Sistemas Operativos

Taxas de Leitura/Escrita de processos em bash

Trabalho 1

2022/23

Professor Nuno Lau

Professor Guilherme Campos

Gonçalo Sousa, NºMec: 108133

Liliana Ribeiro, NºMec: 108713

universidade de aveiro

Índice

1. Introdução

Este relatório foi elaborado no contexto da disciplina de Sistemas Operativos. Este primeiro trabalho prático consiste no desenvolvimento de um script em bash que apresenta as estatísticas das leituras e escritas dos diferentes processos que está a efetuar. Para isto foi então desenvolvido o script **rwstat.sh**, sendo que lhe foram passados determinados argumentos com informações que o programa terá de passar de acordo com os mesmos.

Para a elaboração deste trabalho recorremos aos diversos materiais disponibilizados para a disciplina, desde apontamentos teóricos aos exercícios desenvolvidos nos guiões práticos. Utilizámos também a plataforma GitHub para criar um repositório colaborativo, de modo a facilitar a partilha de informações entre nós.

2. Estruturação do Código

2.1 Declaração de Variáveis Globais

Primeiramente, começámos por definir arrays associativos, visto que estes permitem guardar valores associados a um "chave" (**key**)

```
#Arrays
declare -A arrayPID=()
declare -A arrayRChar=()
declare -A arrayWChar=()
```

arrayPID Guarda as informações de cada processo que passou pela validação do código, sendo a 'key' o PID

arrayRChar Guarda as linhas rchar

arrayWChar Guarda as linhas wchar

Foram também declaradas algumas variáveis, nomeadamente:

```
TodayDate=$(date +"%s")
regexNum='^[0-9]+([.][0-9]+)?$'
regexDate='^((Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec))
+(0?[1-9]|[12][0-9]|3[01]) +([01]?[0-9]|2[0-3]):[0-5][0-9]'
reverse=0
WOrdem=0
gamPidMin=0

if [ "$(uname -m | grep '64')" != "" ]; then
   gamPidMax=4194304
else
   gamPidMax=32768
fi
```

TodayDate Que representa a data atual em segundos

regexNum Regex de um número inteiro

regexDate Regex de uma data num determinado formato

reverse Que inicialmente toma o valor 0

WOrdem Que inicialmente toma o valor 0

gamPidMin Que inicialmente toma o valor 0

gamPidMax Que, através de uma condição **if,** tomará um valor consoante a arquitetura seja de 32 ou 64 bits

2.2 Tratamento das Opções

2.2.1 Função menu()

A função **menu** serve para expor ao utilizador como executar o script **rwstat.sh,** uma vez que indica as opções disponíveis, a sua funcionalidade e as que necessitam de argumentos.

As opções válidas dividem-se em dois tipos: As que servem para filtrar os argumentos e as que servem para os ordenar.

A opção **-c** permite selecionar os processos através de uma opção regex passada como argumento, fazendo com que só os processos sob essa forma apareçam;

A opção **-s** permite selecionar os processos por data inicial, sendo o argumento a data que pretende que os processos tenham iniciado;

A opção **-e** permite selecionar os processos por data final, sendo o argumento a data que pretende que os processos tenham finalizado;

A opção **-u** permite selecionar apenas os processos que estejam a ser criados por um utilizador em específico;

A opção **-m** e **-M** permitem selecionar os processos através de uma gama específica de PIDS;

A opção **-p** permite ao utilizador selecionar o número de processos que deseja visualizar;

A opção **-r** é uma opção de ordenação reversa e a opção **-w** permite a ordenação da tabela por valores de escrita dos processos;

```
function menu() { # Menu de execução do programa.
   echo "------------------------  Menu de Execução do Programa
------------------------"
   echo
   echo "             ~~~~~~~~~~~~~~~~~~~~ Filtros ~~~~~~~~~~  "
```

```
        echo " -c  -----> Seleção  dos  processos  a  visualizar  pode
ser  realizada através de uma expressão regular."
        echo " -s  -----> Seleção de processos a visualizar num
periodo temporal - data mínima"
        echo " -e  -----> Seleção de processos a visualizar num
periodo temporal - data máxima"
        echo " -u  -----> Seleção de processos a visualizar através do
nome do utilizador"
        echo " -m  -----> Seleção de processos a visualizar através de
uma gama de pids - minimo"
        echo " -M  -----> Seleção de processos a visualizar através de
uma gama de pids - máximo"
        echo " -p  -----> Número de processos a visualizar"
        echo
        echo "        ~~~~~~~~~~~~~~~~  Ordenação (Escolher apenas
uma)  ~~~~~~~~~~~~~        "
        echo " -r  -----> Ordenação reversa"
        echo " -w  -----> Ordenação da tabela por valores de escrita"
        echo
        echo " NOTA: o último argumento terá de ser o número de
segundos pretendido"
        echo
"-------------------------------------------------------------------
-------"

    }
```

2.2.2 getops

getops é inicializado com um ciclo **while** que possui as opções válidas, sendo que as opções estão separadas por **:** sempre que é necessário passar argumentos.

