Lista de Exercícios 3 - POO-II - INE5404 - Dicionários

Exercício 1. Escreva uma função que conta a frequência de ocorrência de cada palavra em um texto (arquivo txt) e armazena tal quantidade em um dicionário, onde a chave é a vogal considerada.

Exercício 2. Escreva uma função que apaga do dicionário anterior, todas as palavras que sejam *'stopwords*'. Ver https://gist.github.com/alopes/5358189

Exercício 3. Escreva um programa que lê duas notas de vários alunos e armazena tais notas em um dicionário, onde a chave é o nome do aluno. A entrada de dados deve terminar quando for lida uma string vazia como nome. Escreva uma função que retorna a média do aluno, dado seu nome.

Exercício 4. Uma pista de Kart permite 10 voltas para cada um de 6 corredores. Escreva um programa que leia todos os tempos em segundos e os guarde em um dicionário, onde a chave é o nome do corredor. Ao final diga de quem foi a melhor volta da prova e em que volta; e ainda a classificação final em ordem (1o o campeão). O campeão é o que tem a menor média de tempos.

Exercício 5. Escreva um programa para armazenar uma agenda de telefones usando um dicionário. Cada pessoa pode ter um ou mais telefones e a chave do dicionário é o nome completo da pessoa. Seu programa deve ter as seguintes funções:

incluir_novo_nome – essa função acrescenta um novo nome na agenda, com um ou mais telefones. Ela deve receber como argumentos o nome e os telefones.

incluir_telefone – essa função acrescenta um telefone em um nome existente na agenda. Caso o nome não exista na agenda, você deve perguntar se a pessoa deseja incluí-lo. Caso a resposta seja afirmativa, use a função anterior para incluir o novo nome.

excluir_telefone – essa função exclui um telefone de uma pessoa que já está na agenda. Se a pessoa tiver apenas um telefone, ela deve ser excluída da agenda.

excluir_nome – essa função exclui uma pessoa da agenda.

consultar_telefone – essa função retorna os telefones de uma pessoa na agenda.

Exercício 6. Criar 10 frozensets com 30 números aleatórios cada, e construir um dicionário que contenha a soma de cada um deles.

Exercício 7. Implemente a classe MeuDicionario usando duas listas, uma para chave e uma para valores. Deve ser possível inserir, remover e acessar elementos no dicionário (use magic methods), além de verificar se um valor encontra-se no dicionário (em caso positivo, retorna a chave correspondente, caso contrário retorna False).

Exercício 8. Crie uma classe MatrizEsparsa que pode ser construida das duas formas abaixo, e contenha métodos para mostrar a matriz no formato String e também no formato de dicionário.

1) receber uma dupla (tupla com 2 elementos) indicando quantas linhas e colunas tem a matriz, e um dicionário com duplas como chave (linha, coluna) e um valor numérico.
2) Receber uma string no seguinte formato:

matriz = '''0 8 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 2 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0
0 0 0 7 0 0 0 0 0
0 0 0 0 0 0 4 0 0'''

*dica: use os métodos splitlines() e split da classe String, e o método get() da classe dict