimento, foram realizados testes para verificar o correto funcionamento do cadastro de clientes, montagem de pedidos, aplicação de descontos e uso do cashback. Os cálculos dos valores finais estavam certos e o sistema respondeu bem a entradas inválidas. Também foram simulados atendimentos seguidos para testar se o funcionamento geral do programa estava certo, que funcionou conforme o esperado em todos os casos. Também tentei criar um pedido com um cliente já existente, para ver se o programa estava recuperando os dados corretamente, e no início não estava funcionando, ele não recuperava o valor do cashback corretamente, porém depois dos ajustes necessários funcionou perfeitamente. Acredito que meu programa funcionaria em qualquer situação, não achei nenhuma falha nele, mas talvez exista alguma e eu não tenha percebido.

4. CONCLUSÃO

Acredito que todos os requisitos foram cumpridos e acredito que uma maneira de melhorar ainda mais esse código seria deixá-lo mais bonito visualmente para o usuário, o que eu poderia tentar implementar mais para frente, porém agora eu foquei mais no funcionamento do código do que na sua aparência.

5. BIBLIOGRAFIA CONSULTADA

HASHTAG PROGRAMAÇÃO. *Como Funcionam Classes e Programação Orientada a Objetos em Python*. YouTube, 8 dez. 2022. Disponível em: https://www.youtube.com/watch?v=97A_Cyyh-eU. Acesso em: 22 maio 2025.

HASHLDASH. *22 - Python Kivy - Salvando informações e persistência de dados.* YouTube, 3 out. 2018. Disponível em: https://youtu.be/6A8igVPUNLc. Acesso em: 21 maio 2025.