Seguidamente é incluído uma condição **if** que irá verificar que o primeiro caracter do argumento passado como opção é um "**-**" uma vez que se isso se verificar significa que o argumento passado é a próxima opção e que nenhum argumento foi passado como a opção anterior.

A próxima etapa é também uma condição **if** que pega no argumento passado como opção e verifica se este está vazio. Se estiver vazio este guarda no **arrayOpc** com key '**option'** o valor '**empty**'. Caso não

esteja vazio este guarda no **arrayOpc** com key '**option**' o valor do argumento passado.

Depois disto utilizámos um **case** em função da variável '**option**' . Para cada opção, este irá validar o argumento e guardá-lo numa variável própria. Este será também responsável por, caso não seja inserida uma opção válida, mostrar ao utilizador uma mensagem de erro e mostrar novamente o **menu** de modo a que possa ser inserido uma opção válida.

Cada segmento responsável pela filtração dos argumentos no **case** inclui uma condição **if** que verifica que o argumento é passado como regex adequada. Caso isto aconteça, o argumento é guardado numa variável própria. Caso isto não aconteça, será apresentada uma mensagem de erro e novamente o **menu**.

Os segmentos responsáveis pela ordenação dos argumentos irão verificar se a opção **-r** e **-w** têm valor 1. Caso isto aconteça então recorremos a uma condição **if** que irá expor uma mensagem de erro e apresentar também o **menu** para que seja escolhido apenas uma opção de ordenação.

```
while getopts "c:s:e:u:m:M:rw" option; do

    #confere se o argumento passado não é uma suposta opção
    if [[ ${OPTARG:0:1} == - ]]; then
        echo "ERRO --> A opcao -$option requer um argumento"
        echo
        exit 1
    fi

    #----Tratamento de opcoes---

    case $option in
    c)
        #verifica se é uma string
        if [[ $OPTARG =~ $regexNum ]]; then
            echo "ERRO --> Insira uma expressão válida" >&2
            echo
```

```
            menu
            exit 1
        fi

        #Guarda a expressão regular
        expReg=$OPTARG
        ;;

    s)
        #verifica se é uma data
        if ! [[ "$OPTARG" =~ $regexDate ]]; then
            echo "ERRO --> Insira uma data válida" >&2
            echo
            menu
            exit 1
        fi

        #Guarda a data minima
        dateMin=$OPTARG
        dateMin=$(date --date="$dateMin" +"%s")
        ;;

    e)
        #verifica se é uma data
        if ! [[ $OPTARG =~ $regexDate ]]; then
            echo "ERRO --> Insira uma data válida" >&2
            echo
            menu
            exit 1
        fi

        #Guarda a data maxima
        dateMax=$OPTARG
        dateMax=$(date --date="$dateMax" +"%s")
        ;;

    u)
        #verifica se é uma string
        if [[ $OPTARG =~ $regexNum ]]; then
            echo "ERRO --> Insira um nome de utilizador válido"
>&2
            echo
            menu
```

```
            exit 1
        fi

        #Guarda o nome de utilizador
        utilizador=$OPTARG
        ;;

    m)
        #verifica se é um número
        if ! [[ $OPTARG =~ $regexNum ]]; then
            echo "ERRO --> Insira um PID válido" >&2
            echo
            menu
            exit 1
        fi

        #Guarda o PID
        gamPidMin=$OPTARG
        ;;

    M)
        #verifica se é um número
        if ! [[ $OPTARG =~ $regexNum ]]; then
            echo "ERRO --> Insira um PID válido" >&2
            echo
            menu
            exit 1
        fi

        #Guarda o PID
        gamPidMax=$OPTARG
        ;;

    p)
        #verifica se é um número
        if ! [[ $OPTARG =~ $regexNum ]]; then
            echo "ERRO --> Insira uma número de processos válido"
>&2
            echo
            menu
            exit 1
        fi
```

```
            #Guarda o PID
            nProc=$OPTARG
            ;;

        r)
            reverse=1
            ;;

        w)
            WOrdem=1
            ;;
        *)
            #O argumento passado não está listado
            echo "insira uma das opções listadas" >&2
            echo
            menu
            exit 1
            ;;
        esac
    done
```

2.2.3 Validação dos argumentos passados

Para verificar se é passado como último argumento o número de segundos e que é passado pelo menos um argumento é necessário uma condição **if** que irá mostrar uma mensagem e erro e mostrar novamente o **menu** caso isto não aconteça.

