



universidade  
de aveiro



# Taxas de Leitura/Escrita de processos em bash

Sistemas Operativos – Professor Nuno Lau

Departamento de Eletrónica, Telecomunicações e Informática

Universidade de Aveiro

Bernardo Pinto – 105926  
José Mendes - 107188

## Índice

|   |    |
|---|----|
| Introdução                                      | 2  |
| Abordagem do problema                           | 3  |
| Tratamento de argumentos                        | 6  |
| Obtenção dos processos e respetivas informações | 8  |
| Funcionamento concreto das opções               | 10 |
| Testes realizados                               | 18 |
| Testes de erro                                  | 20 |
| Conclusão                                       | 22 |
| Bibliografia                                    | 23 |

## Introdução

No âmbito da disciplina de Sistemas Operativos, o presente trabalho prático incide sobre a obtenção de estatísticas sobre a leitura e escrita (número de bytes de I/O) dos vários processos que estão em execução. Como aprendemos nesta Unidade Curricular, um processo é um programa em execução, que pode ser criado quando o sistema é iniciado, através de uma chamada ao sistema por um processo ou pelo pedido de um utilizador.

Assim, o objetivo deste trabalho prático é o desenvolvimento de um script em bash para obter estatísticas sobre as leituras e escritas que os processos estão a efetuar. Esta ferramenta permite visualizar o número de bytes de I/O que um processo leu/escreveu e também a taxa de leitura/escrita num determinado período de tempo.

Ao longo deste documento será apresentada a metodologia utilizada no desenvolvimento deste trabalho. Em complemento, estão presentes todas as etapas e respetivas descrições, todos os tratamentos de erros e testes realizados para garantir que tudo foi analisado.

## Abordagem do problema

Inicialmente, começamos por desenvolver uma maneira de guardar todas as informações inseridas pelo usuário como argumento do programa **rwstat** num *array* declarado acima como **user\_input**. Este *array* será preenchido com todos os parâmetros passados ao script, utilizando o comando “\$@”, para que essa informação fique acessível posteriormente.

```
declare -a user_input;
cont=0

for i in $@
do
    user_input[cont]=$i # declaração do array para salvar e deixar acessível os valores de $@
    cont=$((cont+1))
done
```

Figura 1 - Preenchimento do array com valores de \$@

Após o armazenamento da informação, foi necessário verificar o tamanho do *array*, através do comando “\${#user\_input[@]}”, para que fosse possível extrair apenas a informação referente ao tempo inserido. Teve-se como possibilidade os seguintes casos:

- **Tamanho igual a 1** – O tempo encontra-se no índice 0 do *array*;
- **Tamanho superior a 1** – O tempo encontra-se no índice -1 do *array*;

```
if [[ ${#user_input[@]} -eq 1 ]];then
tempo=${user_input[0]} # ./rwstat.sh 10

elif [[ ${#user_input[@]} -gt 1 ]];then
tempo=${user_input[-1]}

fi
```

Figura 2 - Atribuição de valor a variável \$tempo

Posteriormente, é declarada uma função que retorna uma mensagem de erro, **error\_message()**, que vai exibir as instruções para o uso correto do programa e das suas opções, caso o utilizador proceda incorretamente ao seu uso.

```
error_message(){
    echo "
TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROSTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de
I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os items aparecem na tabela (reverse);
-w : ordena os items da tabela através dos valores 'escritos' por cada um (write values);
"
}
```

Figura 3 - Instruções de utilização

Em seguida foram implementadas estruturas condicionais para realizar a validação do tempo (i.e. o tempo deve ser um número inteiro) e, em caso positivo, testar se a opção que antecede o tempo (índice -2) necessita ou não de um argumento. Uma vez que, obrigatoriamente, o último argumento fornecido ao script já representa o tempo (índice -1).

```
if [[ ! $tempo =~ ^[0-9]+$ ]];then #verifico se o ultimo parametro é um numero
    echo "0 ultimo argumento deve ser um numero (tempo em segundos)"
    error_message
    exit 1
elif [[ ${#user_input[@]} -gt 1 ]] && [[ ${user_input[-2]} = "-c" || ${user_input[-2]} = "-u" || ${user_input[-2]} = "-p" ||
${user_input[-2]} = "-s" || ${user_input[-2]} = "-e" || ${user_input[-2]} = "-m" || ${user_input[-2]} = "-M" ]];then
# verifico se a informação presente no índice -2 necessita de argumento
    echo "0 argumento "${user_input[-2]}" necessita de um argumento"
    error_message
    exit 1
fi
```

Figura 4 - Validação do tempo e índice -2

Ainda na fase de seleção dos argumentos inseridos, foi implementada, através de estruturas condicionais e de repetição, uma funcionalidade que **separa as opções escolhidas do tempo** e armazena em uma nova variável denominada “\$argumentos\_validos”.

