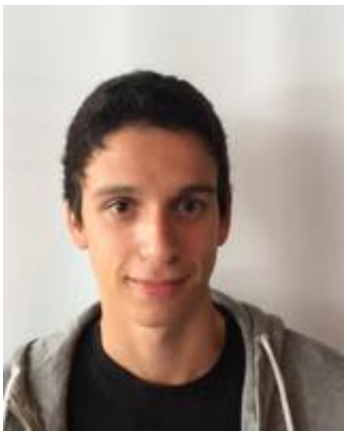
Projeto Sistemas Operativos: Processamento de Notebooks

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. Este 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.

O programa aceita três tipos de comandos:

- Uma linha que começa com “$”: “$ ls -l”
- Uma linha que começa com “$n|” sendo n um inteiro positivo: “$3| cat | sort”
- Uma linha que começa com “$|” equivalente a “$1|”: “$| head -1 > lixo”

Para além disto, o notebook aceita linhas de texto, as quais não são interpretadas pelo programa.

As linhas de comandos aceitam pipe, redirecionamento de input/output e número arbitrário de argumentos.

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

Funcionamento

O programa lê uma linha do ficheiro de cada vez (readln) e interpreta-as uma a uma escrevendo o resultado para um ficheiro temporário que mais tarde é renomeado para o nome do ficheiro original (o notebook). Há dois casos: quando se trata de uma linha normal 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, como referido anteriormente, como uma linha começada por “$1|”.

Para qualquer um destes casos, o programa aceita linhas com pipes (“|”) ou redirecionamentos de input/output (se fizer sentido) e cria filhos para tratar cada um dos comandos. Para isso, o programa, no caso dos pipelines, cria pipes para os filhos comunicarem entre si executa os filhos concorrentemente. No caso de redirecionamento de input/output o filho interpreta se deve ou não fazer redirecionamento e trata ele próprio do redirecionamento. À semelhança da bash, apenas o primeiro recebe input do utlizador ou do notebook e apenas o último filho escreve o output no ficheiro temporário.

Para além disto, o programa permite:

- reprocessamento do notebook, correndo os comandos que podem ter sido alterados pelo utilizador, e ignorando os outputs que lá estiverem, ou seja, tudo o que estiver compreendido entre “>>>” e “<<<”.
- cancelamento do processamento, através do envio de um sinal quando o utilizador prime “Ctrl-C”, não alterando 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.
- deteção de erros, parando o processamento quando estes ocorrem, quer seja no programa principal ou num dos filhos, e comunicando ao utilizador o erro ocurrido e a respetiva linha. Também aqui o notebook não altera.

Exemplo de notebook

Notebook antes de 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 após 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
<<<