TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 2ª feira 09:00-10:30
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `iguais` que recebe duas cadeias de caracteres correspondente a dois ficheiros de entrada e devolve `True` se os ficheiros forem iguais e `False` caso contrário. As comparações devem ser *case insensitive*. Por exemplo, a seguinte interação produz `True`.

```
>>> iguais('texto.txt', 'texto.txt')
True
```

Solução 1 (comparação por linha):

```
def iguais(file1, file2):
    fh1 = open(file1, 'r')
    lines1 = fh1.readlines()
    fh1.close()
    fh2 = open(file2, 'r')
    lines2 = fh2.readlines()
    fh2.close()
    if len(lines1) != len(lines2):
       return False
    lines1 = list(map(lambda l: l.upper(), lines1))
    lines2 = list(map(lambda l: l.upper(), lines1))
    for i in range(len(lines)):
       if lines1[i] != lines2[i]:
            return False
    return True
```

Solução 2 (comparação única string):

```
def iguais(file1, file2):
   fh1 = open(file1, 'r')
   str1 = fh1.read()
   fh1.close()
   fh2 = open(file2, 'r')
   str2 = fh2.read()
   fh2.close()
   return str1.lower() == str2.lower()
```

Solução 3 (comparação única string e with):

```
def iguais(infile1, infile2):
   with open(infile1, 'r') as fr1, open(infile2, 'r') as fr2:
       return fr1.read().lower() == fr2.read().lower()
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 2ª feira 10:30-12:00
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `filtra_positivas` que recebe duas cadeias de caracteres correspondentes a um ficheiro de entrada e outro de saída. Cada linha do ficheiro de entrada contém um número de aluno e uma nota, separados por uma vírgula. A sua função deve devolver o número total de notas maiores ou iguais a 9.5 e escrever no ficheiro de saída todas as linhas correspondentes a estas notas. Por exemplo, se o ficheiro `texto.txt` tiver:

```
x11111,17
x11112,9.5
x11113,18
x11114,8
x11115,10
X11116,5
```

A seguinte interacção devolve o número *4* e escreve no ficheiro `positivas.txt`

```
>>> filtra_positivas('texto.txt','positivas.txt')
4
```

O conteúdo do ficheiro `positivas.txt` passa a ser o seguinte:

```
x11111,17
x11112,9.5
x11113,18
X11115,10
```

Solução:

```
def filtra_positivas(fname_in, fname_out):
    f_in = open(fname_in, 'r')
    lines = f_in.readlines()
    f_in.close()

    f_out = open(fname_out, 'w')
    num_positivas = 0
    for line in lines:
       entries = line.split(',')
       nota = eval(entries[1])
       if nota >= 9.5:
           f_out.write(line)
           num_positivas += 1

    f_out.close()
    return num_positivas
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 4ª feira 08:00-09:30
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `recorta` que recebe uma cadeia de caracteres contendo o nome de um ficheiro de entrada e um número inteiro não negativo n, e divide o ficheiro de entrada em vários ficheiros de saída de n caracteres cada (o último poderá ter menos caracteres), cujos nomes se constroem com o nome do ficheiro de entrada seguido de números inteiros positivos, de forma a que a concatenação dos ficheiros com nomes sucessivos produz um ficheiro com o conteúdo do ficheiro original. Por exemplo, se o ficheiro `fich` contiver o texto:

```
Um ficheiro para fazer uns testes.
```

e se for produzida a interacção:

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

o ficheiro `fich1` contém o texto: `Um ficheiro para faz` e o ficheiro e o `fich2` contém o texto: `er uns testes.`

Solução 1:

```
def recorta(nome, n):
      f_in = open(nome, 'r')
      f_num = 1
      chs = f_in.read(n)

      while chs: # O ciclo termina quando o conteudo foi todo lido
            f_out = open(nome+str(f_num), 'w')
            f_out.write(chs)
            f_out.close()

            f_num += 1
            chs = f_in.read(n)
```

Solução 2:

```
def recorta(nome, n):
   f = open(nome, 'r')
   fileno = 1
   written = 0
   f1 = open(nome + str(fileno), "w")
```

```
    while True:
        ch = f.read(1)
        if not ch:
            break
        f1.write(ch)
        written +=1
        if written == n:
            f1.close()
            fileno += 1
            written = 0
            f1 = open(nome + str(fileno), "w")
    f1.close()
```

Solução 3:

```
def recorta(nome, n):
    with open(nome, 'r') as fr:
        data = fr.read()

    fileno = 1
    for i in range(0, len(data), n):
        with open(nome + str(fileno), 'w') as fw:
            fw.write(data[i:i + n])
        fileno += 1
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 5ª feira 10:30-12:00
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `junta_ficheiros_ordenados` que recebe três cadeias de caracteres correspondendo a dois ficheiros de entrada e um de saída. Cada um dos ficheiros de entrada contém números inteiros positvos ordenados por ordem crescente, contendo cada linha apenas um número. A sua função deve produzir um ficheiro ordenado de números (contendo um número por linha) correspondente à junção dos números existentes nos dois ficheiros de entrada. Por exemplo, se é produzida a seguinte interacção:

