universidade de aveiro
theoria poiesis praxis

Gestão de armazenamento: Monitorização do espaço ocupado

Sistemas Operativos

Prof. Carlos Borges

Tiago Albuquerque, 112901

Guilherme Mesquita, 112957

Índice

Introdução

No âmbito da unidade curricular de Sistemas Operativos, este trabalho tem como objetivo o desenvolvimento de scripts em bash que permitem monitorizar o espaço ocupado em disco, e a sua variação ao longo do tempo, por ficheiros com determinadas propriedades, facilitando assim a gestão do armazenamento. Conseguiremos assim visualizar o total do espaço ocupado em disco (em bytes) por todos os ficheiros selecionados em todas as diretorias passadas como parâmetro e nas suas descendentes.

Ao longo deste relatório será apresentada a metodologia utilizada na resolução do problema proposto, assim como todos os testes realizados para verificarmos a validade da solução.

Serão ainda apresentadas as diversas etapas de resolução do problema juntamente com uma breve exposição do raciocínio adotado. Surgirão ainda algumas imagens do código fonte e diagramas de modo a facilitar a compreensão de toda a solução.

Abordagem usada para resolver o problema

Com o propósito de simplificar a resolução do problema e diminuir a sua suscetibilidade a erros, necessitamos de dividir a resolução em diferentes etapas, cada uma com o seu objetivo definido.

Etapas de resolução adotadas:

1. Análise e verificação de todos os possíveis argumentos passados pelo utilizador;
2. Extração e processamento da informação;
3. Exposição dos resultados obtidos.

A partir deste ponto, será então dado enfâse a cada uma destas etapas do processo de resolução do problema.

Análise e verificação de argumentos

Script spacecheck.sh

O resultado dos scripts em bash pode variar consoante as diversas combinações entre as opções passadas como argumentos pelo utilizador. Deste modo, existe a necessidade de estas serem devidamente validadas.

Com o objetivo de impedir que o utilizador digite uma opção diferente das pretendidas (-n; -d; -s; -l; -r; -a), criámos uma função **erro** (figura 1) que termina automaticamente a execução do programa retornando uma mensagem alusiva ao acontecimento (“ERRO! Opção inválida. Tente novamente!”).

```
4   function erro {
5     echo "ERRO! Opção inválida. Tente novamente!"
6     exit 1
7   }
```

Figura 1

Procedemos também à criação de variáveis que representam as diferentes opções possíveis (figura 2), inicializando-as com um valor default (valor utilizado quando o utilizador não passa uma determinada opção nos argumentos), assim como uma simples variável que contém todos os argumentos que passamos na linha de comandos (**all_args**).

```
regex=".*"  #A expressão regular começa vazia (abrange todos os ficheiros)
data="0"  #-d
size_min="0" #Tamanho minimo estipulado como default
sort_option="-k1,1nr"  #Inicialmente, a ordenaçao e feita do maior para o menor (em termos de armazenamento)
limit_lines="0" #Inicializar com zero

all_args="$@"
```

Figura 2

Iniciando uma análise mais pormenorizada de modo a determinar quais os argumentos passados pelo utilizador, utilizamos o comando **getops**, juntamente com uma estrutura **switch-case** (figura 3). Sempre que o utilizador deseja ordenar a tabela por ordem alfabética (neste caso usando a opção **-a**), por exemplo, existem variáveis (neste caso a **sort_option**) que vão guardar opções de comandos de modo a serem utilizadas futuramente em pipes/pipelines.

```
reverse_sort="false"  #Variavel que criei para controlar o reverse (-r)
while getopts ":n:d:s:l:ra" option; do
  case "$option" in
    n)
      regex="$OPTARG" ;;
    d) #Verificar se o formato da data que passamos como argumento tem o formato correto (Ex: "Sep 10 10:00")
      if [[ "$OPTARG" =~ ^[A-Z][a-z]{2}\ [0-9]{1,2}\ [0-9]{2}:[0-9]{2}$ ]]; then
        data="$OPTARG"
      else
        echo "ERRO: Formato incorreto da data. Exemplo: 'Sep 10 10:00'."
        exit 1
      fi
      ;;
    s)
      size_min="$OPTARG" ;;
    r)
      sort_option="-k1,1n"
      reverse_sort="true" ;;
    a)
      sort_option="-k2,2" ;;
    l)
      if [[ "$OPTARG" =~ ^[1-9][0-9]*$ ]]; then
        limit_lines="$OPTARG"
      else
        echo "ERRO: O argumento da opção -l deve ser um número inteiro positivo."
        exit 1
      fi
      ;;
    ?)
      erro ;;
  esac
done
```

