

Introdução a Computação e Lógica

Ewérton Veríssimo da Silva

PLANEJADOR DE VIAGENS

Agilson Felix de Araújo | 2024106510037

Jaciara Carla da Silva | 2024106510013

João Paulo Nóbrega Pereira da Silva | 2024106510024

João Vitor Gonçalves Valentim | 2024106510052

INTRODUÇÃO

Objetivo: Apresentar o sistema desenvolvido para auxiliar usuários no planejamento de viagens.

Visão geral: Explicar a estrutura do sistema e como cada função contribui para a experiência do usuário.

Bibliotecas importadas

Módulo “OS”: Facilitar a interação com o sistema de arquivos e diretórios, proporcionando funções para verificação, leitura e escrita de arquivos, bem como para manipulação de diretórios.

Módulo “time”: Controlar e medir o tempo durante a execução do programa, permitindo pausas temporizadas e outras operações relacionadas ao tempo.

Módulo “datetime”: Manipular e validar datas e horas de maneira precisa, permitindo conversões entre strings e objetos de data/hora, além de realizar operações com datas.

Bibliotecas importadas

```
● ● ●  
1 import os  
2 import time  
3 from datetime import datetime
```

Arquivos de Dados

usuarios.txt: Armazena os dados dos usuários no formato “usuario:senha”.

viagens.txt: Armazena os dados das viagens, associadas a cada usuário.

Arquivos de Dados



```
1 armazena_usuarios = 'usuarios.txt'  
2 armazena_viagens = 'viagens.txt'
```

Função cadastrar_usuario

Propósito

- Registrar um novo usuário no sistema.

Explicação

- Salva os novos usuários no arquivo de usuários.
 - Verifica se o usuário já existe.
 - Solicita o nome de usuário e a senha.

Função cadastrar_usuario

```
● ● ●  
1 def cadastrar_usuario(usuarios):  
2     usuario = input("Digite o nome de usuário: ")  
3     if usuario in usuarios:  
4         print("O usuário que você acabou de colocar já existe!")  
5     else:  
6         senha = input("Digite a senha: ")  
7         usuarios[usuario] = senha  
8         salvar_usuarios(usuarios)  
9         print("Seu usuário acaba de ser registrado com sucesso!")  
10
```

Função autenticar_usuario

Propósito

- Permitir que usuários autentiquem suas credenciais para acessar o sistema.

Explicação

- Solicita nome de usuário e senha.
- Verifica se as credenciais são válidas.
- Retorna o usuário autenticado ou informa se houve erro nas credenciais.

Função autenticar_usuario

```
1 def autenticar_usuario(usuarios):
2     usuario = input("Digite o nome de usuário: ")
3     senha = input("Digite a senha: ")
4     if usuario in usuarios and usuarios[usuario] == senha:
5         print("Login bem-sucedido!")
6         return usuario
```

Função carregar_usuarios

Propósito

- Carregar usuários previamente cadastrados a partir do arquivo.

Explicação

- Verifica a existência do arquivo de usuários.
- Lê os usuários do arquivo e os armazena em um dicionário.

Função carregar_usuarios

```
● ● ●  
1 def carregar_usuarios():  
2     usuarios = {}  
3     if os.path.exists(armazena_usuarios):  
4         with open(armazena_usuarios, 'r', encoding='utf-8') as arquivo_usua:  
5             for linha in arquivo_usua:  
6                 usuario, senha = linha.strip().split(':')  
7                 usuarios[usuario] = senha  
8     return usuarios
```

Função salvar_usuarios

Propósito

- Salvar usuários cadastrados de volta no arquivo.

Explicação

- Escreve os usuários do dicionário no arquivo de usuários.

Função salvar_usuarios



```
1 def salvar_usuarios(usuarios):
2     with open(armazena_usuarios, 'w', encoding='utf-8') as arquivo_usua:
3         for usuario, senha in usuarios.items():
4             arquivo_usua.write(f'{usuario}:{senha}\n')
```

Função adicionar_viagem

Propósito

- Permitir que usuários adicionem uma nova viagem ao sistema.

Explicação

- Solicita informações como destino, datas de início e término, e atividades.
- Valida as datas para garantir que a data de início não seja anterior à data atual.
- Adiciona a viagem ao registro do usuário e salva no arquivo de viagens.

Função adicionar_viagem

```
● ● ●  
1 def adicionar_viagem(viagens, usuario):  
2     destino = input("Digite o seu destino: ")  
3  
4     while True:  
5         try:  
6             data_inicio_str = input("Digite a data de início (DATA/MÊS/ANO): ")  
7             data_inicio = validar_data(data_inicio_str)  
8  
9             data_termino_str = input("Digite a data de término (DATA/MÊS/ANO): ")  
10            data_termino = validar_data(data_termino_str)  
11  
12            if data_termino < data_inicio:  
13                raise ValueError("A data de término não pode ser anterior à data de início.")  
14  
15            if data_inicio < datetime.now().date():  
16                raise ValueError("A data de início não pode ser anterior à data atual.")  
17  
18            break  
19  
20        except ValueError:  
21            print(f"Data inválida. Use o formato DATA/MÊS/ANO.")  
22
```

Função adicionar_viagem

```
● ● ●  
1  atividades = input("Adicione atividades para essa viagem separado por vírgula: ").split(',')  
2  
3  if usuario not in viagens:  
4      viagens[usuario] = []  
5  
6  viagens[usuario].append({  
7      "destino": destino,  
8      "data_inicio": data_inicio_str,  
9      "data_termino": data_termino_str,  
10     "atividades": atividades  
11 })  
12  
13 salvar_viagens(viagens)  
14 print("Viagem adicionada com sucesso!")  
15
```

Função listar_viagens

Propósito

- Exibir todas as viagens registradas para um usuário.