```
>>> junta_ficheiros_ordenados('fich1', 'fich2', 'fich3')
>>>
```

Se o ficheiro `fich1` contiver:

```
2
4
5
12
25
```

e o ficheiro `fich2` contiver:

```
1
7
15
```

Então o ficheiro `fich3` contém:

```
1
2
4
5
7
12
15
25
```

Solução:

```
def junta_ficheiros_ordenados(infile1, infile2, outfile):
   with open(infile1, 'r') as fr1, open(infile2, 'r') as fr2, open(outfile,
'w') as fw:
       data = [int(e) for e in fr1] + [int(e) for e in fr2]
       for e in sorted(data):
           fw.write(str(e)+'\n')
```

Solução:

```
def junta_ficheiros_ordenados(infile1, infile2, outfile):
   with open(infile1, 'r') as fr1, open(infile2, 'r') as fr2,
open(outfile, 'w') as fw:

        num1 = fr1.readline()
        num2 = fr2.readline()

        while num1 != '' and num2 != '':
            if int(num1) < int(num2):
                fw.write(num1)
                num1 = fr1.readline()
            else:
                fw.write(num2)
                num2 = fr2.readline()

        while num2 != '' and num1 == '':
            fw.write(num2)
            num2 = fr2.readline()

        while num1 != '' and num2 == '':
            fw.write(num1)
            num1 = fr1.readline()
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 6ª feira 12:30-14:00
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva uma função, `conta_duplicacoes`, que recebe uma cadeia de caracteres, correspondente ao nome de um ficheiro, lê esse ficheiro, linha a linha, e devolve um tuplo, cujo primeiro elemento é o número de linhas do ficheiro e segundo elemento é um dicionário cujos índices são caracteres e cujo valor corresponde ao número de vezes o esse caracter apareceu duas vezes seguidas no ficheiro. Por exemplo, se o ficheiro `cdex.txt` contiver

```
hggsf rr sftra
hhsr
pooooyy rtt
Hfddsa
```

então

```
>>> conta_duplicacoes('cdex.txt')
(4, {'d': 1, 'g': 1, 'h': 1, 'o': 3, 'r': 1, 't': 1, 'y': 1})
```

Solução:

```
def conta_duplicacoes(nome):
    f = open(nome, 'r')

    n_linhas = 0
    res = {}
    for linha in f:
        n_linhas += 1
        for c in range(1, len(linha)):
            if linha[c] == linha[c - 1]:
                res[linha[c]] = res.get(linha[c], 0) + 1

    f.close()
    return(n_linhas, res)
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 6ª feira 14:00-15:30
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `cria_f_index` que recebe uma cadeia de caracteres correspondente a um ficheiro de entrada e devolve um dicionário cujas chaves correspondem ao primeiro caracter das linhas e cujo valor corresponde ao número de linhas que começam por um determinado caractere. Por exemplo, se o ficheiro texto.txt tiver:

```
Isto é um exemplo
De um ficheiro
Onde existem 4 linhas
De palavras sem nexo
```

É produzida a seguinte interacção:

```
>>> cria_f_index('texto.txt')
  {'O': 1, 'I': 1, 'D': 2}
```

Solução:

```
def cria_f_index(file):
    f = open(file, 'r', encoding='UTF-8')
    lines = f.readlines()
    f.close()
    d = {}
    for line in lines:
       if not line[0] in d:
            d[line[0]] = 0

       d[line[0]] = d[line[0]] + 1
    return d
```

TÉCNICO LISBOA

Fundamentos de Programação - 2018/2019 Aula Prática 11 (30 minutos) Turno 6ª feira 15:30-17:00
Nome:
Número:
Data:
Curso:

Capítulo 10 - Ficheiros

Escreva a função `cria_f_index` que recebe uma cadeia de caracteres correspondente ao nome de um ficheiro de entrada e devolve um dicionário cujas chaves correspondem ao primeiro caracter das linhas e cujo valor corresponde ao número de linhas que começam por um determinado caractere. Por exemplo, se o ficheiro `texto.txt` tiver:

```
Isto é um exemplo
De um ficheiro
Onde existem 4 linhas
De palavras sem nexo
```

É produzida a seguinte interacção:

```
>>> cria_f_index('texto.txt')
  {'O': 1, 'I': 1, 'D': 2}
```

Solução:

```
def cria_f_index(file):
    f = open(file, 'r', encoding='UTF-8')
    lines = f.readlines()
    f.close()
    d = {}
    for line in lines:
       if not line[0] in d:
            d[line[0]] = 0

       d[line[0]] = d[line[0]] + 1
    return d
```