Como exposto no **getops**, caso ambas as opções **-r** e **-w** tenham valor 1, recorremos a uma condição **if** que irá expor uma mensagem de erro e apresentar também o **menu** para que seja escolhido apenas uma opção de ordenação.

```
    if ! [[ ${@: -1} =~ $regexNum ]]; then
        echo "ERRO --> Insira como último argumento o número de
segundos que pretende"
        echo
        menu
        exit 1
    fi
```

```
LastArg=${@: -1}

if [[ "$reverse" -eq 1 && "$WOrdem" -eq 1 ]]; then
   echo "ERRO --> Insira apenas uma ordem"
   echo
   menu
   exit 1
fi
```

2.3 Leitura de rchar e wchar

2.3.1 Função processos()

A função processos é iniciada por um ciclo for que irá ler todos os ficheiros dentro do diretório e, recorrendo a uma condição if, irão ser filtrados os ficheiros que não estão no format /proc/[PID] e será visto se está dentro da gama de PIDS introduzido.

Em seguida será novamente utilizada uma condição if que irá verificar se o file io e comm existem e estão no modo de leitura no diretório PID.

O próximo passo será verificar, para cada opção (**-c**, **-s**, **-e**, **-u**), se a respetiva variável existe e, caso exista, filtrar os ficheiros que não seguem as condições. No caso do nome do protocolo e do utilizador, recorremos a duas condições **if** para verificar as duas condições acima referidas, sendo que definimos também as variáveis **XExpReg** e **XUtilizador**. No caso da data mínima e máxima, é necessário especificar o formato da data que se deve introduzir para depois utilizar as condições **if** (especificada acima). Também foi necessário definir as variáveis **XDate**, **dateSeg**, **dateMin** e **dateMax**.

Em seguida guardámos estas informações num array associativo 2D em que a 1ª key é o **PID** do processo e a 2ª key é a informação que vamos guardar relativa ao **PID**

Posteriormente, iremos voltar a ler e guardar no **arrayRChar** e **arrayWChar** (com key **PID**) os valores **rchar** e **wchar,** recorrendo a **grep** e **tr**.

A seguir recorremos ao comando **sleep** que corresponde ao intervalo de tempo dado colocado pelo utilizador. O programa entrará depois no ciclo **for** que utilizada novamente os valores **Rchar** e **Wchar** antes do sleep time e depois do sleep time para fazer comparações. Depois é calculado o **rateR** e o **rateW**, que irão ser guardados no array de informação **arrayPID.**

```
function processos() {

    cd /proc
    for PID in $(ls -a); do

        if ! [[ "$PID" =~ $regexNum && "$PID" -ge $gamPidMin &&
"$PID" -le $gamPidMax ]]; then
            continue
        fi


        if ! [[ -f "$PID/io" && -f "$PID/comm" && -r "$PID/io" &&
-r "$PID/comm" ]]; then
            continue
        fi

        XExpReg=$(cat $PID/comm | tr " " "_")
        if [[ -n $expReg ]]; then
            if ! [[ $XExpReg =~ $expReg ]]; then
                continue
            fi
        fi

        #utilizador
        XUtilizador=$(ps -o user= -p $PID)
        if [[ -n $utilizador ]]; then
            if ! [[ $XUtilizador =~ $utilizador ]]; then
                continue
            fi
        fi
```

```
            #data minima e data maxima
            LANG=en_us_8859_1
            XDate=$(ps -o lstart= -p $PID)
            dateSeg=$(date --date="$XDate" +"%s")
            if [[ -n $dateMin ]]; then
                if ! [[ $dateSeg -ge $dateMin ]]; then
                    continue
                fi
            fi

            if [[ -n $dateMax ]]; then
                if ! [[ $dateSeg -le $dateMax ]]; then
                    continue
                fi
            fi

            #Guardar a informação no array associativo 2D:
            arrayPID[$PID, COMM]=$XExpReg
            arrayPID[$PID, USER]=$XUtilizador
            arrayPID[$PID, DATE]=$XDate

            #Guardar os valores de rchar e wchar
            rchar=$(cat $PID/io | grep rchar | tr -dc '0-9')
            wchar=$(cat $PID/io | grep wchar | tr -dc '0-9')
            arrayPID[$PID, READB]=$rchar
            arrayPID[$PID, WRITEB]=$wchar
            arrayRChar[$PID]=$rchar
            arrayWChar[$PID]=$wchar

        done

        #damos o intervalo de tempo colacado pelo utilizador
        sleep $LastArg

        #Buscar denovo os valores RChar e WChar para depois fazer as
comparações
        for PID in "${!arrayRChar[@]}"; do #Nota: aqui usamos as keys
do array: arrayRChar, mas poderiamos usar as keys do array: arrayWChar

            #rchar e wchar antes do sleep time
            rcharOld=${arrayRChar[$PID]}
            wcharOld=${arrayWChar[$PID]}
```

```
            #rchar e wchar depois do sleep time
            rcharNew=$(cat $PID/io | grep rchar | tr -dc '0-9')
            wcharNew=$(cat $PID/io | grep wchar | tr -dc '0-9')

            #calcular o rateR
            sub=$(($rcharNew-$rcharOld))
            rateR=$(echo "scale=2; $sub/$LastArg" | bc -l)
            #calcular o rateW
            sub=$wcharNew-$wcharOld
            rateW=$(echo "scale=2; $sub/$LastArg" | bc -l)

            #Guardar o rateR e o rateW no array de informação
            arrayPID[$PID, RATER]=$rateR
            arrayPID[$PID, RATEW]=$rateW

        done

    }
```