Para desenvolver essa funcionalidade foi necessário, primeiramente, testar se tinham sido inseridas opções (i.e. *array* de opções com tamanho superior a um) ou se apenas o tempo tinha sido passado como parâmetro ao script (i.e. tamanho do array igual a um), e com isso temos as seguintes possibilidades:

- **Tamanho do array igual a um** – apenas o tempo é salvo na variável “\$argumentos\_validos”
- **Tamanho do array superior a um** – todos os argumentos, exceto o tempo, são salvos na variável “\$argumentos\_validos”

Por fim, uma vez que os argumentos inseridos estão contidos numa só variável, foi implementado um ciclo *for* para percorrê-la e, caso necessário, concatenar a opção e o caractere “:” que representa a necessidade de a opção ser seguida de um argumento. No caso da opção não necessitar de um argumento, esta será concatenada apenas à sua respectiva letra. Após o fim do ciclo *for*, todas as opções escolhidas pelo usuário estarão presentes na mesma variável \$options, que será utilizada posteriormente no **getopts**.

```
todos_argumentos=$@

if [[ ${#user_input[@]} -gt 1 ]];then
argumentos_validos=${todos_argumentos:0:-${${#tempo}}} # selecionando as opções passadas sem o ultimo parametro (tempo).
else
argumentos_validos=todos_argumentos # ./rwstat.sh 10
fi

for j in $argumentos_validos
do
if [[ $j = "-c" || $j = "-u" || $j = "-p" || $j = "-s" || $j = "-e" || $j = "-m" || $j = "-M" ]];then
options="$options"${j:1}":" # concatenar string de opções que necessitam de argumentos (:)
elif [[ $j = "-r" || $j = "-w" ]];then
options="$options"${j:1} # concatenar string de opções que nao necessitam de argumentos
fi
done
```

Figura 5 - Seleção dos argumentos inseridos

```
bernardo~$ ./rwstat.sh -u 'bernardo' -r -c '^code.*' 10

tempo: 10
"$todos_argumentos": -u bernardo -r -c ^code.* 10
"$argumentos_validos:" -u bernardo -r -c ^code.*
Options: u:rc:
```

Figura 6 - Exemplo de valores atribuídos as variáveis

## Tratamento de argumentos:

Em seguida utilizamos o comando interno “**getopts**” para analisar e tratar os argumentos inseridos na linha de comando.

Inicialmente foi declarada, para cada opção, uma *flag* com valor inicial zero, que significa que aquela opção não está ativa no momento, no entanto, sempre que a opção é utilizada, o valor da sua *flag* é incrementado e adquire um valor diferente de zero. Este indica que aquela opção está a ser utilizada.

Para além disso, em opções que não são de ordenação, foi instanciada uma nova variável que receberá o valor do **OPTARG** daquela opção, ou seja, o local onde o valor inserido pelo usuário como argumento daquela opção vai ser salvo. Por exemplo:

- **-u ‘Bernardo’ 10** – A opção de seleção por nome de utilizador terá o **OPTARG** igual a **‘Bernardo’**, que será salvo na variável “**\$name\_user**”
- **-m ‘1000’ 10** – A opção de seleção por gama de PIDS terá o **OPTARG** igual a **“1000”**, que será salvo na variável “**\$pids1**”

Na sequência, implementamos uma **estrutura condicional** para testar se duas opções tinham sido escolhidas mais do que uma vez, o que faz com que a sua *flag* adquirisse um valor superior a 1 e resultasse em uma mensagem de erro seguida do encerramento do programa.

```
if [[ c_flag -gt 1 ]] || [[ u_flag -gt 1 ]] || [[ p_flag -gt 1 ]] || [[ s_flag -gt 1 ]] || [[ e_flag -gt 1 ]] || [[ m_flag -gt 1 ]] || [[ M_flag -gt 1 ]] || [[ c_flag -gt 1 ]] || [[ r_flag -gt 1 ]] || [[ w_flag -gt 1 ]]; then
    echo "Foram escolhidas opções repetidas"
    error_message
    exit 1
fi
```