Figura 3

De salientar ainda que elaborámos duas verificações (para as opções **-d** e **-l**). No caso da seleção de ficheiros através da indicação do tamanho mínimo do ficheiro (opção **-d**) temos uma verificação (figura 4) encarregue de analisar se a variável **date** foi passada conforme pedida (por exemplo, “Sep 10 10:00”). Caso a data não tenha sido passada corretamente nos argumentos irá ser executada uma mensagem de erro (“ERRO: Formato incorreto da data.”) juntamente com o término forçado da execução do programa.

```
33      d)
34        #Verificar se o formato da data que passamos como argumento tem o formato correto (Ex: "Sep 10 10:00")
35        if date -d "$OPTARG" >/dev/null 2>&1; then
36          data="$OPTARG"
37        else
38          echo "ERRO: Formato incorreto da data."
39          exit 1
40        fi
41        ;;
```

Figura 4

Do mesmo modo, para a opção **-l** elaborámos uma verificação (figura 5) que consta em averiguar se o número de linhas passadas nos argumentos (logo a seguir à identificação da opção -l) corresponde a um número inteiro positivo (maior que zero). Caso contrário, mais uma vez, terminará a execução do script, assim como será mostrada a mensagem de erro apresentada.

```
l)
  if [[ "$OPTARG" =~ ^[1-9][0-9]*$ ]]; then
    limit_lines="$OPTARG"
  else
    echo "ERRO: O argumento da opção -l deve ser um número inteiro positivo."
    exit 1
  fi
  ;;
```

Figura 5

Script spacerate.sh

Antes de iniciar a análise do script **spacerate.sh**, temos de salientar que os ficheiros analisados e passados por argumentos para o script são ficheiros que contém o **output gerado** do script **spacecheck.sh**. Redirecionamos o output (do spacecheck.sh) para um ficheiro .*txt usando o seguinte codigo no terminal de comandos:
./spacecheck.sh /home/tiagoalb/Desktop/SO > output1.txt

No script spacerate.sh começamos por estabelecer, os requisitos básicos para que o programa funcione normalmente:

- Verificar se foram fornecidos dois ficheiros para comparar (figura 6):

```
if [ "$#" -ne 2 ]; then
    echo "Por favor, forneça dois arquivos para comparar."
    exit 1
fi
```

Figura 6

- Verificar se os ficheiros passados nos argumentos estão presentes no mesmo diretório que os dois scripts (spacecheck.sh e spacerate.sh) (figura 7):

```
  if [ ! -f "$file_new" ] || [ ! -f "$file_old" ]; then
      echo "Os ficheiros especificados não existem."
      exit 1
  fi
```

Figura 7

De seguida, criamos variáveis onde vamos armazenar os arquivos especificados como argumentos de modo a podermos, através de loops **for,** desenvolver o código para obtermos o resultado pretendido (Figura 8).

```
file_new="$1"
file_old="$2"
```

Figura 8

Extração e processamento de informação

Script spacecheck.sh

Após a extração e validação dos argumentos passados pelo utilizador, é necessário conhecer quais os diretórios que vão ser analisados ou que contém arquivos que correspondem às expressões regulares que passarmos como argumentos. Para tal recorremos à função **diretoriosPretendidos** que juntamente com duas expressões condicionais facilitam essa mesma análise. De realçar que o uso das expressões condicionais facilita o controlo da seleção dos ficheiros, considerando a presença ou não da opção **-d** (figura 9).

Nesta função recorremos ao comando **find**, para pesquisar arquivos dentro do diretório especificado - "**$1**", e ao comando **sort -u**, que recebe os resultados do comando find e garante que não existem diretórios repetidos (removendo linhas duplicadas).

```
function diretoriosPretendidos {
  if [ "$data" != "0" ]; then
    #Se a opçao -d foi fornecida, usamos o -not -newermt para filtrar consoante a data de modificaçao do arquivo, conforme e pedido
    find "$1" -type d -not -newermt "$data" 2>/dev/null | sort -u
  else
    #Se a opçao -d nao foi fornecida, nao a podemos considerar para nao perdermos diretórios
    find "$1" -type d 2>/dev/null | sort -u
  fi
}
```

