MAB114 – Computação II – EP1

1. [Opcional] Desejamos criar um programa que permita que um atleta guarde e tenha acesso a todos os tempos que realizou nos treinos para uma prova de corrida. Os tempos são guardados em minutos em um arquivo de texto. Cada nova linha do arquivo representa um novo treino. Crie uma classe para realizar a interação com o usuário e a manipulação do arquivo de tempo. A classe deve ter os seguintes atributos e métodos:
    a. Método construtor - recebe como argumento o nome do arquivo que será utilizado e cria um atributo para guardar esse argumento.
    b. Método para inserir o tempo – interage com o usuário para pedir o tempo em minutos. Escreve o valor passado no final do arquivo cujo nome está guardado no atributo criado no método construtor. Garante que uma nova linha é criada no arquivo.
    c. Método para calcular a média dos tempos –calcula a média de todos os tempos do arquivo cujo nome foi passado no método construtor. Retorna o float que representa a média. Considere que o arquivo existe e que todas as linhas do arquivo possuem números. Ou seja, não é necessário fazer nenhum tratamento de erro.
    d. Método que calcula a quantidade de treinos – retorna a quantidade de treinos que foi realizada. Lembre-se que cada novo treino está em uma nova linha e ao inserir um novo tempo criamos uma linha nova.
    e. Método para corrigir o último número que foi inserido – o método deve mostrar ao usuário qual foi o último tempo inserido, perguntar para ele qual é o tempo correto e reescrever o arquivo de forma a inserir o tempo correto no lugar do último.
    f. Método para encontrar o tempo mínimo – busca em cada linha do arquivo qual é o menor tempo e retorna o número do treino com menor tempo e o tempo em si. O número do treino é igual ao número da linha que se encontra o menor tempo.

Para as questões a seguir, considere os seguintes imports:

```
import numpy as np
import numpy.matlib as npmat
import matplotlib.image as mpimg
```

2. Considere o seguinte trecho de um arquivo cvs que contém as de notas dos alunos na primeira prova, segunda prova e a média das duas notas:

```
notas.csv
1  Nome,P1,P2,Media
2  Ana,7.2,9.3,8.3
3  Pedro,5.7,7.1,6.4
4  Luiz,5.3,7.1,6.2
5  Maria,9.8,6.0,7.9
6  Julia,7.4,7.7,7.6
7  Paulo,7.2,8.6,7.9
8  Joao,5.2,9.9,7.6
9  Marcos,4.3,4.7,4.5
10 Camila,8.2,4.0,6.1
11 Gustavo,7.0,7.6,7.3
12 Caio,8.7,6.6,7.6
13 Larissa,5.2,6.9,6.1
14 Henrique,6.1,8.0,7.1
```

Utilizando o numpy, crie um código em Python para ler o arquivo csv e manipular os dados do array criado. Calcule e imprime a média de notas da P1 e a média de notas da P2. Crie um array do numpy que contém somente os nomes e notas dos alunos com média menor

que 7.0. Imprima o nome dos alunos com média menor que 7.0 e a porcentagem de alunos com média menor que 7.0 em relação à quantidade total de alunos da tabela.

3. Uma forma de melhorar o contraste de uma imagem muito escura é aplicar a cada pixel as seguintes funções em sequência:

$$p'_{x,y} = \sqrt{p_{x,y}}$$

$$p''_{x,y} = (p'_{x,y} - p'_{min})/(p'_{max} - p'_{min})$$

Onde $p_{x,y}$ é o valor do pixel na linha x e coluna y da matriz, p'_{min} é o valor do menor pixel da matriz após a raiz quadrada e p'_{max} é o valor do maior pixel da matriz após a raiz quadrada.
Utilizando o numpy, e sabendo que o numpy possui uma função de raiz quadrada (sqrt) escreva um código em Python que aplica as funções acima a cada pixel da imagem im e cria uma nova variável que pode ser interpretada pelo numpy como uma imagem. Considere que im é definida por:

```
im = mpimg.imread('extintor.png')
```