Figura 5 - Bloco condicional que verifica a inserção de opções repetidas

```

c_flag=0
u_flag=0
p_flag=0
s_flag=0
e_flag=0
m_flag=0
M_flag=0
r_flag=0
w_flag=0

while getopts "$options" opt; do

    case "${opt}" in
        c )
            name_comm=${OPTARG}
            c_flag=$((c_flag+1))
            ;;

        u )
            name_user=${OPTARG}
            u_flag=$((u_flag+1))
            ;;

        p )
            num_proc=${OPTARG}
            p_flag=$((p_flag+1))
            ;;

        s )
            data_min=${OPTARG}
            datamin=`date +%Y/%m/%d" -d "${data_min}"`
            horamin=$(date -u -d `echo "${data_min:7}"` +%s")
            s_flag=$((s_flag+1))
            ;;

        e )
            data_max=${OPTARG}
            datamax=`date +%Y/%m/%d" -d "${data_max}"`
            horamax=$(date -u -d `echo "${data_max:7}"` +%s")
            e_flag=$((e_flag+1))
            ;;

        m )
            pids1=${OPTARG}
            m_flag=$((m_flag+1))
            ;;

        M )
            pids2=${OPTARG}
            M_flag=$((M_flag+1))
            ;;

        ##### ORDENAÇÃO (NAO NECESSITAM DE ARGUMENTOS) #####

        r )
            r_flag=$((r_flag+1))
            ;;

        w )
            w_flag=$((w_flag+1))
            ;;

    esac

done

```

Figura 6 - Comando getopts



## Obtenção dos processos e respetivas informações

Numa visão geral, no início é utilizado o comando **cd /proc** para aceder á diretoria **/proc**, uma vez que esta possui os **PID's** que contém informações, como o nome do processo e o número de bytes de leitura e escrita, essenciais para este trabalho prático.

Iniciámos dois *loops for* para percorrer todos os processos denominados com números (pois na diretoria **/proc** as diretorias que são números representam os **PID's**), um antes do comando **sleep \$tempo** (valor introduzido pelo utilizador), e um depois.

Simplificadamente, após ler os valores de **READB** e **WRITEB** no primeiro *for*, vamos fazer um **sleep** e, de seguida, o segundo *for* faz a leitura de novos valores destas variáveis para ser possível calcular a taxa de escrita e leitura.

```
cd /proc # acede à diretoria
for pid in *[0-9] ; do # for loop para todos os itens com numeros (neste caso só os PIDs)
    if [[ -f "$pid/io" ]] && [[ -r "$pid/io" ]] && [[ -f "$pid/comm" ]] && [[ -f "$pid/status" ]]; then # ve se
        /proc/$pid/io file existe e é readable
        # (-f --> vê se existe do tipo FILE) (-r --> vê se é readable); vê se comm e status de cada pid      também sao
        readable

        PIDs+=($pid) # adiciona cada PID ao array
        READB=$(cat $pid/io | grep -i 'rchar' | grep -o -E '[0-9]+')
        read_bytes+=($READB) # adiciona o rchar de cada PID ao array
        WRITEB=$(cat $pid/io | grep -i 'wchar' | grep -o -E '[0-9]+')
        writen_bytes+=($WRITEB) # adiciona o wchar de cada PID ao array
        #read_bytes e writen_bytes vão ser usados para calcular o resto (raters) depois do sleep
    fi
done
```

Figura 7 - Captura da informação

Para que seja possível se lerem os valores e obter informações sobre cada **PID**, é necessário verificar se as subdiretorias **/io**, **/comm** e **/status** existem (é feito ao usar **-f** que verifica se existe e é ficheiro) e verificar se a subdiretoria **/io** pode ser lida (pois em muitos casos o utilizador não tem permissões para tal).

De seguida, para guardar dados que serão úteis no segundo *loop for*, após o *sleep*, foram declarados três *arrays* anteriormente: **PIDs**, que guarda todos os **PID** lidos, **read\_bytes**, guarda a quantidade de bytes I/O lidos por cada processo, e **writen\_bytes**, que guarda a quantidade de bytes de I/O escritos por cada processo.

Inicialmente, no primeiro *loop for*, usamos os comandos **cat** para ler o ficheiro e **grep** (-i "<string>" para apenas ficar a variável desejada e -o -E '[0-9]+' para ficar somente os dígitos).

```

declare -a PIDs # declara array que guarda todos os PIDs na diretoria /proc/
declare -a read_bytes # array que guarda o rchar de cada PID
declare -a writen_bytes # array que guarda o wchar de cada PID
declare -a info; #declarar array que vai guardar todas as informações de cada PID

for (( i = 0; i < ${#PIDs[@]}; i++ )); do
    pid="${PIDs[$i]}" #vai buscar o pid na posição i
    if [[ -f "$pid/io" ]] && [[ -r "$pid/io" ]] && [[ -f "$pid/comm" ]] && [[ -f "$pid/status" ]]; then # as mesmas
    verificações de cima
        COMM=$(cat $pid/comm | tr " " "_") # este tr é usado para substituir os espaços por um "_" para não dar erro
        futuramente
        USER=$(ps -o uname= -p "$pid")
        DATE=$(ls -ld "$pid" | awk '{printf("%s %s %s\n", toupper( substr( $6, 1, 1 ) ) substr( $6, 2 ), $7, $8)}')
        #toupper e substr --> o primeiro char do mês maiusculo
        READB2=$(cat $pid/io | grep -i 'rchar' | grep -o -E '[0-9]+')
        WRITEB2=$(cat $pid/io | grep -i 'wchar' | grep -o -E '[0-9]+')
        RATER=$(echo "scale=2; ($READB2-${read_bytes[$i]})/$tempo" | bc -l) # scale=2 para ter 2 casas decimais e
        RATEW=$(echo "scale=2; ($WRITEB2-${writen_bytes[$i]})/$tempo" | bc -l)

        READB=$(expr $READB2 - ${read_bytes[$i]})
        WRITEB=$(expr $WRITEB2 - ${writen_bytes[$i]})

        info+=($COMM $USER $pid $READB $WRITEB $RATER $RATEW $DATE) # guarda toda a informação de cada PID
    fi
done

