Ifes Campus Serra
BSI – Bacharelado de Sistemas de Informação
Programação II

Lista de Exercícios: Arquivos em Python

Obs: quando não for fornecido pelo professor, crie as bases de dados apropriadas para a realização dos testes de implementação.

Enunciados

1. Crie um arquivo de entrada a partir de um texto qualquer capturado da internet. Um artigo de jornal, por exemplo. Nomeie esse arquivo como documento1.txt. Construa um programa python para remover os acentos desse arquivo. O arquivo deve ser lido linha a linha. O conteúdo sem acentos deve ser escrito em um novo arquivo. Cada linha lida da entrada deve ter os seus acentos removidos, quando então é gravada no arquivo de saída (pelo programa principal). Construa uma função para remover os acentos. A função deve ser chamada para cada linha lida do arquivo documento1.txt. A função deve retornar o texto de entrada sem os acentos.
2. Para fazer este exercício você precisa baixar o arquivo de dados bdalunos.txt. O arquivo está disponível no mesmo local da lista de exercícios, na sala Moodle da disciplina. O arquivo contém 20 linhas, cada uma representando 1 aluno. Em cada linha constam os seguintes dados separados por ; (ponto e vírgula): cpf do aluno, nome, endereço e idade.

   Construa um programa Python que leia este arquivo linha a linha, faça o devido processamento e exiba na tela os resultados. O processamento deve fazer os seguintes cálculos:

   a) Calcular a média das idades dos alunos.
   b) Obter o nome e a idade do aluno com a maior idade.
   c) Obter o nome e a idade do aluno com a menor idade.
   d) Obter o nome do estado da federação que mais aparece nos endereços

3. Faça um programa que leia um arquivo texto contendo uma lista de endereços IP e gere um outro arquivo, contendo um relatório dos endereços IP válidos e inválidos.

   O arquivo de entrada possui o seguinte formato (conteúdo exemplo):

```
200.135.80.9
192.168.1.1
8.35.67.74
257.32.4.5
85.345.1.2
1.2.3.4
9.8.234.5
192.168.0.256
```

Sugestão: para testar o seu programa, gere um arquivo texto de entrada copiando e colando os valores do conteúdo de exemplo.

O arquivo de saída possui o seguinte formato:

```
[Endereços válidos:]
200.135.80.9
192.168.1.1
8.35.67.74
1.2.3.4

[Endereços inválidos:]
257.32.4.5
85.345.1.2
9.8.234.5
192.168.0.256
```

4. Um arquivo chamado bdveiculos.txt possui os dados de 150 veículos. Os dados de cada veículo são: placa, modelo, marca, quilometragem. Cada dado de veículo ocorre em apenas 1 linha de arquivo. Assim, a cada 4 linhas do arquivo temos os dados de um veículo completo (ver figura após o enunciado). Sabendo disto, programe o que é pedido a seguir.

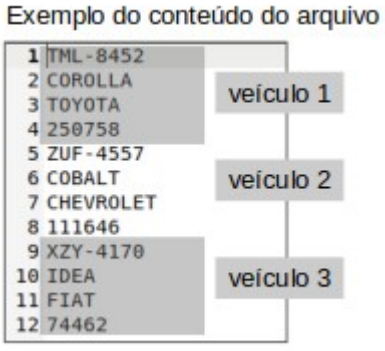
   a) Construa um programa Python que leia todo o arquivo bdveiculos.txt, linha a linha, e armazene o conteúdo em uma lista com o seguinte layout (função main()):

   ```
   [[<dados veículo 1>],[<dados veículo 2],.., [<dados último veículo>]]
   ```

   b) No mesmo programa, construa uma função chamada **filtro** com o seguinte cabeçalho: **filtro(<lista de veículos do item a>, <string marca de automovel>)**.

   A lógica da função deve abrir um arquivo texto de saída com nome igual ao 2º parâmetro (marca de automóvel). Em seguida, a função deve percorrer a lista de veículos e escrever no arquivo de saída somente os dados dos veículos com marca igual ao valor do último parâmetro. No arquivo de saída, os dados de cada veículo devem ocorrer todos em uma mesma linha, separados por vírgula.

   c) No programa principal, como última tarefa, chamar a função filtro 4 vezes para salvar em arquivos separados os veículos das marcas FIAT, TOYOTA, FORD e RENAULT.



5. Um arquivo chamado bdmatrizes.txt possui os dados de uma quantidade indeterminada de matrizes de inteiros. A organização do arquivo está representada pela figura 1.a abaixo. Construa um programa Python onde a funçao main() deve ler os dados de cada matriz do arquivo de entrada, 1 matriz por vez.

