Projeto de Laboratório de Computadores 2019/20 - my_sort

O projeto a desenvolver em Laboratório de Computadores consiste no desenvolvimento de vários utilitários que poderão ser utilizados para o processamento, na linha de comandos, de dados guardados em ficheiros.

Nesta ficha de trabalho deverá implementar o comando my_sort.

Nota: este enunciado poderá ainda sofrer alterações.

Comando my_sort

Sinopse:

```
my_sort [OPÇÕES]... [FICHEIROS]
```

Descrição:

Implemente, em C o comando my_sort que leia o conteúdo de ficheiros e ordene esse contéudo (linhas do ficheiro) por uma ordem específica, escrevendo o resultado na saída padrão. Este comando deverá ordenar as linhas do ficheiro por ordem lexicográfica, usando determinadas opções. O comando poderá também ser usado para ordenar o ficheiro numericamente. Se o(s) ficheiro(s) for(em) apenas um "-" (ou não for indicado nenhum nome), o comando my_cut deverá ler da entrada padrão.

Em particular, deverá implementar o comando my_sort **com processamento de opções** da linha de comandos, nomeadamente:

- -r comparar/orderar por ordem decrescente.
- -n comparar/ordenar de acordo com o valor numérico.

Valor de retorno:

O programa deverá retornar (função exit()) o valor **0**, caso termine normalmente, ou um valor **>0**, caso ocorra algum erro.

Situações de erro:

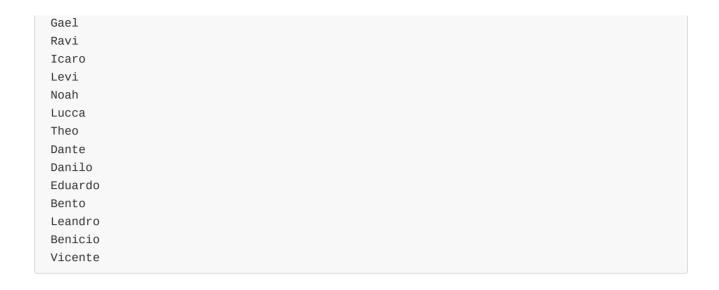
Caso um ficheiro não consiga ser aberto, deverá ser impressa, a mensagem de erro (no stderr):

./my_sort: <file>: No such file or directory

Exemplos de utilização

Considere o ficheiro file com o seguinte conteúdo:

Valentim



Exemplo 1:

o comando

```
$./my_sort file
```

escreverá na saída padrão:

```
Benicio
Bento
Danilo
Dante
Eduardo
Gael
Icaro
Leandro
Levi
Lucca
Noah
Ravi
Theo
Valentim
```

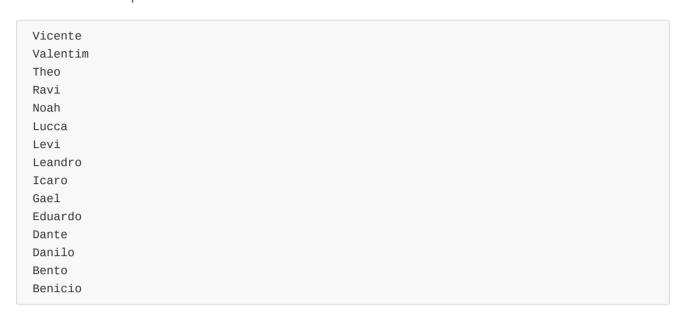
Exemplo 2:

A opção -r juntamente com o comando my_sort , ordena o conteúdo do ficheiro por ordem decrescente.

Considerando o ficheiro file acima, o comando

```
$./my_sort -r file
```

escreverá na saída padrão:



Exemplo 3:

A opção __n, juntamente com o comando my_sort, ordena por ordem crescente as linhas do ficheiro, que, para efeitos de comparação, deverão ser interpretadas como sendo números.

Considere o ficheiro *numeros* com o seguinte conteúdo:

```
34
23
15
674
12
53
1
56
4
3
```

o comando

```
$./my_sort -n numeros
```

escreverá na saída padrão:

```
1
3
4
12
15
23
34
53
56
674
```

Exemplo 4:

Considerando o ficheiro numeros acima, o comando

```
$./my_sort -nr numeros
```

escreverá na saída padrão:

```
674
56
53
34
23
15
12
4
3
1
```

Exemplo 5:

Considere o ficheiro numeros2 com o seguinte conteúdo:

```
23.1

12.2

4

125.14

2.34

7.1

25.8

0.23

19
```

o comando

```
$./my_sort -n numeros2
```

escreverá na saída padrão:

```
0.23
2.34
4
6
7.1
12.2
19
23.1
25.8
125.14
```

Exemplo 6:

Considerando o ficheiro file3 (redirecionado para a entrada padrão), o comando

```
$./my_sort < file3</pre>
```

escreverá na saída padrão:

```
A, 4, u
B, 10, a
C, 3, e
D, 2, o
E, 6, i
F, 5, a
```

Exemplo 7:

Considerando os ficheiros numeros, numeros2 e file2, definidos anteriormente, o comando

```
$./my_sort numeros - numeros2 < file2</pre>
```

escreverá na saída padrão:

```
0.23
1
12
12.2
125.14
15
19
2.34
23
23.1
25.8
3
34
4
4
53
56
```

```
6
674
7.1
A 4
B 10
C 3
D 2
E 6
F 5
```

Sugestões:

- Leia o manual do comando sort num terminal (\$ man sort).
- Faça experiências com o comando sort num terminal. O seu programa deverá reproduzir o comportamento do comando sort.
- Leia o manual das funções que considerar utilizar.
- Para o processamento de opções, poderá recorrer à função getopt() das bibliotecas *unistd.h* e *getopt.h*. Poderá encontrar exemplos de utilização desta função nas ligações: <u>link1</u>, <u>link2</u>, e <u>link3</u>.