```

Figura 8 - Captura da informação

De seguida, é utilizado o comando **sleep \$tempo**, em que tempo é o valor introduzido pelo utilizador. Este comando vai permitir calcular as taxas de leitura/escrita, daí o número de segundos ser um parâmetro obrigatório introduzido.

Neste segundo *loop for*, é criada uma variável para cada coluna que vai estar presente na tabela final (**COMM**, **USER**, **DATE**, **RATER**, **RATEW** e cada valor que preenche os *arrays* **read\_bytes** e **writen\_bytes** é usado para calcular os **READB** e **WRITEB**). Os valores de cada variável são obtidos através dos comandos **cat**, **tr**, **ps**, **ls**, **awk**, **grep**, **echo** e **bc**, utilizando também opções de certos comandos.

Este segundo *for* garante que apenas serão lidos os processos que fizeram parte do primeiro *loop* e caso um processo deixe de existir durante o tempo de **sleep**, não fará parte deste *for*.

As verificações são iguais às do primeiro *loop*, a única diferença é que este vai usar apenas processos que já faziam parte antes do **sleep**. Além de ser mais eficiente, não vão ocorrer futuros erros, dado que apenas procura processos ou usados antes ou que deixaram de existir durante o intervalo de tempo. Cada valor necessário é guardado em variáveis referidas anteriormente e são calculadas as taxas de bytes lidos e escritos ao subtrair os novos valores (**READB2** e **WRITEB2**), e os valores que foram calculados antes do **sleep**, guardados em *arrays* anteriormente declarados (**read\_bytes[\$i]** e **writen\_bytes[\$i]**) e dividir pelo número de segundos definidos pelo utilizador (**\$tempo**).

Todos os valores vão ser guardados dentro de **info**, *array* anteriormente declarado, e vão ser usados para aplicação de filtros introduzidos pelo utilizador. Posteriormente irá se imprimir a tabela.

## Funcionamento concreto das opções:

Primeiramente, é válido lembrar que quando uma opção é chamada na linha de comando a sua respectiva *flag* é ativada recebendo um valor diferente de zero. Portanto, o funcionamento de todas as opções é iniciado com um bloco condicional para testar se a *flag* daquela opção está ativa ou não.

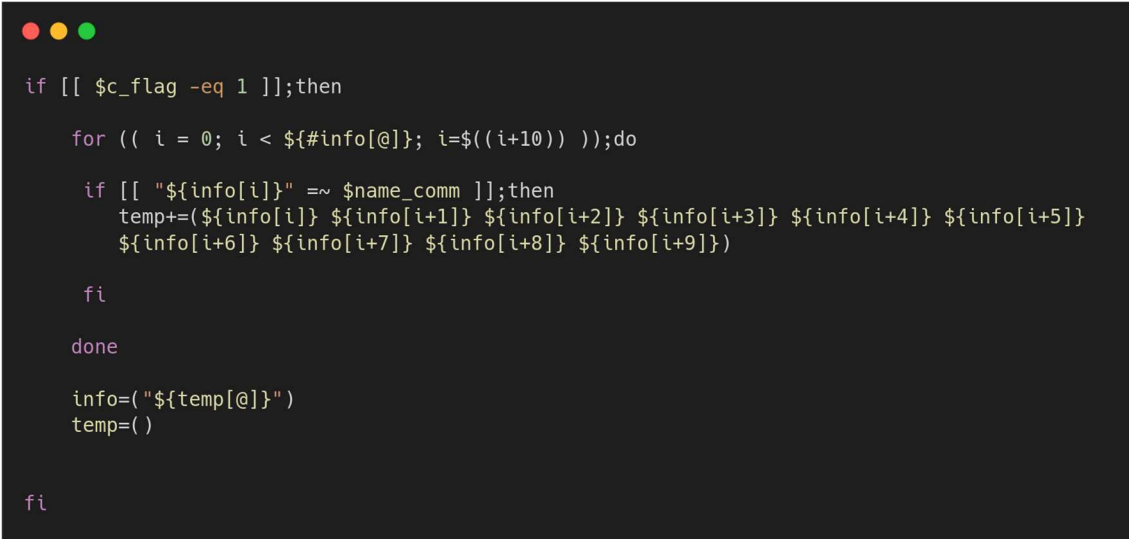
Além disso, foi utilizado um *array* temporário “*temp*” para que a informação filtrada fosse armazenada temporariamente ao decorrer de cada opção e depois repassada para o *array* final “*info*”. No fim de cada implementação do filtro, os **PIDs** pretendidos e as suas respectivas informações são adicionadas ao *array* final, e o *array* temporário é esvaziado para poder ser reutilizado.

