```
import json
import os
import folium
import webbrowser

FILE_NAME = "onibus.json"

class Onibus:
    def __init__(self, numero: str, rota: str, capacidade: int, cor: str, coordenadas=None):
        self.numero = numero
        self.rota = rota
        self.capacidade = capacidade
        self.cor = cor
        self.coordenadas = coordenadas if coordenadas else []

    def to_dict(self):
        return {
            "numero": self.numero,
            "rota": self.rota,
            "capacidade": self.capacidade,
            "cor": self.cor,
            "coordenadas": self.coordenadas
        }

    @staticmethod
    def from_dict(data):
        return Onibus(
```

```
            data["numero"], data["rota"], int(data["capacidade"]), data["cor"],
data.get("coordenadas", [])
        )

class GerenciadorOnibus:
    def __init__(self, arquivo=FILE_NAME):
        self.arquivo = arquivo
        self.dados = self.carregar_dados()

    def carregar_dados(self):
        if not os.path.exists(self.arquivo):
            return {}
        try:
            with open(self.arquivo, "r", encoding="utf-8") as file:
                dados_json = json.load(file)
                return {d["numero"]: Onibus.from_dict(d) for d in dados_json}
        except json.JSONDecodeError:
            print("Erro ao carregar dados. Arquivo JSON inválido.")
            return {}

    def salvar_dados(self):
        try:
            with open(self.arquivo, "w", encoding="utf-8") as file:
                json.dump([onibus.to_dict() for onibus in self.dados.values()], file, indent=4,
ensure_ascii=False)
        except Exception as e:
            print(f"Erro ao salvar dados: {e}")

    def inicializar_dados(self):
```

```
        if self.dados:
            return
        dados_iniciais = [
            Onibus("860", "Tapanã - UFPA", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4505, -48.4910)]),
            Onibus("862", "Tapanã - Felipe Patroni", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4600, -48.4900)]),
            Onibus("866", "Tapanã - Ver-o-Peso", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4550, -48.5030)]),
            Onibus("869", "Tapanã - Presidente Vargas", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4520, -48.4800)]),
            Onibus("861", "Tapanã 2 - Ver-o-Peso", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4555, -48.5055)]),
            Onibus("893", "Tapanã (Socipe) - Castanheira", 45, "Laranja", [(-1.4092, -48.4671), (-1.4700, -48.4895)]),
            Onibus("639", "Pratinha - Castanheira (Via Tapanã)", 45, "Laranja", [(-1.4092, -48.4671), (-1.4750, -48.4880)]),
            Onibus("643", "Pratinha - UFPA", 45, "Laranja", [(-1.4092, -48.4671), (-1.4500, -48.4950)]),
            Onibus("757A", "Jardim Europa - Presidente Vargas", 45, "Laranja", [(-1.4092, -48.4671), (-1.4530, -48.4825)]),
            Onibus("757B", "Jardim Europa - Ver-o-Peso", 45, "Laranja", [(-1.4092, -48.4671), (-1.4560, -48.5050)]),
            Onibus("665", "Cordeiro - Presidente Vargas", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4510, -48.4810)]),
            Onibus("667", "Cordeiro - Ver-o-Peso", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4550, -48.5035)]),
            Onibus("669", "Cordeiro - Castanheira", 45, "Amarelo", [(-1.4092, -48.4671), (-1.4700, -48.4900)])
        ]
        self.dados = {onibus.numero: onibus for onibus in dados_iniciais}
        self.salvar_dados()
```

```
    def buscar_onibus(self, numero: str):
        return self.dados.get(numero)

    def visualizar_onibus(self):
        if not self.dados:
            print("Não há ônibus cadastrados.")
        else:
            for onibus in self.dados.values():
                print(f"Número: {onibus.numero}, Rota: {onibus.rota}, Capacidade:
{onibus.capacidade}, Cor: {onibus.cor}")

    def adicionar_onibus(self):
        try:
            numero = input("Digite o número do ônibus: ")
            if numero in self.dados:
                print("Ônibus já existe.")
                return
            rota = input("Digite a rota do ônibus: ")
            capacidade = int(input("Digite a capacidade do ônibus: "))
            cor = input("Digite a cor do ônibus (em português): ")
            coordenadas = []
            while True:
                resposta = input("Deseja adicionar coordenadas? (s/n): ")
                if resposta.lower() == 's':
                    lat = float(input("Digite a latitude: "))
                    lon = float(input("Digite a longitude: "))
                    coordenadas.append((lat, lon))
```

```
            elif resposta.lower() == 'n':
                break
            else:
                print("Opção inválida. Tente novamente.")
        novo_onibus = Onibus(numero, rota, capacidade, cor, coordenadas)
        self.dados[numero] = novo_onibus
        self.salvar_dados()
        print("Ônibus adicionado com sucesso!")
    except ValueError:
        print("Erro ao adicionar ônibus. Verifique os dados.")

