TRABALHO PRÁTICO - DOCUMENTO DESCRITIVO

NOME: Bianca Karen Dias Oliveira

PDITA: 049

CIDADE: Itabira

DISCIPLINA: Programação Básica com Python

INTRODUÇÃO

Tema: Padaria.

1 Usuários

O sistema terá três tipos de usuários: Funcionário, Gerente e Cliente.

2 Produtos

Os produtos possuem os seguintes atributos: Nome, Código e Preço. O código de cada produto possui três dígitos e é gerado a partir de um número aleatório entre 1 e 999. A empresa modelada vende produtos como pães, doces e bebidas. Abaixo, a lista de produtos já cadastrados:

Nome	Código	Preço
Pão francês	359	0,50
Pão de queijo	136	3,00
Pão doce	498	1,00
Café pequeno	969	1,50
Café grande	789	2,50
Café com leite pequeno	599	2,00
Café com leite grande	925	3,00
Bolo fatia	667	2,50
Sonho	977	4,00
Suco pequeno	773	3,50

Suco grande	356	4,50
Refrigerante	078	2,50
Coxinha de frango	809	6,25
Pastel	349	4,50
Empada de frango	594	5,50
Pastel assado	622	6,50
Fatia de torta doce	712	3,25
Tortinha de frango	458	7,00
Capuccino pequeno	324	3,50
Capuccino grande	652	5,00
Chá pequeno	918	1,00
Chá grande	901	2,00
Rocambole fatia	755	6,00
Misto quente	726	7,50
Bolo no pote	598	10,00
Brigadeiro	389	3,00
Bombom	401	3,50
Pudim grande	560	25,00
Sanduíche natural	742	5,00

IMPLEMENTAÇÃO

1 Organização das informações do arquivo

1.1 Arquivo de usuários

A estrutura de dados utilizada para carregar os dados dos usuários será uma lista de tuplas

1.2 Arquivo de produtos

Assim como o arquivo de usuários, os dados do arquivo de produtos serão carregados como uma lista de tuplas.

3 Estruturação dos arquivos de registros

3.1 Registros de usuários

O arquivo de usuários é do tipo .csv e para cada usuário há os seguintes atributos: Nome, Tipo (que poderá ser gerente, funcionário e cliente) e Senha. Cada linha possui os três atributos separados por vírgula.

3.2 Registro de produtos

O arquivo de produtos também é do tipo .csv e, conforme a tabela, possui os seguintes atributos: Nome, Código e Preço. O separador desse arquivo também é a vírgula.

3 Funcionalidades

3.1 CRUD de usuários

Para manipulação do arquivo de usuários, o programa contém as seguintes funcionalidades:

- Cadastro de usuário (C)
- Login de usuário (R)
- Listagem dos usuários (R)
- Leitura do arquivo de usuários (R)
- Busca de usuário (R)
- Atualização de dados do usuário (U)
- Remoção do usuário (D)

3.2 CRUD de produtos

Para o arquivo de produtos, o programa foram desenvolvidas as seguintes funcionalidades:

- Cadastro de produto (C)
- Busca por produto (por nome ou código) (R)
- Listagem de todos os produtos ordenados por nome ou por preço (R)
- Leitura do arquivo de produtos (R)
- Atualização de preço do produto (U)
- Remoção de produto (D)

CONCLUSÃO

O trabalho realizado envolve vários conceitos aprendidos durante a disciplina e, devido ao tema, é necessário ter vários aspectos em mente na hora de criar o programa (por exemplo, verificar se o usuário existe antes de cadastrá-lo ou verificar se um preço digitado é válido), por isso é preciso que haja várias verificações para recepção correta dos dados e funcionamento do sistema, mesmo que seja simples quando comparado com soluções reais.

No quesito manipulação de arquivos, por exemplo, foi preciso ler os arquivos, transformar cada linha em tupla, armazenar essas tuplas como itens de uma lista e por fim descartar o cabeçalho, mudança crucial para que não houvessem problemas como o de conversão de tipo, no caso do preço dos produtos, quando essa conversão for necessária.

Devido a esse processo, o primeiro item de cada lista passa a ter endereço 0, então na execução das funções de busca houve um problema para reconhecer o endereço 0 como linha e não como valor Falso, já que uma verificação para atestar a inexistência de um produto era escrita na forma "*if not linha*" então para a linha de endereço 0 seria valor Falso para a existência do produto. Como solução, a verificação passou a ser de tipo, então se linha tem valor *int*, o produto existe, se tiver valor *bool*, não existe.

Outra questão é a estrutura escolhida para carregar os dados. Como mencionado anteriormente, foi utilizada a tupla, onde cada item da tupla é um atributo do produto/usuário. Porém, para ler e escrever os dados foi utilizada a biblioteca *csv,* e após um estado mais avançado do desenvolvimento do trabalho, surgiu uma forma mais rápida de manipular os dados, pois foi descoberto que essa biblioteca consegue tratar os arquivos .csv de forma mais simples usando dicionários, através das funções *DictWrite*r (para criação do escritor) e *DictReader* (para criação do leitor). Quando o leitor é chamado, para cada linha recebida é possível atribuir uma chave para cada item do dicionário, enquanto que para criação do escritor, além do arquivo, há passagem como parâmetro de uma lista com os nomes das colunas, no campo *fieldnames*, o que facilitaria a escrita. Assim, talvez fosse uma escolha melhor utilizar dicionários para carregamento dos dados.

Portanto, poderiam haver modificações nesse sentido e também a adoção de alguma forma para reduzir o tamanho do código e otimizar o seu funcionamento,

mesmo que no caso do projeto atual ele já esteja modularizado praticamente em sua totalidade.