- **Opção “-c” – Seleção dos processos através do comando associado:**

Sabendo que a primeira informação de comando associado está armazenada no **índice 0**, e que cada processo ocupa **10 índices**, é gerado assim um padrão. A partir desta informação, foi implementado um ciclo *for* com parâmetro inicial igual a 0, sempre menor do que o tamanho do *array info* e que é incrementado de 10 em 10 índices.

A partir desse ponto já temos acesso, dentro do ciclo *for*, apenas às informações dos comandos associados, que poderão ser comparadas, através do operador de correspondência de expressão regular “*=~*”, com o **OPTARG** da opção **-c** que está armazenado na variável “*\$name\_comm*”.

Uma vez que a comparação é verdadeira o *array* temporário é incrementado com toda a informação daquele processo



```
if [[ $c_flag -eq 1 ]];then
    for (( i = 0; i < ${#info[@]}; i=$((i+10)) ));do
        if [[ "${info[i]}" =~ $name_comm ]];then
            temp+=("${info[i]} ${info[i+1]} ${info[i+2]} ${info[i+3]} ${info[i+4]} ${info[i+5]}
                ${info[i+6]} ${info[i+7]} ${info[i+8]} ${info[i+9]})"
        fi
    done
    info=( "${temp[@]}" )
    temp=( )
fi
```

Figura 9 – Bloco da opção “-c”

- **Opção “-u” – Seleção dos processos através do nome do utilizador:**

Seguindo a mesma logica da opção -c a seleção do processo através do nome do utilizador é iniciado com a implementação de um ciclo *for*, que tem como parâmetro uma variável com valor inicial 1, sempre menor do que o tamanho do *array info* e que é incrementada com valor 10. Uma vez que a informação do nome do utilizador está armazenada no índice 1 e um processo completo ocupa 10 índices com suas informações.

Em seguida, é realizada a comparação, “==”, entre o **nome do utilizador** naquele índice e o **OPTARG** da opção, que está armazenado na variável “\$name\_user”.

Por fim, considerando que a comparação retornou **TRUE**, o *array* temporário é preenchido com as informações daquele processo.

```
if [[ $u_flag -eq 1 ]];then #-u
    for (( i = 1; i < ${#info[@]}; i=$((i+10)) ));do
        if [[ "${info[i]}" == $name_user ]];then
            temp+=("${info[i-1]} ${info[i]} ${info[i+1]} ${info[i+2]} ${info[i+3]} ${info[i+4]}
                ${info[i+5]} ${info[i+6]} ${info[i+7]} ${info[i+8]})
        fi
    done
    info=("${temp[@]}")
    temp=()
fi
```

Figura 10 - Bloco da opção "-u"

- **Opções “-m -M” – Seleção dos processos através das gamas de PIDS:**

Assim como nas opções anteriores, uma vez que a primeira informação de PID está armazenada no índice 2 do *array info* e que um processo ocupa 10 índices, o armazenamento dos dados de **PIDS** também segue um padrão.

Em seguida, considerando que as opções -m e -M representam, respetivamente, o valor mínimo do filtro de **PIDS** e o valor máximo, foram implementados os blocos condicionais que englobassem todas as combinações possíveis. Ou seja:

1. **Opção “-m” – Seleção apenas por valor mínimo de PID:**

Considerando que a *flag* da opção “-m” está ativa e o da opção “-M” está inativa, foi suficiente testar se o valor do **PID** presente no índice atual do *loop* é maior ou igual do que o valor mínimo desejado, armazenado na variável “\$pids1”.

2. **Opção “-m” e “-M” – Seleção por intervalo:**

Uma vez que as *flags* das duas opções estão ativas, foi adicionado ao bloco condicional uma expressão que testa se o valor do **PID** presente no índice atual do *loop* está contido no intervalo entre o valor mínimo e o valor máximo desejado, armazenados nas variáveis “\$pids1” e “\$pids2”, respetivamente.