3. Função print

Para darmos print a valores utilizámos uma função print com diversos ciclos e condições para dar display à informação recorrendo a **printf**'s

```
function print() {

   #Começamos por ver o número de processos que vamos imprimir
   #Se o utilizador não tiver definido nenhum nProc mostramos todos os
processos
   if ! [[ -n $nProc ]]; then
       nProc=${#arrayRChar[@]}
   fi

   printf "%-30s %-20s %5s %15s %15s %15s %15s %15s %15s %16s\n" "COMM"
"USER" "PID" "READB" "WRITEB" "RATER" "RATEW" "DATE"

   for PID in "${!arrayRChar[@]}"; do
```

```
        printf "%-30s %-20s %5s %15s %15s %15s %15s %15s %15s %8s %-1s
%-1s \n" "${arrayPID[$PID, COMM]}" "${arrayPID[$PID, USER]}" "$PID"
"${arrayPID[$PID, READB]}" "${arrayPID[$PID, WRITEB]}"
"${arrayPID[$PID, RATER]}" "${arrayPID[$PID, RATEW]}" "${arrayPID[$PID,
DATE]}" | head -n ${nProc}

    done

}
```

4. Testagem dos resultados

4.1 Teste 1 - Sem o número de segundos

```
goncalo@goncalo-VivoBook-ASUS:~/SO/projeto/ProjetoSO$ ./rwstat
ERRO --> Insira como último argumento o número de segundos que pretende

-----------------------  Menu de Execução do Programa  -----------------------

            ~~~~~~~~~~~~~~~~~~~~~ Filtros  ~~~~~~~~~~~~~~~~~~~~~
 -c  -----> Seleção  dos  processos  a  visualizar  pode  ser  realizada através de uma expressão regular.
 -s  -----> Seleção de processos a visualizar num periodo temporal - data mínima
 -e  -----> Seleção de processos a visualizar num periodo temporal - data máxima
 -u  -----> Seleção de processos a visualizar através do nome do utilizador
 -m  -----> Seleção de processos a visualizar através de uma gama de pids - minimo
 -M  -----> Seleção de processos a visualizar através de uma gama de pids - máximo
 -p  -----> Número de processos a visualizar

       ~~~~~~~~~~~~~~~~  Ordenação (Escolher apenas uma)  ~~~~~~~~~~~~~
 -r  -----> Ordenação reversa
 -w  -----> Ordenação da tabela por valores de escrita

 NOTA: o último argumento terá de ser o número de segundos pretendido
--------------------------------------------------------------------------------
```