Figura 9

Seguidamente, para determinar o espaço ocupado por um determinado ficheiro elaboramos a função **calcularEspacoArquivos** (figura 10).

```
function calcularEspacoArquivos {
  local counter=0

  diretoriosPretendidos "$1" | while read -r sub_dir; do
    total_size=0

    if [ ! -r "$sub_dir" ] || [ ! -x "$sub_dir" ]; then
      total_size="NA"
    else
      for file in "$sub_dir"/*; do
        if [ -f "$file" ] && [[ "$file" =~ $regex ]]; then    #Verificar se e um arquivo valido e se corresponde a expressao regular
          du_result=$(du -b "$file")
          file_size=$(echo "$du_result" | awk '{print $1}')
          if [ "$?" -ne 0 ]; then
            total_size="NA"
            break #Se houver algum erro, sair do loop, o resultado sera NA
          elif [ "$file_size" -ge "$size_min" ]; then
            total_size=$((total_size + file_size))
          fi
        fi
      done
    fi

    echo -e "$total_size\t$sub_dir"
    counter=$((counter+1))

    if [ "$limit_lines" -gt 0 ] && [ "$counter" -ge "$limit_lines" ]; then
      break  #Se o counter atingir o limite, sair do loop
    fi
  done
}
```

Figura 10

Nesta função iteramos sobre os diretórios obtidos da função anterior (**diretoriosPretendidos**) e com o auxílio de uma loop **for** e de uma expressão condicional verificamos se estamos a analisar um ficheiro válido e correspondente à expressão regular pretendida. Efetuamos o loop **for** com o objetivo de iterar sobre os ficheiros presentes nos subdiretórios. No fim teremos informações referentes aos ficheiros, como o tamanho total (soma dos tamanhos) de todos os ficheiros que correspondem à expressão regular que passamos com argumento (variável **total_size**). De notar que o loop **for** encontra-se dentro de uma expressão condicional (**else**) para facilitar o controlo dos ficheiros de que não possuímos permissões de acesso (figura 11).

```
if [ ! -r "$sub_dir" ] || [ ! -x "$sub_dir" ]; then
  total_size="NA"
else
  for file in "$sub_dir"/*; do
    if [ -f "$file" ] && [[ "$file" =~ $regex ]]; then    #Verificar se e um arquivo valido e se corresponde a expressao regular
      du_result=$(du -b "$file")
      file_size=$(echo "$du_result" | awk '{print $1}')
      if [ "$?" -ne 0 ]; then
        total_size="NA"
        break #Se houver algum erro, sair do loop, o resultado sera NA
      elif [ "$file_size" -ge "$size_min" ]; then
        total_size=$((total_size + file_size))
      fi
    fi
  done
fi
```

Figura 11

Ainda dentro da mesma função, temos mais uma expressão condicional que facilita o uso da opção **-l** (figura 12).

```
if [ "$limit_lines" -gt 0 ] && [ "$counter" -ge "$limit_lines" ]; then
  break  #Se o counter atingir o limite, sair do loop
fi
```

Figura 12

Script spacerate.sh

Para melhor processarmos toda a informação, elaboramos uma forma de ler o conteúdo dos arquivos especificados, usando uma combinação de comandos que vai ser explicado já a seguir. O comando **grep -v '^$' "$file…"** vai procurar as linhas que não estão vazias/em branco, no ficheiro que é passado. Este comando é executado de igual forma para o caso do segundo ficheiro. Após a análise do comando **grep**, temos a seguinte sucessão de comandos **tail -n +3** onde vai ser selecionada a partir de que linha é que vai começar a análise do output. Utilizamos o comando **mapfile** para ler as linhas dos arquivos e armazenar o conteúdo em **arrays** com um nome sugestivo (figura 13)**.**

```
mapfile -t file_new_array < <(grep -v '^$' "$file_new" | tail -n +3)
mapfile -t file_old_array < <(grep -v '^$' "$file_old" | tail -n +3)
```

Figura 13

Declaramos também três variáveis que representam **arrays**/dicionários, que será determinante para a comparação dos dois ficheiros (figura 14).

```
declare -A size_new_mapping
declare -A size_old_mapping
declare -A dirs
```