3. **Opção “-M” – Seleção apenas por valor máximo de PID:**

Como última possibilidade foi realizada a implementação do código para realizar a seleção apenas por valor máximo de **PID**, ou seja, selecionar através do *loop* os processos que tenham **PID** menor ou igual ao valor máximo passado como argumento, armazenado na variável “\$pids2”.

Portanto, no final do bloco condicional de cada possibilidade o *array* temporário é incrementado, dentro do *loop*, com todas as informações do processo selecionado pela condição, que ao sair do *loop* são repassadas ao *array* permanente.

Vale ressaltar que a escolha das opções não se altera independentemente da ordem de inserção. Ou seja, executar o comando “./rwstat.sh -m ‘1000’ -M ‘5000’ 10” tem o mesmo resultado de “./rwstat.sh -M ‘5000’ -m ‘1000’ 10”

```

if [[ $m_flag -eq 1 ]];then

    for (( i = 2; i < ${#info[@]}; i=$((i+10)) ));do

        if [[ $M_flag -eq 0 && ${info[i]} -ge $pids1 ]];then # -m

            temp+=(${info[i-2]} ${info[i-1]} ${info[i]} ${info[i+1]} ${info[i+2]} ${info[i+3]}
                ${info[i+4]} ${info[i+5]} ${info[i+6]} ${info[i+7]})

            elif [[ ${info[i]} -ge $pids1 && ${info[i]} -le $pids2 ]];then # -m -M

                temp+=(${info[i-2]} ${info[i-1]} ${info[i]} ${info[i+1]} ${info[i+2]} ${info[i+3]}
                    ${info[i+4]} ${info[i+5]} ${info[i+6]} ${info[i+7]})

            fi

        done

        info=("${temp[@]}")
        temp=()

    elif [[ $M_flag -eq 1 ]];then

        for (( i = 2; i < ${#info[@]}; i=$((i+10)) ));do

            if [[ ${info[i]} -le $pids2 ]];then # -M

                temp+=(${info[i-2]} ${info[i-1]} ${info[i]} ${info[i+1]} ${info[i+2]} ${info[i+3]}
                    ${info[i+4]} ${info[i+5]} ${info[i+6]} ${info[i+7]})

            fi

        done

        info=("${temp[@]}")
        temp=()

    fi

```

Figura 11 - Bloco das opções "-m -M"

- **Opções “-s -e” – Seleção dos processos através de datas-limite:**

Uma vez que as informações sobre as datas no *array info* se encontram nas posições 7, 8 e 9, representando o mês, o dia e a hora respetivamente, é criado um *loop for* que começa no índice 7 e enquanto esta variável for superior ao tamanho do *info*, vai incrementar 10, pois cada *PID* ocupa 10 índices do mesmo.

1. **Opção “-s” – Seleção apenas por data mínima:**

Considerando que a *flag* “-s” está ativa e “-e” inativa, basta comparar a data introduzida pelo utilizador com as variáveis **dataatual** e **horaatual** definidas anteriormente. Neste caso, vai verificar que estas variáveis vêm depois de **datamin** e **horamin**, definidas com os dados do utilizador. Caso se verifique, adiciona ao *array* temporário.

2. **Opção “-e” – Seleção apenas por data máxima:**

Considerando que a *flag* “-e” está ativa e “-s” inativa, basta comparar a data introduzida pelo utilizador com as variáveis **dataatual** e **horaatual**, caso, se verifique que estas variáveis vêm antes de **datamax** e **horamax**, definidas com os dados do utilizador. Caso se verifique, adiciona ao *array* temporário.

