

Fundação Universidade Federal do Rio Grande
Centro de Ciências Computacionais
Algoritmos e Estruturas de Dados I

1 Laboratório de Arquivos

Faça o conjunto de funções descrito abaixo. Para envio não é necessário um programa teste, apenas as funções serão avaliadas. Siga atentamente as instruções de cada função.

1.1 Instruções

- Altere o arquivo em conjunto chamada arquivos.py.
- Não altere o nome do arquivo.
- Você deve preencher o código das funções nas posições indicadas.
- Não é permitido trocar o nome das funções ou o número de parâmetros delas.
- Cada função deve abrir, ler, escrever e fechar os arquivos necessários.
- Todas as funções devem tratar o erro para o caso do arquivo de entrada não existir.
- Arquivos que não forem devidamente fechados, serão descontados.
- Caso seja preciso, você pode declarar outras funções.
- O programa para testar as funções fica a seu critério, ele não será avaliado.
- Envie apenas o arquivo arquivos.py com as devidas alterações.

1.2 Funções

1. **(2,0)** Função `checkDoisPontos(arquivo)`: Recebe o nome de um arquivo python e para cada linha que contenha um comando `if` ou `elif`, mostrar na tela, o número da linha, a linha toda e “[OK]” caso tenha “:” ao final da linha e “[ERRO]” caso o programador tenha esquecido os dois pontos ao final do `if` ou do `elif`.

Atenção:

- Considere que pode haver um comentário depois dos dois pontos.
- Considere que podem haver linhas com os caracteres “if” mas esse não representam o comando `if`. Por exemplo, isso pode acontecer caso uma variável tenha “if” como parte do seu nome `tiff=0`. Nesse caso a sua função deve ignorar a linha.

Exemplos de Saída:

```
Linha 10: if x>10: [OK]
Linha 12: if(A==0 and B==10) [ERRO]
Linha 1001: elif tiff==ifsul: # Comentário qualquer [OK]
```

A função deve retornar o número de erros encontrados.

2. **(2,0)** Função `toMaiusculo(arquivo)` deve receber o nome de um arquivo texto e criar um arquivo de saída com o nome do arquivo mais a letra “M” ao final do nome do arquivo e gravar todo o conteúdo original em maiúsculo. A função deve retornar o número de caracteres convertidos para maiúsculo.

Ex.: Dado o arquivo “lorem.txt” com:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In sed quam vel lectus ultricies tincidunt ac id lacus. Aliquam elit ipsum, luctus id ipsum at, maximus elementum lectus.

Sua função deve criar “loremM.txt” com:

LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT. IN SED QUAM
VEL LECTUS ULTRICIES TINCIDUNT AC ID LACUS. ALIQUAM ELIT IPSUM, LUCTUS
ID IPSUM AT, MAXIMUS ELEMENTUM LECTUS.

3. **(2,0)** Função `topScores(nome,score)` que deve gerenciar as dez maiores pontuações de um jogo em um arquivo.

A função deve criar (caso não exista) ou alterar (caso já exista) um arquivo chamado “topscores.txt” cada vez que for chamada. A cada chamada ela deve abrir o arquivo existente, verificar se a nova pontuação está entre os top 10 e, caso esteja, alterar o arquivo existente, armazenando o nome e a pontuação do jogador.

Atenção, inicialmente o arquivo não existe (ele deve ser criado na primeira chamada) e nas chamadas subsequentes da função, basta apenas adicionar a nova pontuação ao arquivo. A partir do momento que o arquivo já tenha 10 pontuações, a função deve gerenciar para que apenas as 10 maiores estejam armazenadas. O arquivo não precisa manter as pontuações ordenadas, mas fica a seu critério.

A função deve retornar `True` se o arquivo foi alterado (teve um score Top 10) ou `False` caso contrário.

Os testes serão feitos chamando a função pelo menos 20 vezes, alternando valores de pontuação altas e baixas.

1.3 Manipulação de Imagens em Tons de Cinza (PGM)

Um arquivo PGM (Portable Gray Map) é um tipo de arquivo texto que representa uma imagem. O arquivo é constituído de um cabeçalho e uma matriz que representa o valor de cada pixel na imagem. Para abrir um arquivo PGM, você pode usar o Gimp (<https://www.gimp.org>), o IrfanView (www.irfanview.com) ou o site <http://paulcuth.me.uk/netpbm-viewer/>, etc. No caso do site indicado, figuras muito pequenas podem ser difíceis de visualizar.

Exemplo de arquivo .pgm e sua respectiva representação gráfica:

P2	
3 3	
10	
5 0 0	
0 5 0	
0 0 5	

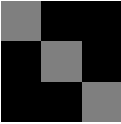


Figura 1: Arquivo .pgm e sua respectiva representação gráfica

Onde as três primeiras linhas do arquivo são o cabeçalho da imagem. A primeira linha contém uma palavra-chave “P2” que é obrigatória e identifica o tipo de imagem PGM. A segunda linha contém dois números que correspondem ao número de colunas e linhas da imagem (matriz), respectivamente.

A terceira linha contém um número que é o maior número da imagem (`maxval`). Nenhum valor de pixel pode ser maior que ele. Os demais números do arquivo correspondem aos tons de cinza da imagem armazenados em forma de uma matriz de inteiros. Cada tom de cinza é um número entre 0 e `maxval`, com 0 indicando “preto” e `maxval` indicando “branco”.