Figura 14

De seguida iniciamos dois loops **for** que iteram sobre cada linha dos arrays correspondentes a cada um dos ficheiros, e onde a variável **line** contém o valor da linha atual em cada iteração. Utilizamos o comando **awk** para extrair o tamanho e armazenar o valor na variável **line**. Construímos também o array/dicionário usando o nome do arquivo como key e o valor da variável size como o valor associado a cada key (figura 15).

```
for line in "${file_new_array[@]}"; do
    size=$(echo "$line" | awk '{print $1}')
    dirname=$(echo "$line" | cut -f2- -d$'\t')
    size_new_mapping["$dirname"]="$size"
done


for line in "${file_old_array[@]}"; do
    size=$(echo "$line" | awk '{print $1}')
    dirname=$(echo "$line" | cut -f2- -d$'\t')
    size_old_mapping["$dirname"]="$size"
done
```

Figura 15

Exposição dos resultados obtidos

Script spacecheck.sh

De modo a apresentar os resultados pedidos da forma pretendida, utilizamos um loop **for** (figura 16), onde se trabalha toda informação previamente obtida.

```
for dir in "$@"; do
  echo -e "\nEstamos no diretório -> $dir\n"
  echo -e "SIZE\tNAME\t$(date +%Y%m%d)    $all_args" #Cabeçalho

  #Para controlar quando usamos o -r -a -n; se usarmos o -r:"$reverse_sort" == "true" e se usarmos o -a:"$sort_option" == "-k2,2"
  if [ "$reverse_sort" == "true" ] && [ "$sort_option" == "-k2,2" ]; then
    #Se a opcao -r foi usada em conjunto com -a -n, reverta a ordem de classificacao
    calcularEspacoArquivos "$dir" | sort -t$'\t' $sort_option -r
  else  #Caso contrario, ou seja, quando nao usamos o -r -a -n
    calcularEspacoArquivos "$dir" | sort -t$'\t' $sort_option
  fi
done
```

Figura 16

O output será uma tabela, juntamente com um simples cabeçalho, onde estará toda a informação pedida, mostrando a análise que é feita aos subdiretórios do diretório definido nos argumentos.

Teremos diferentes possibilidades de ordenar/printar a tabela (como já foi brevemente falado anteriormente):

- Ordem inversa da ordem de leitura (por defeito os ficheiros aparecem por ordem decrescente de espaço ocupado) -> opçao **-r**;
- Ordem alfabética -> opçao **-a**;
- Limitar o número de linhas da tabela -> opçao **-l**

Em relação à forma como algumas informações (nomeadamente o campo SIZE) dos ficheiros devem aparecer, ou quais é que devem aparecer, temos, respetivamente, as opções **-s** (tratam o tamanho mínimo do ficheiro) ou **-d** (trata os ficheiros pela especificação da data máxima de modificação).

Script spacerate.sh

Para controlar que diretórios é que estão presentes nos dois ficheiros output, criamos um dicionário **dirs** (figura 17). Este dicionário tem como valor das keys todos os diretórios presentes nos outros dois dicionários (size_new_mapping e size_old_mapping), e valores associados a cada valor da key será definido com 0 (zero).

```
for dir in "${!size_new_mapping[@]}" "${!size_old_mapping[@]}"; do
    dirs["$dir"]=0
done
```

Figura 17

De modo a apresentar os resultados pedidos da forma pretendida, utilizamos um loop **for** (figura 18), onde se trabalha toda informação previamente obtida. Iniciamos o loop iterando sobre o valor das keys do array **dirs**. Depois são extraídos os valores necessários para efetuar a análise. Olhando agora para as expressões condicionais, o código verifica se as variáveis **size_old** e **size_new** estão vazias, ou seja, não têm nenhum valor associado, fazendo de seguida toda a análise pretendida.

```
for dir in "${!dirs[@]}"; do
    size_new="${size_new_mapping["$dir"]}"
    size_old="${size_old_mapping["$dir"]}"

    # Se não há correspondência no primeiro arquivo, então é uma adição
    if [ -z "$size_old" ]; then
        echo -e "$size_new\t$dir\tNEW"
    # Se não há correspondência no segundo arquivo, então é uma remoção
    elif [ -z "$size_new" ]; then
        echo -e "-$size_old\t$dir\tREMOVED"
    else
        # Calcular a diferença real de tamanhos
        size_diff=$((size_old - size_new))

        # Imprimir a diferença real de tamanho
        echo -e "$size_diff\t$dir"
    fi
done | (sort -k1,1nr) | ($alphabetical && sort -k2 || cat) | ($reverse && tac || cat)
```