Explicação

- Verifica se o usuário possui viagens registradas.
- Lista todas as viagens registradas para o usuário, mostrando destino, datas e atividades.

Função listar_viagens

```
● ● ●  
1 def listar_viagens(viagens, usuario):  
2     if usuario in viagens:  
3         for idx, viagem in enumerate(viagens[usuario], 1):  
4             print(f"Viagem {idx}: {viagem}")  
5     else:  
6         print("Nenhuma viagem encontrada para este usuário.")
```

Função validar_data

Propósito

- Validar se uma data está no formato correto e não é anterior à data atual.

Explicação

- Verifica se o usuário possui viagens registradas.
- Lista todas as viagens registradas para o usuário, mostrando destino, datas e atividades.

Função validar_data

```
● ● ●  
1 def validar_data(data_str):  
2     return datetime.strptime(data_str, '%d/%m/%Y').date()
```

Função excluir_viagem

Propósito

Permitir que usuários excluam uma viagem registrada.

Explicação

- Exibe todas as viagens registradas para o usuário.
- Solicita ao usuário o número da viagem que deseja excluir.
- Remove a viagem selecionada e salva as alterações no arquivo de viagens.

Função excluir_viagem

```
● ● ●  
1 def excluir_viagem(viagens, usuario):  
2     if usuario in viagens and viagens[usuario]:  
3         listar_viagens(viagens, usuario)  
4         viagem_idx = int(input("Digite o número da viagem que deseja excluir: ")) - 1  
5         if 0 <= viagem_idx < len(viagens[usuario]):  
6             viagens[usuario].pop(viagem_idx)  
7             salvar_viagens(viagens)  
8             print("A viagem acabou de ser excluída")  
9         else:  
10            print("O índice da viagem que você quer cancelar é inválido!")  
11     else:  
12         print("Não encontramos nenhuma viagem para esse usuário.")
```

Função carregar_viagens

Propósito

Permitir que usuários excluam uma viagem registrada.

Explicação

- Exibe todas as viagens registradas para o usuário.
- Solicita ao usuário o número da viagem que deseja excluir.
- Remove a viagem selecionada e salva as alterações no arquivo de viagens.

Função carregar_viagens

```
● ● ●  
1 def carregar_viagens():  
2     viagens = {}  
3     if os.path.exists(arohana_viagens):  
4         with open(arohana_viagens, 'r', encoding='utf-8') as file:  
5             usuario_atual = None  
6             for linha in file:  
7                 linha = linha.strip()  
8                 if linha.startswith('Usuário:'):br/>9                     usuario_atual = linha.split(':')[1]  
10                    viagens[usuario_atual] = []  
11                elif usuario_atual:  
12                    partes = linha.split('|')  
13                    viagem = {  
14                        "destino": partes[0],  
15                        "data_inicio": partes[1],  
16                        "data_termino": partes[2],  
17                        "atividades": partes[3].split(',')  
18                    }  
19                    viagens[usuario_atual].append(viagem)  
20    return viagens
```

Função salvar_viagens

Propósito

- Salvar viagens registradas de volta no arquivo.

Explicação

- Escreve as viagens do dicionário no arquivo de viagens.

Função salvar_viagens



```
1 def salvar_viagens(viagens):
2     with open(arizona_viagens, 'w', encoding='utf-8') as file:
3         for usuario, lista_viagens in viagens.items():
4             file.write(f"Usuário:{usuario}\n")
5             for viagem in lista_viagens:
6                 atividades = ','.join(viagem['atividades'])
7                 file.write(f"{viagem['destino']}|{viagem['data_inicio']}|{viagem['data_termino']}|{atividades}\n")
```

Função main

Propósito

- Coordenar a execução do sistema.

Explicação

- Escreve as viagens do dicionário no arquivo de viagens.

Função Main

```
● ● ●  
1 def main():  
2     usuarios = carregar_usuarios()  
3     viagens = carregar_viagens()  
4  
5     while True:  
6         print("\n1. Fazer Login\n2. Cadastrar\n3. Sair")  
7         escolha = input("Escolha uma opção: ")  
8  
9         if escolha == '1':  
10            usuario = autenticar_usuario(usuarios)  
11            if not usuario:  
12                print("Usuário ainda não possui conta. Por favor, cadastre-se.")  
13                continue
```

Função Main

```
1  while True:
2      print("\n1. Adicionar Viagem\n2. Listar Viagens\n3. Deletar Viagem\n4. Sair")
3
4      opcao = input("Escolha uma opção: ")
5
6      if opcao == '1':
7          adicionar_viajem(viagens, usuario)
8      elif opcao == '2':
9          listar_viajens(viagens, usuario)
10     elif opcao == '3':
11         excluir_viajem(viagens, usuario)
12     elif opcao == '4':
13         break
14     else:
15         print("A opção que você inseriu é inválida!")
16
17     elif escolha == '2':
18         cadastrar_usuario(usuarios)
19
20     elif escolha == '3':
21         print("Saindo...")
22         time.sleep(3)
23         print('Sistema finalizado com sucesso')
24         break
25
26     else:
27         print("A opção que você inseriu é inválida!")
28 main()
```