3. **Opção “-s” e “-e” – Seleção através das duas condições:**

Por sua vez, caso ambas as opções sejam utilizadas, vai verificar se as variáveis **dataatual** e **hora atual** estão entre as variáveis mínimas e máximas, **datamin/horamin** e **datamax/horamax**, respetivamente. Se ambas as condições se verificarem, todas as informações referentes a esse *PID* serão adicionadas ao *array info*.

```

if [[ $s_flag -eq 1 ]];then #-s

for (( i = 7; i < ${#info[@]}; i=$((i+10)) ));do
dataatual=${info[i]}" " "${info[i+1]}"
horaatual=$(date -u -d `echo "${info[i+2]}"` +"%s")
if [[ $e_flag -eq 0 ]];then #-s
    if [[ `date +"%Y/%m/%d" -d "${dataatual}"` > $datamin ]];then
        temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
        ${info[i]} ${info[i+1]} ${info[i+2]})
    elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == "$datamin" ]] && [[ $horaatual -ge $horamin ]];then
        temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
        ${info[i]} ${info[i+1]} ${info[i+2]})
    fi
else #-s -e
    if [[ `date +"%Y/%m/%d" -d "${dataatual}"` > $datamin ]] && [[ `date +"%Y/%m/%d" -d "${dataatual}"` <
    $datamax ]];then
        temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
        ${info[i]} ${info[i+1]} ${info[i+2]})
    elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == $datamin ]] && [[ $horaatual -ge $horamin ]];then
        if [[ `date +"%Y/%m/%d" -d "${dataatual}"` < $datamax ]];then
            temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
            ${info[i]} ${info[i+1]} ${info[i+2]})
        elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == $datamax ]] && [[ $horaatual -le $horamax ]];then
            temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
            ${info[i]} ${info[i+1]} ${info[i+2]})
        fi
    elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == $datamax ]] && [[ $horaatual -le $horamax ]];then
        if [[ `date +"%Y/%m/%d" -d "${dataatual}"` > $datamin ]];then
            temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
            ${info[i]} ${info[i+1]} ${info[i+2]})
        elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == "$datamin" ]] && [[ $horaatual -ge $horamin ]];then
            temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
            ${info[i]} ${info[i+1]} ${info[i+2]})
        fi
    fi
fi
done

info=("${temp[@}")
temp=()

elif [[ $e_flag -eq 1 ]];then #-e

for (( i = 7; i < ${#info[@]}; i=$((i+10)) ));do
dataatual=${info[i]}" " "${info[i+1]}"
horaatual=$(date -u -d `echo "${info[i+2]}"` +"%s")
if [[ `date +"%Y/%m/%d" -d "${dataatual}"` < $datamax ]];then
    temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
    ${info[i]} ${info[i+1]} ${info[i+2]})
elif [[ `date +"%Y/%m/%d" -d "${dataatual}"` == $datamax ]] && [[ $horaatual -le $horamax ]];then
    temp+=(${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]} ${info[i-2]} ${info[i-1]}
    ${info[i]} ${info[i+1]} ${info[i+2]})
fi
done

info=("${temp[@}")
temp=()
fi

```

Figura 12 - Bloco das opção "-s -e"



- **Opção “-w” – Ordenação de processos pelos seus valores escritos (write values):**

Este filtro, quando ativado, ordena os processos apresentados na tabela por ordem decrescente os valores de **WRITEB** de cada processo.

Inicialmente é declarado um *array* **write\_order** em que vão sendo adicionadas os **PID's** e correspondentes informações já por ordem decrescente. Igualamos o **temp** ao *array* que possui as informações que vão ser apresentadas na tabela, **info**, e é iniciado um *loop while* que irá estar em funcionamento enquanto **temp** não estiver vazio. Dentro deste *loop* é definida a variável **max**, inicialmente definida como o **WRITEB** do primeiro processo no *array*, e a variável **comp**, que vai indicar em que posição do *array temp* se encontra o processo com **WRITEB** maior (inicialmente 0, pela razão estabelecida anteriormente). O *for* tem como propósito percorrer os processos todos de **temp** e verificar qual processo possui maior **WRITEB** (é iniciado na posição 4 do *array* pois é aí que esta os dados escritos pelo primeiro processo). A cada iteração do *while* é adicionado o processo que possui maior **WRITEB** de **temp** ao **write\_order** e é removido do *array* temporário. Quando o *array temp* estiver vazio o *loop while* termina e o *array* que guarda as informações para serem depois apresentadas numa tabela, **info**, é igualado ao *array* que já possui os processos ordenados como pretendido, **write\_order**.

```
if [[ $w_flag -eq 1 ]];then #-w

declare -a write_order
temp=("${info[@]}") # para fazer alterações sem mudar o array principal
while [ ${#temp[@]} != 0 ];do
    max=${temp[4]} # primeiro
    comp=0
    for (( i=4; i <= ((${#temp[@]}-5)); i=$((i+10)) ));do
        if [[ $max -lt ${temp[$i]} ]];then
            max=${temp[$i]}
            comp=$((i-4)) # vai dar o valor onde esta o menor para depois ajudar na remoção
        fi
    done
    # write_order+=("${temp[@]:$comp:$((comp+10))}")
    write_order+=("${temp[comp]} ${temp[comp+1]} ${temp[comp+2]} ${temp[comp+3]} ${temp[comp+4]} ${temp[comp+5]}
        ${temp[comp+6]} ${temp[comp+7]} ${temp[comp+8]} ${info[comp+9]}")
    temp=("${temp[@]:0:$comp}" "${temp[@]:$((comp+10))}")
done
temp=()
info=("${write_order[@]}")
fi
```

Figura 13 - Bloco da opção “-w”

- **Opção “-r” – Ordenação por ordem inversa:**

Este filtro faz com que toda a informação que já foi selecionada até o momento, com o uso de outros filtros ou não, seja armazenada e mais tarde impressa na ordem inversa. Por isso, foi elaborado um *loop for* que tem como parâmetro inicial o último elemento de **info**, a variável tem de ser maior ou igual a zero e é decrementado de 10 em 10 unidades.