Na verdade um arquivo PGM, não tem muita distinção entre espaços e quebras de linha (“\n”). As seguintes versões da mesma imagem são válidas:

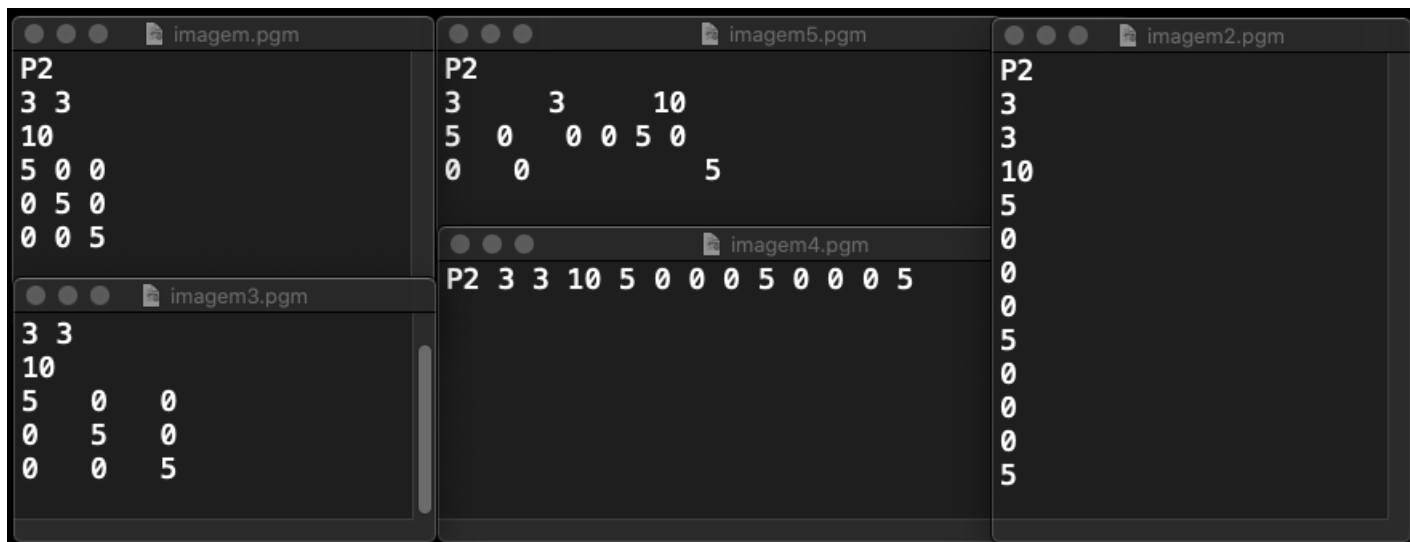


Figura 2: Diferentes versões da mesma imagem

Ou seja em arquivos do tipo .PGM, espaços e quebras de linha (“\n”) tem o mesmo tratamento e múltiplos espaços são ignorados.

No arquivo `arquivos.py` existe uma função que remove os múltiplos espaços de uma string. Você pode usá-la. Não esqueça de tratar também as quebras de linha (“\n”).

A partir disso, crie a seguinte função:

4. (2,0) Função `espelhaPGM(imagem,novaImagem)` que espelha uma imagem. A função recebe o nome de um arquivo de imagem .pgm e cria uma nova imagem (nome em `novaImagem`) .pgm com a imagem espelhada. A função deve retornar `True` se a operação foi realizada com sucesso e `False` caso contrário. Exemplo:

Entrada: `imagem.pgm`

```
P2
3 3
10
5 0 0
0 5 0
0 0 5
```



Saída: `novaImagem.pgm`

```
P2
3 3
10
0 0 5
0 5 0
5 0 0
```



Figura 3: Entrada – > Saída

ATENÇÃO: Não esqueça das quebras linha ao final de cada linha (inclusive a última).

1.4 Manipulação de Imagens Coloridas (PPM)

De forma similar as figuras do tipo PPM (Portable Pixel Map) representam imagens coloridas. As figuras do tipo PPM contém o identificador “P3”, e assim como nos arquivos PGM, a segunda linha contém o número de colunas e linhas da matriz, respectivamente, e a terceira linha contém um número que é o maior valor de pixel permitido para aquela imagem (`maxval`) e representa o branco.

O que diferencia arquivos PPM de PGM, é que nos arquivos PPM, cada elemento da matriz é representado por três valores: grau do tom vermelho, grau do tom verde e grau do tom azul. Esse tipo de representação para cores é conhecido como RGB (Red, Green and Blue).

Exemplo de arquivo PPM do tipo P3 e sua respectiva representação gráfica:

```
P3
3 2
255
255 0 0 0 255 0 0 0 255
255 255 0 255 255 255 0 0 0
```

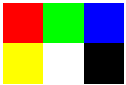


Figura 4: Exemplo de arquivo .ppm e sua respectiva representação gráfica

A partir disso, faça a seguinte função:

5. **(2,0)** Faça a função `rotaciona90PPM(imagem,novaImagem)` que recebe o nome de uma imagem .ppm e cria uma nova imagem .ppm (nome dado em `novaImagem`) rotacionada em 90°. A função deve retornar **True** se a operação foi realizada com sucesso e **False** caso contrário. Exemplo:

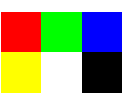
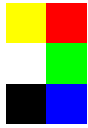
Entrada: <code>imagem.ppm</code>		Saída: <code>novaImagem.ppm</code>	
P3		P3	
3 2		2 3	
255		255	
255 0 0 0 255 0 0 0 255		255 255 0 255 0 0	
255 255 0 255 255 255 0 0 0		255 255 255 0 255 0	
		0 0 0 0 0 255	

Figura 5: Entrada — > Saída

Note que nesse exemplo o número de linhas e colunas são diferentes.

ATENÇÃO: Não esqueça das quebras linha ao final de cada linha (inclusive a última).