Descrição do Problema:

Rafael é um entusiasta da culinária e adora experimentar novas receitas de diversos países. No entanto, ele enfrenta dificuldades em organizar suas receitas favoritas e muitas vezes acaba perdendo as que mais gostou. Como um programador dedicado, você decidiu ajudá-lo a criar um sistema de Gerenciamento de Receitas para que Rafael possa manter o controle de suas descobertas gastronômicas.

Requisitos funcionais:

1. Cadastro de Receitas: O sistema deve permitir que Rafael cadastre informações sobre cada receita, incluindo nome, país de origem, ingredientes e modo de preparo.

2. CRUD de Receitas: Rafael deve poder adicionar, visualizar, atualizar e excluir receitas de sua coleção através de um menu interativo.

3. Filtragem por País: O sistema deve permitir a visualização das receitas de acordo com o país de origem, facilitando a busca por culinárias específicas.

4. Armazenamento em Banco de Dados: Todas as informações sobre as receitas devem ser armazenadas em um banco de dados para que persistam além da execução do programa (arquivo .txt ou .csv).

5. Lista de Favoritos: Rafael deve poder marcar suas receitas favoritas para acessá-las facilmente em uma lista separada.

6. Sugestão de Receitas Aleatórias: O sistema deve apresentar uma funcionalidade para sugerir receitas aleatórias de diferentes países, incentivando Rafael a experimentar novos pratos.

7. Ter pelo menos uma outra funcionalidade a mais que não está descrita aqui neste documento. Sejam criativos e divirtam-se!

Requisitos não funcionais:

1. Deve ser feito em Python sem o uso de bibliotecas adicionais.
    a. Utilizar a linha de comando para entrada e saída;
    b. Exceções de bibliotecas:
        - os -> os.system("clear") ou “cls”.
2. Os dados devem ser salvos em um arquivo no formato .csv ou .txt;
    a. O trabalho deve ser feito em grupo.
    b. Trabalhos que não forem feitos em grupo perderão 50% da nota.

3. O código deve estar organizado, portanto, deve conter:
    a. Funções para dividir o código de forma lógica e evitar repetições;
    b. Tratamento de exceções, para garantir que seu código esteja pronto para tratar casos inesperados.
    c. Legibilidade do código, incluindo nomeação de variáveis e funções.
4. Deve ser feito um manual do usuário, explicando como utilizar a ferramenta e restrições gerais que a aplicação tenha.
    a. Fiquem à vontade para escolher como será feito esse manual. Pode ser um pdf, site, vídeo, carta...
5. Não será aceito entregas atrasadas.
6. Apresentação:
    a. A equipe deve apresentar o projeto feito para os professores.
    b. Todos envolvidos da equipe devem explicar alguma parte, e perguntas direcionadas serão feitas durante a apresentação.
    c. O manual deve conter o fluxograma do projeto.
7. A entrega será em uma atividade do classroom
    a. O que deve ser entregue:
        - Código da aplicação.
        - Manual do usuário.

Critérios de avaliação:

- Apresentação (50 pontos - nota individual):
    - Participação durante a apresentação do projeto;
    - Perguntas durante a apresentação.
- Código (50 pontos - nota por grupo):
    - Legibilidade e Organização do código;
    - Tratamento de erros;
    - Utilização de Arquivos;
    - Apresentação da ferramenta e manual do usuário;
    - Funcionalidade extra.