Relatório do Trabalho Prático de POO

José Costa (A82136) Luís Alves (A80165)

Miguel Carvalho (A81909)

Grupo 52

2 de Junho de 2018

Conteúdo

1 Introdução

Este projeto foi desenvolvido no âmbito da Unidade Curricular de *Sistemas Operativos*, sendo que foi proposto como trabalho prático, a criação de um processador de notebooks de comandos.

2 Sobre o programa

2.1 Parsing

Ao realizarmos parsing de um notebook, recolhemos a seguinte informação útil sobre cada comando:

- Comando com os respetivos argumentos
- Index (n) do comando no notebook

- Index do comando cujo output é preciso como input para executar este comando
- Nº de comandos que necessitam do output deste comando
- Array com os indexes dos comandos que necessitam do output deste comando

No entanto, guardamos o comando independentemente dos argumentos, que apenas são parsed devidamente aquando da execução do comando (através da função `get_args()`).

2.2 Execução de Comandos

Após realizarmos o parsing dos comandos e termos para cada um as informações supra mencionadas, corremos um ciclo de execução para cada um, podendo eles correr concorrentemente.

Criamos então um filho, que, caso não requeira o output de nenhum comando, executa regularmente tendo o output redirecionado para um pipe de output simples. Caso requeira o output de outro comando, duplica um pipe para o input do comando a executar.

Após a execução, o pai lê o output do filho e coloca esse output noutro conjunto de pipes, um para cada comando que vá precisar deste output como input, e um outro que servirá para colocar este output no notebook final. O output original (que estava num pipe), é logo libertado.

2.3 Pipes e Pastas

Para guardarmos os resultados intermédios (ver acima), necessitamos de guardar tanto os pipes de input, como os de output. Para isso, criamos uma pasta na diretoria `/tmp/SO_r`, sendo r um nº gerado pseudo-aleatoriamente.

Os Pipes de Input têm o seguinte formato: Pipe_f_t, sendo f o index do comando que fornece o output para o comando de index t (que necessita do conteúdo do pipe como input).

Já os Pipes de output simples, têm o formato Pipe_t, sendo t o index do output do comando n. São esses pipes (um por comando) que contêm o conteúdo que irá depois ser dirigido para os pipes de escrita no notebook (1), e os pipes de input de outros comandos.

De forma a não alterarmos o ficheiro em caso de término precoce (usando CTRL+C), criamos apenas um ficheiro temporário final no fim do processamento de todos os comandos, sendo que apenas terminado esse processo, esse ficheiro é copiado para o ficheiro original.

2.4 Limitações conhecidas

Após vários testes, concluímos que o nosso programa tem algumas limitações, que não conseguimos remover a tempo da entrega:

- Outputs de tamanho superior ao tamanho predefinido de um pipe com nome
- Comandos que contenham pipes (comando1 | comando2)

3 Conclusão

Com a realização deste trabalho, foi-nos possível aplicar as técnicas de programação em Unix lecionadas nas aulas num projeto mais concreto e maior do que os sugeridos nos exercícios. Para além disso, contribuiu para uma maior preparação para o teste.

No entanto, a aparente simplicidade do projeto depressa se mostrou isso mesmo, apenas uma aparência, já que de forma a poder correr tudo conforme os requisitos e sem limitações, exigiria uma maior carga de trabalho que não pudemos disponibilizar.

4 Notebooks de teste

Em todos estes notebooks, o resultado obtido é igual ao resultado esperado ao correrrmos estes comandos no terminal. Por isso, omitem-se os resultados, que facilmente se obtêm correndo o programa com um notebook com esses conteúdos.

4.1 Notebook 1

```
Este comando lista os ficheiros:
$ ls
Agora podemos ordenar estes ficheiros:
$| sort
E escolher o primeiro:
$| head -1
```

4.2 Notebook 2

```
Este comando lista os ficheiros:
$ ls
Agora podemos ordenar estes ficheiros:
$1| sort
Vamos escolher o primeiro:
$2| head -1
Conta quantos bytes tem:
$| wc -c
```

4.3 Notebook 3

```
Este comando acha-se o maior:
$ echo "Sou o maior"
Este comando vai contrariá-lo:
$ echo "Nao, nao es"
```

4.4 Notebook 4

```
Este comando lista os ficheiros:
$ ls
E este escolhe o primeiro:
```

```
$| head -1
```