4.2 Teste 2 - opção de data inválida

```
goncalo@goncalo-VivoBook-ASUS:~/SO/projeto/ProjetoSO$ ./rwstat -s "Dec 32 23:20" 10
ERRO --> Insira uma data válida

-----------------------  Menu de Execução do Programa  -----------------------

            ~~~~~~~~~~~~~~~~~~~~~ Filtros  ~~~~~~~~~~~~~~~~~~~~~
 -c  -----> Seleção  dos  processos  a  visualizar  pode  ser  realizada através de uma expressão regular.
 -s  -----> Seleção de processos a visualizar num periodo temporal - data mínima
 -e  -----> Seleção de processos a visualizar num periodo temporal - data máxima
 -u  -----> Seleção de processos a visualizar através do nome do utilizador
 -m  -----> Seleção de processos a visualizar através de uma gama de pids - minimo
 -M  -----> Seleção de processos a visualizar através de uma gama de pids - máximo
 -p  -----> Número de processos a visualizar

       ~~~~~~~~~~~~~~~~  Ordenação (Escolher apenas uma)  ~~~~~~~~~~~~~
 -r  -----> Ordenação reversa
 -w  -----> Ordenação da tabela por valores de escrita

 NOTA: o último argumento terá de ser o número de segundos pretendido
--------------------------------------------------------------------------------
```

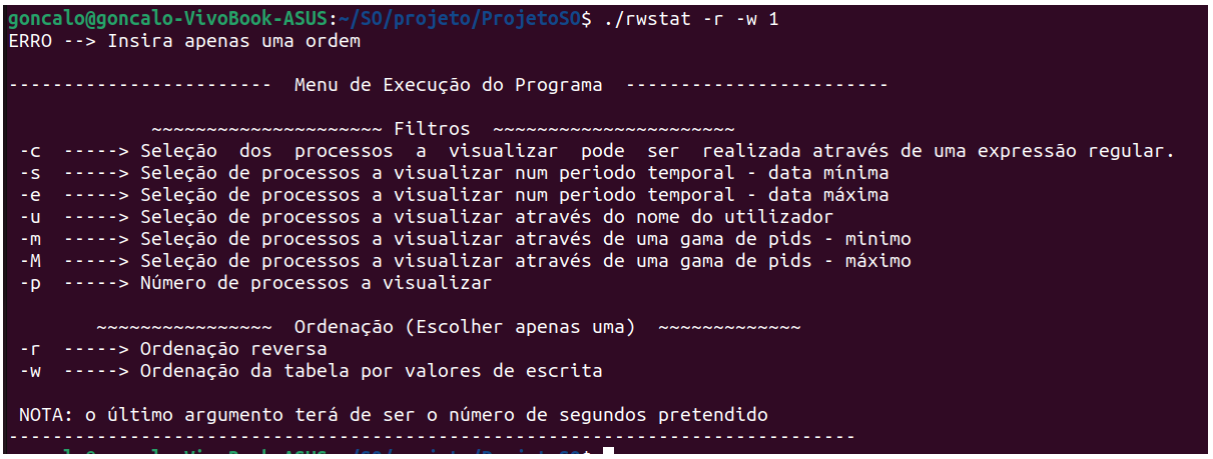
4.3 Teste 3 - ordenar a memória por reverse (-r)

```
goncalo@goncalo-VivoBook-ASUS:~/SO/projeto/ProjetoSO$ ./rwstat -r 1
COMM                 USER                 PID          READB          WRITEB          RATER          RATEW           DATE
chrome               goncalo           125322        1754345        17003237              0              0 Sat Dec  3 03:53:17 2022
chrome               goncalo           125326       71755529       166882937           9.00           9.00 Sat Dec  3 03:53:17 2022
chrome               goncalo           126042       12791155        3673940              0              0 Sat Dec  3 03:53:33 2022
chrome               goncalo           125687      431036947        3742249           1.00           1.00 Sat Dec  3 03:53:20 2022
chrome               goncalo           125584        1364742         121673              0              0 Sat Dec  3 03:53:20 2022
cat                  goncalo           125285          26279          22071              0              0 Sat Dec  3 03:53:17 2022
cat                  goncalo           125284           4208              0              0              0 Sat Dec  3 03:53:17 2022
goa-identity-se      goncalo             2328         574303         175377              0              0 Fri Dec  2 17:21:39 2022
chrome_crashpad      goncalo           125287           7480             64              0              0 Sat Dec  3 03:53:17 2022
gsd-disk-utilit      goncalo             2654          24575           3140              0              0 Fri Dec  2 17:21:41 2022
gnome-session-b      goncalo             2320       10938394          64277              0              0 Fri Dec  2 17:21:38 2022
whatsapp-4linux      goncalo            93757         205342        2295562              0              0 Fri Dec  2 22:59:16 2022
chrome_crashpad      goncalo           125289           7480              0              0              0 Sat Dec  3 03:53:17 2022
slack                goncalo             4960          85954             20              0              0 Fri Dec  2 17:22:18 2022
gnome-keyring-d      goncalo             2158         640857        1193446              0              0 Fri Dec  2 17:21:38 2022
chrome_crashpad      goncalo             5244           6404              0              0              0 Fri Dec  2 17:22:20 2022
evinced              goncalo            14095          15599           1905              0              0 Fri Dec  2 18:49:40 2022
chrome               goncalo           157168         899782         527587              0              0 Sat Dec  3 05:25:55 2022
ibus-engine-sim      goncalo             2783         503081         769224              0              0 Fri Dec  2 17:21:41 2022
chrome               goncalo           125968        1044929         453255              0              0 Sat Dec  3 03:53:31 2022
chrome               goncalo           125821         109454            319              0              0 Sat Dec  3 03:53:25 2022
```

