Sistemas Operativos **Ano lectivo 2025/2026**

Aula Prática Nº 12
Pipes

Objectivo

Estudo da comunicação entre processos usando *pipes*.

Guião

1. *Unnamed pipes*
   a. Consulte o manual da função `pipe()`. Analise o código do ficheiro `pipe1.c` e tente prever o que vai originar a sua execução.
   b. Compile o programa `pipe1.c` e verifique o resultado da sua execução.
   c. Altere o programa `pipe1.c` de modo a que o processo pai responda ao processo filho com "Good morning!"
   d. Consulte o manual da função `dup2()`. Analise o código do ficheiro `pipe2.c`. e tente prever o *output* do processo pai e do processo filho.
   e. Compile o programa `pipe2.c` e verifique o resultado da sua execução. Como foi parar o `printf()` do processo filho ao *output* do processo pai?
   f. Altere no programa anterior o `printf(...)` para um `fprintf(stderr,...)`. Altere também o comportamento do *pipe* para que a mensagem seja lida pelo processo pai.
   g. Analise o código do ficheiro `pipe3.c`. e tente prever o *output* do processo pai e do processo filho. Compile o programa `pipe3.c` e verifique o resultado da sua execução.
   h. Altere o ficheiro `pipe3.c` de modo a que o processo filho execute um `exec` do comando `ls`. Compile o programa alterado e verifique o resultado da sua execução.
   i. Analise o código do programa `pipe5.c` e tente prever o que vai originar a sua execução.
   j. Compile o programa `pipe5.c`. e verifique o resultado da sua execução.
   k. Altere o ficheiro `pipe5.c` de modo a que o programa execute um procedimento semelhante ao executado pela *bash* para o comando `cat passwd | grep sop`, sendo que o `cat` será executado no processo pai e o `grep` no processo filho.

2. *Popen*
   a. Consulte o manual da função `popen()`. Analise o código do ficheiro `popen1.c` e tente prever o que vai originar a sua execução.
   b. Altere o programa `popen1.c` de modo que o programa principal use a função `popen()` para comprimir tudo o que escrever no ficheiro associado, usando o comando `gzip`.

3. *Named pipes*
    a. Consulte o manual da função `mkfifo()`. Analise o código do ficheiro `namedpipe1.c` e tente prever o que vai originar a sua execução.
    b. Compile o programa `namedpipe1.c`. e verifique o resultado da sua execução.
    c. Analise o código do ficheiro `writepipe1.c` e tente prever o que vai originar a sua execução.
    d. Compile o programa `writepipe1.c` e execute-o. Porque não aparece o ">"?
    e. Ainda com o programa `writepipe1` a executar, abra um novo terminal e execute o comando `cat < myfifo1` no novo terminal. Repare que agora já apareceu o ">" no terminal do `writepipe1`. Escreva diversas mensagens no primeiro terminal e verifique o que acontece no segundo.
    f. Escreva o programa `readpipe1.c` que lê do *named pipe* `myfifo1` e escreve o que leu na consola.
    g. Coloque os comandos `writepipe1` e `readpipe1` a comunicarem entre si através do *pipe* `myfifo1`.