Figura 18

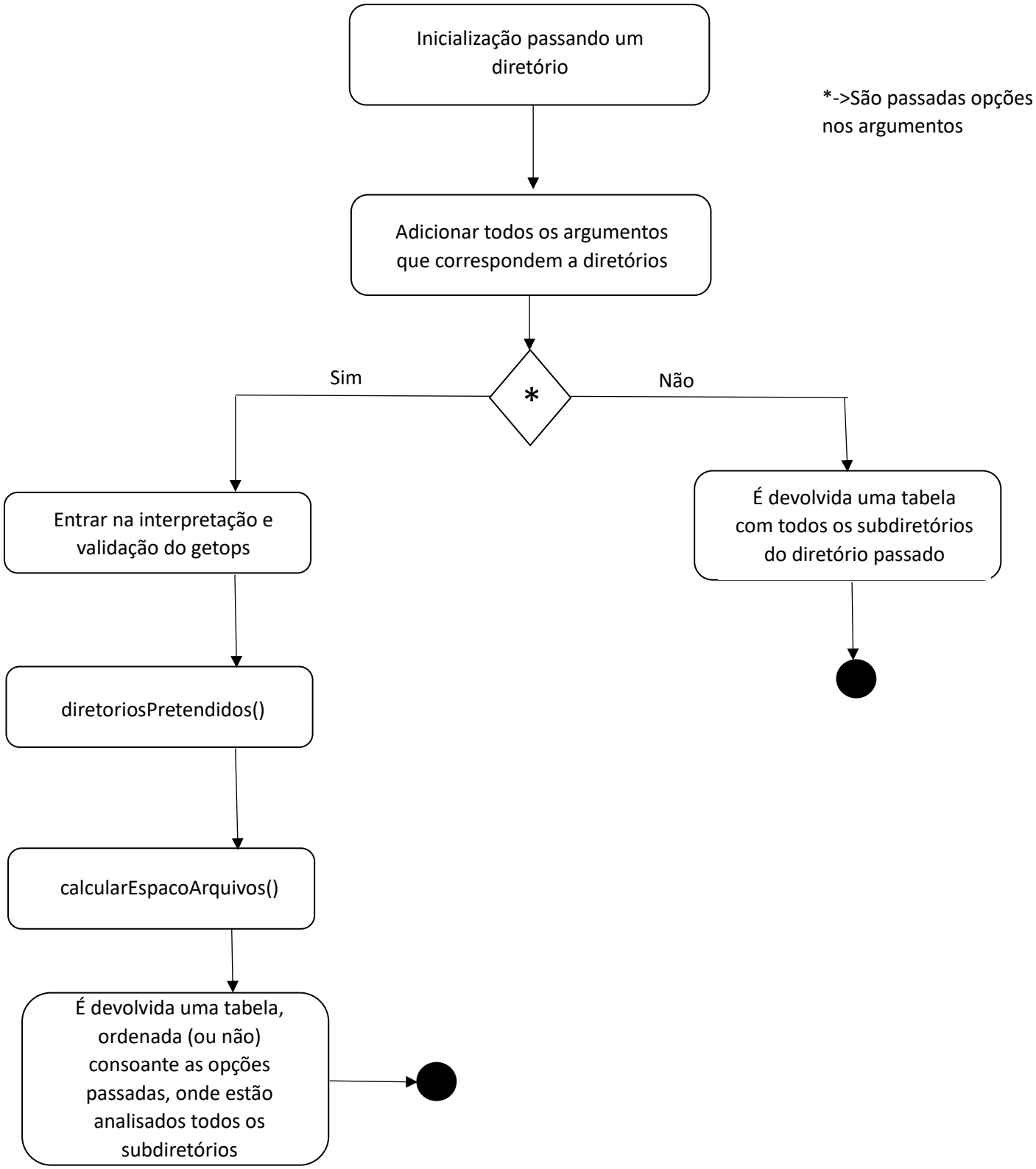
De notar que também podemos ordenar a tabela por ordem inversa, à ordem default, ou por ordem alfabética, usando as opções **-a** e **-r** (figura 19).

```
#Criar variaveis para definir as opcoes default
reverse=false
alphabetical=false

#Analise de todas opcoes que podemos passar na linha de comando (opcoes de ordenacao)
while getopts ":ra" option; do
  case "${option}" in
    r)
      reverse=true
      ;;
    a)
      alphabetical=true
      ;;
    \?)
      echo "Opção inválida: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "A opção -$OPTARG requer um argumento." >&2
      exit 1
      ;;
  esac
done
shift $((OPTIND-1))
```

Figura 19

Execução do script (spacecheck.sh)



Testes

Com o objetivo de verificar se o script funciona como esperado, fizemos diversos testes nos quais dividimos em duas categorias:

- Testes que desencadeiam erros;
- Testes que não desencadeiam erros.

