Projeto de Laboratório de Computadores 2019/20 - my_cat

Projeto de Laboratório de Computadores 2019/20 - my_cat

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_cat`.

Nota: este enunciado poderá ainda sofrer alterações.

Comando my_cat

1. Comando `my_cat`

   Sinopse:

   `my_cat [OPÇÕES]... [FICHEIROS]...`

   Descrição:

   Implemente, em C o comando `my_cat` que leia o conteúdo de ficheiros de forma sequencial, escrevendo esses conteúdos na *saída padrão*. Os ficheiros deverão ser processados pela ordem que aparecem na linha de comandos. Se o ficheiro for apenas um `-` (ou não for indicado nenhum nome), o programa `my_cat` deverá ler da entrada padrão.

   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_cat: <file>: No such file or directory`

   Exemplos:

   O comando

   ```
   $ ./my_cat file1
   ```

   deverá imprimir o conteúdo do ficheiro *file1* para a saída padrão.

   Os comandos

   ```
   $ ./my_cat
   $ ./my_cat -
   ```

copiam a entrada padrão para a saída padrão, até ser lido o caracter `EOF` (`^D').
O comando

```
$ ./my_cat file1 file2 file3
```

deverá imprimir sequencialmente o conteúdo dos ficheiros *file1*, *file2* e *file3* para a saída padrão.
O comando

```
$ ./my_cat file1 - file2 - file3
```

imprime sequencialmente:

- o conteúdo do ficheiro *file1*;
- o que for lido da *entrada padrão* (até ser lido o caracter `EOF`);
- o conteúdo do ficheiro *file2*;
- o que for lido da *entrada padrão*;
- e, finalmente, o conteúdo do ficheiro *file3*.

O comando

```
$ ./my_cat file1 file2 > file3
```

deverá imprimir sequencialmente o conteúdo dos ficheiros *file1*, *file2* no ficheiro *file3*. Repare nesta instrução de shell, o comando não terá que abrir explicitamente o ficheiro *file3* (porquê?)
O comando

```
$ ./my_cat < file1 > file2
```

deverá imprimir sequencialmente o conteúdo do ficheiro *file1* no ficheiro *file2*. Repare nesta instrução de shell, o comando não terá que abrir explicitamente os ficheiro *file1* e *file2* (porquê?)
Caso o ficheiro file1 não consiga ser aberto, o comando

```
$ ./my_cat  file1 file2
```

deverá imprimir a mensagem de erro (no *stderr*):

```
./my_cat: file1: No such file or directory
```

e depois, imprimir o conteúdo do ficheiro *file2*.

2. Altere o programa `my_cat` para suportar as seguintes opções:

- `-n` o output deverá numerar as linhas do output começando pelo número 1. O formato do número que precede cada linha deverá ser `"%6d\t"`.

- `-b` o output deverá numerar apenas as linhas que não sejam vazias **(i.e., que não contenham apenas um caracter mudança de linha)**, começando pelo número 1. O formato do número que precede cada linha deverá ser `"%6d\t"`.
- `-s` eliminar linhas adjacentes vazias **(i.e., que contenham apenas um caracter mudança de linha)**, fazendo com que o output tenha espaçamento simples.

Sugestões

- Leia o manual do comando `cat` num terminal (`$ man cat`).
- Leia o manual das funções que considerar utilizar.
- Faça experiências com o comando `cat` num terminal. O seu programa deverá reproduzir o comportamento do comando `cat`.
    - 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: link1, link2, e link3.
- Outras funções deverá ponderar utilizar: `getc()`, `fgets()`, `putchar()` ou `fputs()`, `fprintf()`, `exit()` .
- Outras funções deverá ponderar utilizar: `getc()`, `fgets()`, `fputc()`, `putchar()`, `fputs()`, `fprintf()`, `exit()`.