

Tarefa T05 – Ler e escrever arquivos

Considere que os dados coletados por sensores em uma cidade estão em um arquivo texto (.txt) contendo o **nome da cidade** (com palavras separadas por apenas um espaço em branco) e uma sequência dados de cada sensor (**nome do sensor**, **número de leituras**, e os **valores** registrados de cada leitura). Considere o seguinte formato:

```
Rio de Janeiro:
Sensor A, 10 leituras: 30 41 60
65 40 20 10 15
3 2
Sensor B, 12 leituras: 20 22 50 51 50 25 10 15 11 5 2 1
Sensor C, 5 leituras: 5 2 3 4 5
```

Note que há separadores (dois pontos, ou vírgula) e que os dados para um sensor são livres de estar em qualquer linha. O sensor pode ter qualquer nome e tamanho (e.g. "Digikey MAX31855" ao invés de "Sensor A"). A palavra "leituras" é fixa. Para finalidades de dimensionamento de vetores, considere que um sensor pode registrar no máximo 100 valores.

O seu programa deve ler o arquivo texto e gravar os valores registrados pelos sensores (números inteiros) em um único arquivo binário contendo o número de leituras seguido dos valores para cada sensor. O nome do arquivo deve ser **data_nomeDaCidade.dat**. O **nomeDaCidade** é o nome lido do arquivo texto com _ no lugar dos brancos. Note que você deve montar esse nome de arquivo, concatenando "data_" com o nome da cidade. Escreva obrigatoriamente uma função auxiliar recursiva para trocar brancos por _. Nesta função recursiva use obrigatoriamente aritmética de ponteiros. No exemplo acima, o arquivo gravado é **data_Rio_de_Janeiro.dat**, que deve conter os seguintes números (onde o número de leituras está indicado em vermelho):

```
10 30 41 60 65 40 20 10 15 3 2 12 20 22 ...
```

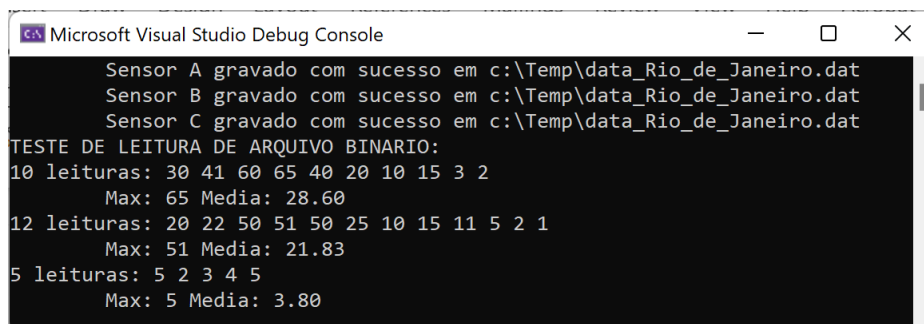
Escreva obrigatoriamente, e use sempre que possível, a função auxiliar `FILE* openFile(const char* file, const char* mode)` que recebe um nome de arquivo e o modo de abertura e abre o arquivo. Essa função imprime uma mensagem de erro seguida de uma interrupção do processamento se ocorrerem problemas na abertura do arquivo. No seu programa, use obrigatoriamente `while (!feof(...))` para ler o arquivo texto¹, mas não use-o para ler o arquivo binário².

Teste a gravação do arquivo da seguinte maneira: depois de fechar o arquivo binário, ainda na *main*, abra novamente o arquivo, leia os dados de cada sensor, imprima esses dados e calcule (e imprima) o maior valor e a média para cada sensor. Note que você não sabe quantos sensores há no arquivo².

Para calcular os valores pedidos, escreva uma função que retorna o maior valor e também disponibiliza (via argumento) a média.

Exiba mensagens de sucesso de escrita. Não faça alocações dinâmicas de memória. Lembre de fechar arquivos quando não precisar mais deles. Veja exemplo de saída na Fig. 1.

Nos testes, USE UM EXEMPLO TOTALMENTE DIFERENTE (o número de sensores e os valores na lista devem ser diferentes) do apresentado neste enunciado. Usar o mesmo exemplo deste enunciado anula a tarefa.



```
Microsoft Visual Studio Debug Console
Sensor A gravado com sucesso em c:\Temp\data_Rio_de_Janeiro.dat
Sensor B gravado com sucesso em c:\Temp\data_Rio_de_Janeiro.dat
Sensor C gravado com sucesso em c:\Temp\data_Rio_de_Janeiro.dat
TESTE DE LEITURA DE ARQUIVO BINARIO:
10 leituras: 30 41 60 65 40 20 10 15 3 2
Max: 65 Media: 28.60
12 leituras: 20 22 50 51 50 25 10 15 11 5 2 1
Max: 51 Media: 21.83
5 leituras: 5 2 3 4 5
Max: 5 Media: 3.80
```

Fig.1

Submeta um arquivo .c com seu programa e um arquivo .pdf contendo o conteúdo do arquivo texto de sua autoria usado nos seus testes e a imagem obtida com print screen da saída do seu programa.

Dica: use um vetor para ir guardando a leitura dos valores registrados pelos sensores.

¹ Somente quando uma operação de leitura encontra um "end of file" é que o **feof()** retorna verdade (i.e. retorna $\neq 0$). Então, o **while** ainda entra uma última vez para que **fscanf** encontre "end of file" e retorne **EOF** (usualmente -1).

² Para saber que você chegou ao fim do arquivo, compare o que retorna da função **fread** com o valor **n** a ser lido; e.g. quando **fread(..., ..., n, fin) == n** for falso, então você chegou ao fim de arquivo (ou ocorreu algum erro).