4.4 Teste 4 - ordenar a memória por write values (-w)

```
goncalo@goncalo-VivoBook-ASUS:~/SO/projeto/ProjetoSO$ ./rwstat -w 1
COMM                 USER                 PID          READB          WRITEB          RATER          RATEW           DATE
chrome               goncalo           125322        1755786        17057747              0              0 Sat Dec  3 03:53:17 2022
chrome               goncalo           125326       71772985       167774033              0              0 Sat Dec  3 03:53:17 2022
chrome               goncalo           126042       12791184        3673969              0              0 Sat Dec  3 03:53:33 2022
chrome               goncalo           125687      433550731        3742825              0              0 Sat Dec  3 03:53:20 2022
chrome               goncalo           125584        1369294         121683              0              0 Sat Dec  3 03:53:20 2022
cat                  goncalo           125285          26279          22071              0              0 Sat Dec  3 03:53:17 2022
cat                  goncalo           125284           4208              0              0              0 Sat Dec  3 03:53:17 2022
goa-identity-se      goncalo             2328         574647         175441              0              0 Fri Dec  2 17:21:39 2022
chrome_crashpad      goncalo           125287           7480             64              0              0 Sat Dec  3 03:53:17 2022
gsd-disk-utilit      goncalo             2654          24575           3140              0              0 Fri Dec  2 17:21:41 2022
gnome-session-b      goncalo             2320       10938394          64277              0              0 Fri Dec  2 17:21:38 2022
whatsapp-4linux      goncalo            93757         205357        2295562              0              0 Fri Dec  2 22:59:16 2022
chrome_crashpad      goncalo           125289           7480              0              0              0 Sat Dec  3 03:53:17 2022
slack                goncalo             4960          85954             20              0              0 Fri Dec  2 17:22:18 2022
gnome-keyring-d      goncalo             2158         641001        1193734              0              0 Fri Dec  2 17:21:38 2022
chrome_crashpad      goncalo             5244           6404              0              0              0 Fri Dec  2 17:22:20 2022
evinced              goncalo            14095          15599           1905              0              0 Fri Dec  2 18:49:40 2022
chrome               goncalo           157168         899793         527598              0              0 Sat Dec  3 05:25:55 2022
ibus-engine-sim      goncalo             2783         504713         771744              0              0 Fri Dec  2 17:21:41 2022
chrome               goncalo           125968        1044939         453265              0              0 Sat Dec  3 03:53:31 2022
chrome               goncalo           125821         109455            320              0              0 Sat Dec  3 03:53:25 2022
```

4.5 Teste 5 - Introduzir os dois tipos de ordenação (-r e -w)



5. Resultados finais

Após fazermos a testagem do código e analisarmos os resultados podemos concluir que conseguimos com sucesso desenvolver um scrip em bash capaz de obter estatísticas acerca da leitura e escrita de processos em bash.

6. Conclusão

Ao longo deste trabalho conseguimos consolidar os conhecimentos teóricos e práticos sobre processos em bash. Através de tentativa e erro conseguimos desenvolver e aplicar as metodologias necessárias para a realização do **script** e aprofundar os nossos conhecimentos de **bash**.

7. Bibliografia

Utilizámos o material disponibilizado nas aulas teóricas bem como os guiões e exercícios das aulas práticas, bem como o manual de bash disponibilizado no terminal do ubuntu e o livro “Sistemas Operativos” de José Alves Marques.

No caso de algumas dúvidas acerca do desenvolvimento de certos métodos e funções recorremos também a diversos sites web, mais notavelmente o stackoverflow.com.