Script spacecheck.sh

Testes que desencadeiam erros

- **Teste 1:** ./spacecheck.sh /home/tiagoalb/Desktop/SO/Projeto123

Quando o diretório passado como argumento não existe. **Nota:** Apesar de não ser mostrada nenhuma mensagem de erro a execução do código termina devido à inexistência do diretório passado.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1$ ./spacecheck.sh /home/tiagoalb/Desktop/SO/Projeto123
```

- **Teste 2:** ./spacecheck.sh -h /home/tiagoalb/Desktop/SO/Projeto123

Quando a opção passada como argumento não existe, ou seja, não esta presente no getops.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1$ ./spacecheck.sh -h /home/tiagoalb/Desktop/SO/Projeto123
ERRO! Opção inválida. Tente novamente!
```

- **Teste 3:** ./spacecheck.sh -d "2023/10/11" /home/tiagoalb/Desktop/SO/Projeto1

Quando, usando a opção -d, passamos uma data com um formato diferente da apresentada no guião.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1$ ./spacecheck.sh -d "2023/10/11" /home/tiagoalb/Desktop/SO/Projeto1
ERRO: Formato incorreto da data. Exemplo: 'Sep 10 10:00'.
```

- **Teste 4:** ./spacecheck.sh -l 1 /home/tiagoalb/Desktop/SO/Projeto1

Quando, usando a opção -l, passamos um número negativo ou igual a zero.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1$ ./spacecheck.sh -l -1 /home/tiagoalb/Desktop/SO/Projeto1
ERRO: O argumento da opção -l deve ser um número inteiro positivo.
```

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1$ ./spacecheck.sh -l 0 /home/tiagoalb/Desktop/SO/Projeto1
ERRO: O argumento da opção -l deve ser um número inteiro positivo.
```

Testes que não desencadeiam erros

- **Teste 1:** ./spacecheck.sh /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos o diretório assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
/home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    /home/tiagoalb/Desktop/test_a1
33128   /home/tiagoalb/Desktop/test_a1/aaa
26503   /home/tiagoalb/Desktop/test_a1
9544    /home/tiagoalb/Desktop/test_a1/rrr
7240    /home/tiagoalb/Desktop/test_a1/aaa/zzzz
```

- **Teste 2:** ./spacecheck.sh -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido; isto é, os diretórios que contenham ficheiros do tipo da expressão regular passada nos argumentos tem o campo SIZE, respetivo, preenchido com o valor da soma dos espaços ocupados por todos os ficheiros desse tipo. Por sua vez os diretórios que não tenham ficheiros desse tipo, tem o seu campo SIZE a 0 (zero).
De realçar, por fim, que a tabela aparece ordenada por ordem decrescente dos valores do SIZE.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -n .*sh /home/tiagoalb/Desktop/test_a1
18140   /home/tiagoalb/Desktop/test_a1
9534    /home/tiagoalb/Desktop/test_a1/rrr
0       /home/tiagoalb/Desktop/test_a1/aaa
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
```

- **Teste 3:** ./spacecheck.sh -r -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
De realçar, por fim, que a tabela aparece ordenada por ordem inversa (crescente) em relação à ordem de ordenação da opção -n.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-r -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -r -n .*sh /home/tiagoalb/Desktop/test_a1
0       /home/tiagoalb/Desktop/test_a1/aaa
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
9534    /home/tiagoalb/Desktop/test_a1/rrr
18140   /home/tiagoalb/Desktop/test_a1
```