   A função main() deve guardar os elementos de cada matriz em uma lista de inteiros. Em seguida, as dimensões da matriz (quantidades de linhas e colunas) e a lista de inteiros devem ser passadas como argumentos para uma função chamada **organiza(linha, coluna, lst)**. Esta função processa os argumentos de entrada e retorna uma lista de listas com o seguinte layout: A lista maior contém uma quantidade de listas igual a quantidade de linhas da matrix; e as listas internas contém os elementos de cada linha da matriz. Ver figura 1.b e 1.c.

   Por último, a função main() deve invocar a função **print_matriz(<lista de listas representando 1 matriz>)**. Esta função imprime, no console, a matriz passada como argumento, com o visual característico de matrizes. Ver figura 1.d.

   Todos os passos descritos nos parágrafos anteriores devem ser repetidos até todas as matrizes em bdmatrizes.txt serem processadas.

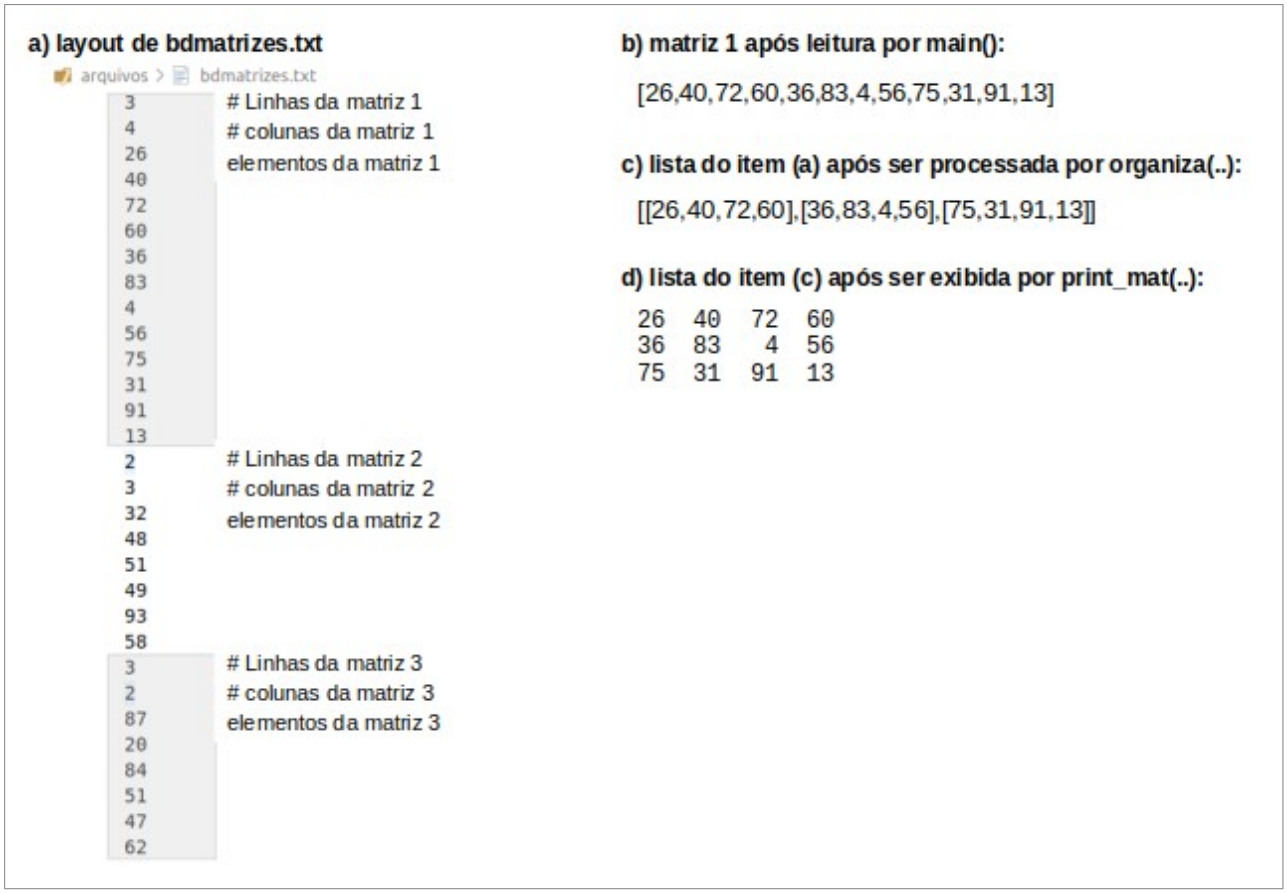


Figura 1: exemplos das estruturas de dadas mencionadas no enunciado.

Bons estudos !!