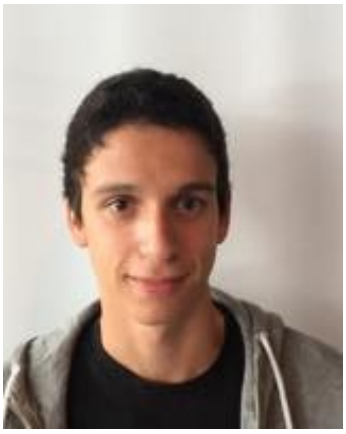
Projeto de Sistemas Operativos: Processamento de Notebooks

Grupo 23

- Ricardo Vieira - A81640
- João Imperadeiro - A80908
- José Gonçalo Costa - A82405



Introdução

Neste projeto, foi-nos proposto que desenvolvêssemos um programa que atua como um processador de notebooks, que deverá ser um comando que dado um nome de um ficheiro, interpreta-o, executa os comandos nele embebidos e acrescenta o resultado da execução dos comandos ao próprio ficheiro.

Os notebooks podem ser compostos por diversos tipos de linhas.

Por um lado, linhas de texto normais (que servem de documentação), as quais não são interpretadas pelo programa, sendo escritas sem alterações.

Por outro, comandos, dos quais se destacam três tipos:

- Comandos começados por “$”: “$ ls -l”
- Comandos começados por “$n|” (sendo n um inteiro positivo): “$3| cat | sort”
- Comandos começados por “$|” (equivalente a “$1|”): “$| head -1 > lixo.txt”

Todos estes comandos podem conter pipes, redirecionamento de input/output e um número arbitrário de argumentos.

O programa permite ainda o reprocessamento do notebook, interrupção da execução do programa e deteção de erros/comunicação dos mesmos ao utilizador.

Funcionamento

Parar entregar as funcionalidades propostas, o nosso programa lê uma linha do notebook de cada vez (através da nossa função readln) e interpreta-a, escrevendo o resultado para um ficheiro temporário que mais tarde é renomeado para o nome do ficheiro original (o notebook).

Ao interpretar as linhas, o programa pode-se deparar com dois casos: quando se trata de uma linha normal (de documentação), limita-se a copiar a linha para o ficheiro temporário; quando se trata de um dos outros três tipos de linhas referidos anteriormente, o programa escreve no ficheiro o output dos comandos respetivos, colocando-o a seguir à linha de comando entre “>>>” e “<<<”.

O programa trata as linhas de maneira diferente consoante o necessário. Se a linha começar por “$”, o programa executa o comando normalmente. Se a linha começar por “$n|”, o programa executa os comandos, enviando-lhes como input o output do n-ésimo comando anterior. No caso de a linha começar por “$|”, o programa interpreta-a, como referido anteriormente, como se se tratasse de uma linha começada por “$1|”.

Para qualquer um destes três casos, o programa aceita comandos com pipes (“|”), redirecionamentos de input/output (se fizer sentido) ou um número arbitrário de argumentos, e cria filhos para executar cada um dos comandos.

No caso dos pipelines, o programa cria pipes para os filhos comunicarem entre si (um pipe para cada par de filhos) e executa os filhos concorrentemente. No caso de redirecionamento de input/output, cabe ao filho interpretar se deve ou não fazer redirecionamento (através da análise dos seus argumentos) e trata ele próprio do redirecionamento antes de fazer exec. À semelhança da bash, apenas o primeiro comando recebe input do utilizador ou do notebook e apenas o último filho escreve o output no ficheiro temporário.

Para além disto, o programa permite:

- reprocessar um notebook, correndo todos comandos (pois podem ter sido alterados pelo utilizador) e ignorando os outputs que lá estiverem, ou seja, tudo o que estiver compreendido entre “>>>” e “<<<”.
- cancelar o processamento de um notebook através do envio de um sinal (Ctrl-C), sendo que o programa não altera o notebook original. Esta funcionalidade foi testada escrevendo no notebook o comando “$ cat” que fica à espera de input do utilizador e nos dá tempo para interromper a execução com Ctrl-C.
- detetar erros, parando o processamento quando estes ocorrem, quer seja no programa principal ou num dos filhos, e comunicando ao utilizador o erro ocorrido e a respetiva linha. Também aqui o notebook não é alterado.

Exemplo de notebook

Notebook antes do processamento:

Este comando lista os ficheiros:
$ ls
Agora podemos ordenar estes ficheiros:
$| sort
E escolher o primeiro:
$| head -1
Enviamos o output do primeiro comando para o ficheiro lixo.txt:
$3| cat > lixo.txt
E ordenamos o output e contamos o número de linhas:
$ sort --reverse < lixo.txt | wc -l
Conta o numero de ficheiros nesta pasta depois da criação do lixo.txt:
$ ls | wc -l
Escreve no notebook o que for introduzido na bash pelo utilizador:
$ cat

Notebook depois do processamento:

Este comando lista os ficheiros:
$ ls
>>>
enunciado-so-2017-18.pdf
exemplo.nb
Makefile
notebook
processadorNB.c
processadorNB.o
README.md
temp.txt
<<<
Agora podemos ordenar estes ficheiros:
$| sort
>>>
enunciado-so-2017-18.pdf
exemplo.nb
Makefile
notebook
processadorNB.c
processadorNB.o
README.md
temp.txt
<<<
E escolher o primeiro:
$| head -1
>>>
enunciado-so-2017-18.pdf
<<<
Enviamos o output do primeiro comando para o ficheiro lixo.txt:
$3| cat > lixo.txt
>>>
<<<
E ordenamos o output e contamos o número de linhas:
$ sort --reverse < lixo.txt | wc -l
>>>
8
<<<
Conta o numero de ficheiros nesta pasta depois da criação do lixo.txt:
$ ls | wc -l
>>>
9
<<<
Escreve no notebook o que for introduzido na bash pelo utilizador:
$ cat
>>>
vou passar a SO
vou passar a SO
vou passar a SO
<<<