Capítulo 10

Ficheiros

1. Escreva a função `conta_linhas` que dada uma cadeia de caracteres com o nome de um ficheiro, devolve o número de linhas que ocorrem no ficheiro e que não estão em branco, ou seja, apenas com o caracter de fim de linha.

2. Escreva uma função que recebe uma cadeia de caracteres, que contém o nome de um ficheiro, lê esse ficheiro, linha a linha, e calcula quantas vezes aparece cada uma das vogais. A sua função deve devolver um dicionário cujas chaves são as vogais e os valores associados correspondem ao número de vezes que a vogal aparece no ficheiro. Apenas conte as vogais que são letras minúsculas. Por exemplo,

   ```
   >>> conta_vogais('testevogais.txt')
   {'a': 36, 'u': 19, 'e': 45, 'i': 16, 'o': 28}
   ```

3. Escreva uma função que recebe como argumentos os nomes de dois ficheiros e que escreve no ficheiro correspondente ao segundo argumento o conteúdo do ficheiro correspondente ao primeiro argumento mas por ordem invertida, ou seja, a primeira linha do primeiro ficheiro será a última linha do segundo ficheiro.

4. Escreva a função `concatena` que recebe uma lista de cadeias de caracteres, cada uma correspondendo ao nome de um ficheiro, e uma cadeia de caracteres, correspondendo ao nome do ficheiro de saída, e concatena o conteúdo dos primeiros ficheiros no ficheiro de saída. Por exemplo, se o ficheiro `fich1` contiver o texto:

   ```
   Um ficheiro para fazer uns testes.
   Este ficheiro contem duas linhas.
   ```

   e o ficheiro `fich2` contiver o texto:

   ```
   Outro ficheiro para fazer os mesmos testes.
   ```

```
Este ficheiro contem mais uma linha
alem desta.
```

é produzida a seguinte interacção:

```
>>> concatena(['fich1', 'fich2'], 'saida')
>>>
```

em que o ficheiro saida contém o texto:

```
Um ficheiro para fazer uns testes.
Este ficheiro contem duas linhas.
Outro ficheiro para fazer os mesmos testes.
Este ficheiro contem mais uma linha
alem desta.
```

5. Escreva a função `procura` que recebe duas cadeias de caracteres, em que a primeira corresponde a uma palavra a procurar e a segunda contém o nome de um ficheiro. A sua função deve escrever no ecrã as linhas do ficheiro que contêm a palavra a procurar. Por exemplo, se o ficheiro `fich` contiver o texto:

   ```
   Outro ficheiro para fazer os mesmos testes.
   Este ficheiro contem mais uma linha
   alem desta.
   ```

   é produzida a seguinte interacção:

   ```
   >>> procura('ficheiro', 'fich')
   Outro ficheiro para fazer os mesmos testes.
   Este ficheiro contem mais uma linha
   >>>
   ```

6. Escreva a função `corta` que recebe duas cadeias de caracteres, uma contendo o nome de um ficheiro de entrada, outra contendo o nome do ficheiro de saída, e um número inteiro não negativo n, e escreve os n primeiros caracteres do ficheiro de entrada no ficheiro de saída, no caso de o ficheiro conter mais que n caracteres, ou todo o conteúdo do ficheiro de entrada no ficheiro de saída, no caso contrário. Por exemplo, se o ficheiro `fich` contiver o texto:

   ```
   Um ficheiro para fazer uns testes.
   Este ficheiro contem duas linhas.
   ```

   é produzida a seguinte interacção:

```
>>> corta('teste', 'saida', 20)
>>>
```

em que o ficheiro saida contém o texto:

```
Um ficheiro para faz
```

7. Escreva a função `ordena_ficheiro` que recebe como argumento uma cadeia de caracteres correspondendo ao nome de um ficheiro e escreve no ecrã as linhas do ficheiro ordenadas por ordem crescente. Por exemplo, se o ficheiro `fich` contiver o texto:

   ```
   Outro ficheiro para fazer os mesmos testes.
   Este ficheiro contem mais uma linha
   alem desta.
   ```

   é produzida a seguinte interacção:

   ```
   >>> ordena('fich')
   Este ficheiro contem mais uma linha
   Outro ficheiro para fazer os mesmos testes.
   alem desta.
   ```

8. Escreva a função `divide` que recebe uma cadeia de caracteres, que contém o nome do ficheiro de entrada, e um inteiro n e divide o ficheiro em dois ficheiros, um primeiro, cujo nome é o nome do ficheiro de entrada seguido de 0, em que cada linha contém os n primeiros caracteres da correspondente linha do ficheiro e outro, cujo nome é o nome do ficheiro de entrada seguido de 1, em que cada linha contém os restantes caracteres da linha correspondente no ficheiro original. Por exemplo, se o ficheiro `fich` contiver o texto:

   ```
   Um ficheiro para fazer uns testes.
   Este ficheiro contem duas linhas.
   ```

   é produzida a seguinte interacção:

   ```
   >>> divide('fich', 20)
   >>>
   ```

   em que o ficheiro `fich0` contém o texto:

   ```
   Um ficheiro para faz
   Este ficheiro contem
   ```

   e o ficheiro `fich1` contém o texto:

```
er uns testes.
duas linhas.
```

9. Escreva a função `separa` que recebe uma cadeia de caracteres, que contém o nome do ficheiro de entrada, outra cadeira de caracteres com apenas um caracter e um inteiro n e divide o ficheiro em dois ficheiros, um primeiro, cujo nome é o nome do ficheiro de entrada seguido de 0, e outro, cujo nome é o nome do ficheiro de entrada seguido de 1. O conteúdo do segundo ficheiro é o mesmo do ficheiro de entrada a que foi retirado o texto de cada linha entre a n-ésima ocorrência do caracter, incluindo o caracter, e a $n+1$-ésima ocorrência do caracter, excluindo o caracter. O primeiro ficheiro tem em cada linha o texto que foi retirado ao texto de entrada para produzir o primeiro texto. Por exemplo, se o ficheiro `fich` contiver o texto:

   ```
   Outro ficheiro para fazer os mesmos testes.
   alem desta.
   Este ficheiro contem mais uma linha
   ```

   é produzida a seguinte interacção:

   ```
   >>> separa('fich', 'h', 1)
   >>>
   ```

   em que o ficheiro `fich0` contém o texto:

   ```
   Outro fic
   alem desta.
   Este ficha
   ```

   e o ficheiro `fich1` contém o texto:

   ```
heiro para fazer os mesmos testes.
   heiro contem mais uma lin
   ```