O que é PEP?

- **PEP** significa **P**ython **E**nhancement **P**roposal (Proposta de Melhoria do Python).
- São documentos que descrevem novos recursos para o Python, processos, ou fornecem informações para a comunidade Python.

O que é PEP 8?

- **PEP 8** é especificamente o **Guia de Estilo Oficial para Código Python**.
- Ele não define como a lógica do seu programa deve funcionar, mas sim **como o código deve parecer visualmente** e ser estruturado.
- O objetivo principal do PEP 8 é melhorar a **legibilidade** e a **consistência** do código Python. A ideia é que todo código Python se pareça familiar, facilitando a leitura e a colaboração entre diferentes desenvolvedores.

Quais tipos de coisas o PEP 8 define?

O PEP 8 cobre uma vasta gama de convenções de estilo, incluindo:

1. **Layout do Código:**
    - **Indentação:** Usar 4 espaços por nível de indentação (não usar tabs).
    - **Comprimento da Linha:** Limitar todas as linhas a um máximo de 79 caracteres (embora alguns projetos usem um pouco mais, como 88 ou 100, 79 é o padrão PEP 8).
    - **Linhas em Branco:** Como usar linhas em branco para separar funções, classes e blocos lógicos de código.
    - **Importações:** Como organizar as declarações import (geralmente no topo do arquivo, em grupos específicos: bibliotecas padrão, bibliotecas de terceiros, bibliotecas locais).
2. **Nomenclatura (Naming Conventions):**
    - **Variáveis:** minúsculas_com_underline (snake_case).
    - **Funções:** minúsculas_com_underline (snake_case).
    - **Classes:** NomeEmCamelCase (CapWords ou CamelCase).
    - **Constantes:** MAIÚSCULAS_COM_UNDERLINE (UPPER_SNAKE_CASE).
    - **Módulos:** Nomes curtos, minúsculos_com_underline (snake_case).

3. **Comentários:**
    - Como escrever comentários de bloco e comentários em linha.
    - A importância de manter os comentários atualizados com o código.
    - Uso de Docstrings (strings de documentação) para explicar o que funções, classes e módulos fazem.
4. **Espaçamento:**
    - Uso de espaços em branco em expressões e declarações (ex: x = 1 em vez de x=1, a + b em vez de a+b).
5. **Recomendações de Programação:**
    - Como comparar com None (usar is ou is not, ex: if minha_var is None:).
    - Uso de startswith() e endswith() em vez de fatiamento de strings para verificar prefixos/sufixos.

Por que seguir o PEP 8 é importante?

- **Legibilidade:** Código bem formatado é muito mais fácil de ler e entender, tanto por você no futuro quanto por outras pessoas.
- **Consistência:** Garante que o código dentro de um projeto (e até entre diferentes projetos Python) tenha uma aparência semelhante.
- **Manutenção:** Código legível é mais fácil de depurar, modificar e manter.
- **Colaboração:** Facilita o trabalho em equipe, pois todos seguem as mesmas convenções.
- **Profissionalismo:** Demonstra atenção aos detalhes e adesão aos padrões da comunidade.

Seu código está **majoritariamente em conformidade com o PEP 8** e demonstra boas práticas de programação Python. As principais áreas para revisão seriam o **comprimento das linhas** e a **ordem dos imports**. Usar uma ferramenta como flake8 ou black ajudaria a identificar e corrigir automaticamente muitas dessas questões de estilo.

Ordem dos Imports: O PEP 8 recomenda a seguinte ordem:

1. Bibliotecas padrão (standard library)

2. Bibliotecas de terceiros (third-party)

3. Bibliotecas locais (do próprio projeto)
   No seu código, pandas e matplotlib (terceiros) vêm antes
   de os, webbrowser, datetime (padrão). A ordem ideal seria:

```
import datetime
import os
import webbrowser # Ordem alfabética dentro dos grupos é recomendada

import matplotlib.pyplot as plt
import pandas as pd
```