- **Teste 4:** ./spacecheck.sh -a -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
De realçar, por fim, que a tabela aparece ordenada por ordem alfabética.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-a -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -a -n .*sh /home/tiagoalb/Desktop/test_a1
18140   /home/tiagoalb/Desktop/test_a1
0       /home/tiagoalb/Desktop/test_a1/aaa
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
9534    /home/tiagoalb/Desktop/test_a1/rrr
```

- **Teste 5:** ./spacecheck.sh -r -a -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
De realçar, por fim, que a tabela aparece ordenada por ordem inversa à ordem apresentada na figura anterior.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-r -a -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -r -a -n .*sh /home/tiagoalb/Desktop/test_a1
9534    /home/tiagoalb/Desktop/test_a1/rrr
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
0       /home/tiagoalb/Desktop/test_a1/aaa
18140   /home/tiagoalb/Desktop/test_a1
```

- **Teste 6:** ./spacecheck.sh -n ".*sh" -l 2 /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
A opção -l vai delimitar quantas linhas é que a tabela vai possuir (neste caso duas).

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-n ".*sh" -l 2 /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -n .*sh -l 2 /home/tiagoalb/Desktop/test_a1
18140   /home/tiagoalb/Desktop/test_a1
0       /home/tiagoalb/Desktop/test_a1/aaa
```

- **Teste 7:**
  ./spacecheck.sh -d "Nov 9 10:00" -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
A opção -d vai selecionar que diretórios é que irão aparecer na tabela de acordo com a data máxima de modificação dos ficheiros.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-d "Nov 9 10:00" -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -d Nov 9 10:00 -n .*sh /home/tiagoalb/Desktop/test_
1
9534    /home/tiagoalb/Desktop/test_a1/rrr
0       /home/tiagoalb/Desktop/test_a1/aaa
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
```

- **Teste 8:** ./spacecheck.sh -s 9000 -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1

Deve fornecer uma tabela com todos os diretórios assim como todos os subdiretórios juntamente com o campo SIZE devidamente preenchido.
A opção -s, vai selecionar os ficheiros através da indicação do tamanho mínimo do ficheiro. Os ficheiros que tenham menos que 9000 bytes (neste caso) aparecem com o seu campo SIZE a 0 (zero).

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-s 9000 -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -s 9000 -n .*sh /home/tiagoalb/Desktop/test_a1
9534    /home/tiagoalb/Desktop/test_a1/rrr
0       /home/tiagoalb/Desktop/test_a1
0       /home/tiagoalb/Desktop/test_a1/aaa
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
```

- **Teste 9:**

./spacecheck.sh -r -a -d "Nov 9 10:00" -s 9000 -n ".*sh" /home/tiagoalb/Desktop/SO/Projeto1
De realçar que todas as combinações são possíveis (um exemplo).

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacecheck.sh
-r -a -d "Nov 9 10:00" -s 9000 -n ".*sh" /home/tiagoalb/Desktop/test_a1

Estamos no diretório -> /home/tiagoalb/Desktop/test_a1

SIZE    NAME    20231113    -r -a -d Nov 9 10:00 -s 9000 -n .*sh /home/tiagoalb/
Desktop/test_a1
9534    /home/tiagoalb/Desktop/test_a1/rrr
0       /home/tiagoalb/Desktop/test_a1/aaa/zzzz
0       /home/tiagoalb/Desktop/test_a1/aaa
```

Script spacerate.sh

Testes que desencadeiam erros

- **Teste 1:** ./spacerate.sh

Caso não passemos nenhum ficheiro output o script não será executado.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacerate.sh
Por favor, forneça dois ficheiros para comparar.
```

- **Teste 2:** ./spacerate.sh ficheiro.txt

Caso só passemos um ficheiro output o script vai retornar um erro, visto que, são precisos dois ficheiros para conseguirmos executar o script.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacerate.sh f
icheiro.txt
Por favor, forneça dois ficheiros para comparar.
```

- **Teste 3:** ./spacerate.sh file1.txt file2.txt

Caso passemos dois ficheiros output que não existam no diretório em análise (no mesmo diretório do script) é retornado um erro e a execução termina.

```
tiagoalb@tiagoalb:~/Desktop/SO/RepositorioProjeto1/SO_Projeto1$ ./spacerate.sh f
icheiro.txt ficheiro2.txt
Os ficheiros especificados não existem.
```