def remover_onibus(self):
    numero = input("Digite o número do ônibus para remover: ")
    if numero in self.dados:
        del self.dados[numero]
        self.salvar_dados()
        print("Ônibus removido com sucesso!")
    else:
        print("Ônibus não encontrado.")

def visualizar_mapa(self, numero: str):
    onibus = self.buscar_onibus(numero)
    if not onibus:
        print("Ônibus não encontrado.")
        return

    if not onibus.coordenadas:
        print("Este ônibus não possui coordenadas registradas.")
```

```
            return

        mapa = folium.Map(location=onibus.coordenadas[0], zoom_start=12)

        folium.PolyLine(onibus.coordenadas, color="blue", weight=5,
opacity=0.7).add_to(mapa)

        for coord in onibus.coordenadas:
            folium.Marker(coord, popup=f"{onibus.numero} - {onibus.rota}").add_to(mapa)

        mapa.save("rota_onibus.html")
        print("Mapa gerado: abra 'rota_onibus.html' no navegador.")
        webbrowser.open("rota_onibus.html")

    def visualizar_todos_mapas(self):
        if not self.dados:
            print("Não há ônibus cadastrados.")
            return

        mapa = folium.Map(location=[-1.4092, -48.4671], zoom_start=12)

        cores_mapa = {
            "amarelo": "yellow",
            "laranja": "orange",
            "vermelho": "red",
            "azul": "blue",
            "verde": "green",
            "roxo": "purple",
```

```
            "preto": "black",
            "cinza": "gray"
        }

        for onibus in self.dados.values():
            if not onibus.coordenadas:
                continue

            cor = cores_mapa.get(onibus.cor.lower(), "blue")

            folium.PolyLine(
                onibus.coordenadas,
                color=cor,
                weight=5,
                opacity=0.7,
                popup=f"{onibus.numero} - {onibus.rota}"
            ).add_to(mapa)

            for coord in onibus.coordenadas:
                folium.Marker(coord, popup=f"{onibus.numero} -
{onibus.rota}").add_to(mapa)

        nome_arquivo = "todas_rotas.html"
        mapa.save(nome_arquivo)
        print(f"Mapa com todas as rotas gerado: abra '{nome_arquivo}' no navegador.")
        webbrowser.open(nome_arquivo)

class SistemaOnibus:
```

```
def __init__(self):
    self.gerenciador = GerenciadorOnibus()
    self.gerenciador.inicializar_dados()

def executar(self):
    while True:
        print("\nSistema de Gerenciamento de Ônibus")
        print("1. Visualizar Ônibus")
        print("2. Buscar Ônibus")
        print("3. Visualizar Rotas no Mapa")
        print("4. Adicionar Ônibus")
        print("5. Remover Ônibus")
        print("6. Visualizar Todas as Rotas no Mapa")
        print("7. Sair")

        opcao = input("Escolha uma opção: ")

        match opcao:
            case "1":
                self.gerenciador.visualizar_onibus()
            case "2":
                self.buscar_onibus()
            case "3":
                self.visualizar_mapa()
            case "4":
                self.gerenciador.adicionar_onibus()
            case "5":
                self.gerenciador.remover_onibus()
```

```
            case "6":
                self.gerenciador.visualizar_todos_mapas()
            case "7":
                print("Saindo do sistema...")
                break
            case _:
                print("Opção inválida. Tente novamente.")

    def buscar_onibus(self):
        numero = input("Digite o número do ônibus para buscar: ")
        onibus = self.gerenciador.buscar_onibus(numero)
        if onibus:
            print(f"Número: {onibus.numero}, Rota: {onibus.rota}, Capacidade:
{onibus.capacidade}, Cor: {onibus.cor}")
        else:
            print("Ônibus não encontrado.")

    def visualizar_mapa(self):
        numero = input("Digite o número do ônibus para visualizar no mapa: ")
        self.gerenciador.visualizar_mapa(numero)

if __name__ == "__main__":
    sistema = SistemaOnibus()
    sistema.executar()
```