Ficha Prática 1

Utilização do Sistema Operativo e Redireccionamento

Revisão. Redireccionamento pipes. Exemplificação com o comando cat.
Exercício de Programação - Criação dum comando tipo "cat" usando o standard library (stdio.h)
Antes Ver : http://www.di.ubi.pt/~operativos/praticos/pdf/2-utilizacao.pdf

1 Revisão: Redireccionamento https://www.di.ubi.pt/~operativos/unixtut/unix4.html

a. Descreva o significado dos símbolos
   - < _____
   - > _____
   - >> _____
   - 2> _____

b. Investigue o programa **cat**
   - Sinatxe e Utilização: Pelo linha de comando : man cat ou info cat
   - Pelo Internet http://www.computerhope.com/unix/ucat.htm
   - Experimente : cat ↵ cat /etc/passwd ↵ cat < /etc/passwd ↵ cat -n /etc/passwd ↵

c. Quantos parâmetros são passados para o programa **`cat`** nos seguintes casos

- **`cat`** `1 - Um Programa recebe sempre o seu nome próprio !`
- **`cat -t /etc/passwd`** _____
- **`cat /etc/passwd`** _____
- **`cat -t < /etc/passwd`** _____

2 Pipe revisão

- Explique o funcionamento e diferença entre as duas linhas com comandos em baixo

  $ curl https://www.rtp.pt/noticias/rss/desporto > tmp ; grep benfica tmp ; rm tmp
  $ curl https://www.rtp.pt/noticias/rss/desporto | grep benfica

- O curl relate informação, estatísticas sobre o acesso a rede, pelo canal de stderr que neste caso não tendo relevância podemos deitar foraQual destes comandos envie este informação para o ficheiro /dev/null
  1. $ curl https://www.rtp.pt/noticias/rss/desporto | grep benfica 2> /dev/null
  2. $ curl https://www.rtp.pt/noticias/rss/desporto 2> /dev/null | grep benfica
  3. $ curl https://www.rtp.pt/noticias/rss/desporto | grep benfica > /dev/null
  4. $ curl https://www.rtp.pt/noticias/rss/desporto > /dev/null | grep benfica

3 Programação em C

Escrever um comando chamado **`mostrar`** cuja funcionalidade seja igual ao comando **`cat`**.
De seguida apresenta-se o ficheiro **`mostrar.c`** para servir de base ao seu desenvolvimento.

`mostrar.c`

```
#include <stdio.h>

void streamCopy ( FILE * entrada, FILE * saida);

int main( int argc, char *argv[] ){

   if ( 1 == argc )
        streamCopy ( stdin,  stdout);
   //else abrir ficheiros – exercício
   return 0;
}

void streamCopy ( FILE * entrada,  FILE * saída ){
    int ch;
    while ( (ch=fgetc(entrada)) != EOF ){
       fputc( ch, saida);
    }
}
```

a. Escrever e compilar o programa **`mostrar.c`** e efetuar os seguintes testes

frutas.txt
laranjas
limoes

```
T1: mostrar
T2: mostrar < frutas.txt
T3: mostrar frutas.txt
(comparar com cat, cat </etc/passwd e cat /etc/passwd)
```

Terminada ?: O seu Programa efectua corretamente os testes ? Repare que o T3 não irá mostrar o ficheiro /etc/passwd portanto o teste passe se nada acontece ☺

b. Inserir código no **`mostrar.c`** de modo a que a aplicação daí resultante permita mostrar o conteúdo de ficheiros passados para a aplicação como parâmetros.

- Evitar erros fatais no programa, verifique fazendo **testes**. Efetuar os testes T1,T2 e T3 e adicionalmente

```
T4: mostrar frutas.txt mostrar.c
T5: mostrar frutas.txt  naoexiste
```

Para abrir um ficheiro utilize a função *fopen* do <stdio.h> . Faça "**man 3 fopen**" para ver o seu sintaxe
Quando for detectado um ficheiro que não existe o programa deverá imprimir no canal de erros padrão (stderr) a mensagem fprintf(**stderr**,"%s : Fatal Error" onde %s é o nome do ficheiro e depois chamar a função **perror(NULL)**

c. Implementa a opção -T. Esta opção faz que caracteres de controlo (\t) são representados pelo simbolo ^I

```
T6:  mostrar -T
T7:  mostrar -T frutas.txt mostrar.c
```

d. Implementa a opção -n. Esta opção faz que no inicio da cada linha é imprimido o numero da linha do input

```
T8:  mostrar -n
T9:  mostrar -n frutas.txt mostrar.c
```

Notas:

Sintaxe: mostrar [-T] [-n] ficheiros
• FILE *stdin, *stdout,*stderr.
o Estão declaradas no ficheiro <stdio.h> que habitualmente está na pasta /usr/include
o Estes ficheiros estarão abertas automaticamente no inicio da execução do programa.
o A linkagem com a biblioteca standard (libc.xx) – a versão estatica libc.a ou dinamica libc.so é feito automáticamente pelo programa de controlo do processo de compilação (p.ex gcc).

Notas/Sugestões

Tratamento das Opçoes. Sugestão

Sintaxe : mostrar [-n] [-T] [files]

Utilizar "Global Variable(s)" para indicar se uma opção estiver ligada ou desligada.

```
#define TRUE 1
#define FALSE 0
void streamCopy(FILE *entrada, FILE *saida);
int opcaoLinhas = FALSE;
int opcaoTabs = FALSE;
int main(int argc, char *argv[]) {
  //consumir arguments e adustar pointer
  //esta maniera não tem de alterar o codigo seguinte
  while (argc > 1 && '-' == argv[1][0] )
  {
    switch ( argv[1][1] ){
      case 'n' : opcaoLinhas = TRUE; break;
      case 'T' : opcaoTabs = TRUE; break;
    }
    argc--;
    argv++;
  }

  if (1 == argc)
    streamCopy(stdin, stdout);
```

```
//resto do programa mantêm-se
```

Other Possible Options : Show Line Endings with $, Remove empty lines , Remove spaces etc

Tratamento da nova linha na função de streamcopy

```
int nlinha=1;              //numero da linha atual
int inicioDaLinha=TRUE;
while ((ch = fgetc (entrada)) != EOF)
  {
    if ( opcaoLinhas && inicioDaLinha ) {
        fprintf(saida,"%6d\t", nlinha++);
        inicioDaLinha = FALSE;
    }
     fputc (ch, saida);
     if (ch=='\n') inicioDaLinha =TRUE;
  }
```

Notas:

↵ O simbolo para carrger no "enter"

Fim do Ficheiro EOF (End Of File): CTR+d (unix) CTR+z (windows) :