Testes que não desencadeiam erros

- **Teste 1:** ./spacerate.sh output1.txt output2.txt

A análise é efetuada conforme pretendida.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1/Work$ ./spacerate.sh output1.txt output2
.txt
792718  /home/tiagoalb/Desktop/SO/Aulas P/Aula02        NEW
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula03
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula05
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula07
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula08
-365563 /home/tiagoalb/Desktop/SO/Aulas P/Aula01
-837326 /home/tiagoalb/Desktop/SO/Aulas P/Aula04        REMOVED
```

- **Teste 2:** ./spacerate.sh -r output1.txt output2.txt

A tabela é ordenada por ordem inversa à ordem apresentada no "Teste 1".

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1/Work$ ./spacerate.sh -r output1.txt outp
ut2.txt
-837326 /home/tiagoalb/Desktop/SO/Aulas P/Aula04        REMOVED
-365563 /home/tiagoalb/Desktop/SO/Aulas P/Aula01
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula08
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula07
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula05
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula03
792718  /home/tiagoalb/Desktop/SO/Aulas P/Aula02        NEW
```

- **Teste 3:** ./spacerate.sh -a output1.txt output2.txt

A tabela é ordenada por ordem inversa alfabética.

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1/Work$ ./spacerate.sh -a output1.txt outp
ut2.txt
-365563 /home/tiagoalb/Desktop/SO/Aulas P/Aula01
792718  /home/tiagoalb/Desktop/SO/Aulas P/Aula02        NEW
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula03
-837326 /home/tiagoalb/Desktop/SO/Aulas P/Aula04        REMOVED
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula05
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula07
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula08
```

- **Teste 4:** ./spacerate.sh -a -r output1.txt output2.txt

A tabela é ordenada por ordem inversa à ordem apresentada no "Teste 3".

```
tiagoalb@tiagoalb:~/Desktop/SO/Projeto1/Work$ ./spacerate.sh -a -r output1.txt o
utput2.txt
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula08
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula07
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula05
-837326 /home/tiagoalb/Desktop/SO/Aulas P/Aula04        REMOVED
0       /home/tiagoalb/Desktop/SO/Aulas P/Aula03
792718  /home/tiagoalb/Desktop/SO/Aulas P/Aula02        NEW
-365563 /home/tiagoalb/Desktop/SO/Aulas P/Aula01
```

De notar que os ficheiros output's (do script spacecheck.sh) analisados são os seguintes:

- **output1.txt**

```
1
2   Estamos no diretório -> /home/tiagoalb/Desktop/SO/Aulas P
3
4   SIZE    NAME    20231110    /home/tiagoalb/Desktop/SO/Aulas P
5   1048309 /home/tiagoalb/Desktop/SO/Aulas P/Aula03
6   726180  /home/tiagoalb/Desktop/SO/Aulas P/Aula01
7   792718  /home/tiagoalb/Desktop/SO/Aulas P/Aula02
8   349771  /home/tiagoalb/Desktop/SO/Aulas P/Aula07
9   290520  /home/tiagoalb/Desktop/SO/Aulas P/Aula05
10  0   /home/tiagoalb/Desktop/SO/Aulas P/Aula08
11
```

- **output2.txt**

```
1
2   Estamos no diretório -> /home/tiagoalb/Desktop/SO/Aulas P
3
4   SIZE    NAME    20231110    /home/tiagoalb/Desktop/SO/Aulas P
5   1048309 /home/tiagoalb/Desktop/SO/Aulas P/Aula03
6   837326  /home/tiagoalb/Desktop/SO/Aulas P/Aula04
7   360617  /home/tiagoalb/Desktop/SO/Aulas P/Aula01
8   349771  /home/tiagoalb/Desktop/SO/Aulas P/Aula07
9   290520  /home/tiagoalb/Desktop/SO/Aulas P/Aula05
10  0   /home/tiagoalb/Desktop/SO/Aulas P/Aula08
11
```

Conclusão

Por fim, podemos concluir que o script passou com sucesso em todos os testes efetuados e implementa todas as funcionalidades previstas.

Ao longo da resolução do trabalho fomo-nos deparando com algumas dificuldades, nomeadamente:
• Diversidade das opções, e das possibilidades, a ter em conta no script spacecheck.sh;
• Elaboração das funções para os fins pedidos;

Concluindo, com a realização deste trabalho foi possível implementar os conteúdos abordados nas aulas práticas e aprofundar os nossos conhecimentos acerca da programação em bash.

Bibliografia

https://stackoverflow.com

Material do E-learning (guiões práticos)

https://www.vivaolinux.com.br

https://linuxhandbook.com

https://riptutorial.com

https://unix.stackexchange.com