Já dentro do ciclo *for* o *array* temporário é incrementado com as informações contidas no índice atual, que, por ser decrementado de 9 até 0, resulta na mesma informação, mas armazenada na ordem inversa. Então, assim como nas outras opções, os dados do *array* temporário são atribuídos ao *array* permanente.

Por fim, é de importante ressaltar que a opção foi implementada após a definição de todas as outras opções, com exceção da opção de números de processos **-p**, considerando que a ordem inversa deve ser aplicada posteriormente à aplicação dos filtros de seleção.

```
if [[ $r_flag -eq 1 ]];then
    for (( i=$(( ${#info[@]}-1 )) ; i >= 0; i=$((i-10)) ));do
        temp+=("${info[i-9]} ${info[i-8]} ${info[i-7]} ${info[i-6]} ${info[i-5]} ${info[i-4]} ${info[i-3]}
        ${info[i-2]} ${info[i-1]} ${info[i]})
    done
    info=("${temp[@]}")
    temp=()
fi
```

Figura 14 - Bloco da opção “-r”

- **Opção “-p” – Seleção do número de processos**

Este filtro, permite ao utilizador escolher o número de processos que pretende ver na tabela apresentada. Neste *loop for*, como cada processo ocupa 10 posições do *array info*, vamos adicionar ao nosso *array* temporário (*temp*) “10\*\$num\_proc”, em que **num\_proc** representa o número introduzido de processos que o utilizador pretende que estejam presentes na tabela. Deste modo, o nosso *array* temporário terá apenas o número de processos pretendido. Por fim, atualizamos o nosso *array info* igualando-o a *temp* e esvaziamos.

```
if [[ $p_flag -eq 1 ]];then #-p
    for (( i = 0 ; i < 10*($num_proc); i++ ));do
        temp+=("${info[i]})"
    done
    info=("${temp[@]}")
    temp=()
fi
```

Figura 15 - Bloco da opção “-p”

## Testes realizados

- `./rwstat.sh 10`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh 10
```

| COMM            | USER   | PID    | READB    | WRITEB | RATER      | RATEW   | DATE        |
|-----------------|--------|--------|----------|--------|------------|---------|-------------|
| code            | mendes | 102297 | 0        | 0      | 0          | 0       | Dec 1 20:05 |
| firefox         | mendes | 12656  | 2857     | 11     | 285.70     | 1.10    | Dec 1 19:31 |
| Socket_Process  | mendes | 12808  | 0        | 0      | 0          | 0       | Dec 1 19:31 |
| Privileged_Cont | mendes | 12833  | 12       | 12     | 1.20       | 1.20    | Dec 1 19:31 |
| snap            | mendes | 12871  | 0        | 0      | 0          | 0       | Dec 1 19:31 |
| WebExtensions   | mendes | 13170  | 2        | 2      | .20        | .20     | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13483  | 2        | 2      | .20        | .20     | Dec 1 19:31 |
| deja-dup-monito | mendes | 13632  | 0        | 0      | 0          | 0       | Dec 1 19:32 |
| gjs             | mendes | 139145 | 184      | 296    | 18.40      | 29.60   | Dec 2 00:51 |
| rwstat.sh       | mendes | 139359 | 10926220 | 26687  | 1092622.00 | 2668.70 | Dec 2 00:57 |
| systemd         | mendes | 1784   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| pipewire        | mendes | 1791   | 8        | 8      | .80        | .80     | Dec 1 19:30 |
| pipewire-media- | mendes | 1792   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| pulseaudio      | mendes | 1793   | 104      | 64     | 10.40      | 6.40    | Dec 1 19:30 |
| dbus-daemon     | mendes | 1804   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gnome-keyring-d | mendes | 1806   | 24       | 48     | 2.40       | 4.80    | Dec 1 19:30 |
| gvfsd           | mendes | 1816   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfsd-fuse      | mendes | 1827   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| xdg-document-po | mendes | 1840   | 24       | 40     | 2.40       | 4.00    | Dec 1 19:30 |
| xdg-permission- | mendes | 1844   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| tracker-miner-f | mendes | 1865   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfs-udisks2-vo | mendes | 1877   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfs-mtp-volume | mendes | 1882   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfs-gphoto2-vo | mendes | 1886   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfs-afc-volume | mendes | 1890   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfs-goa-volume | mendes | 1895   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| goa-daemon      | mendes | 1899   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| goa-identity-se | mendes | 1906   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gdm-x-session   | mendes | 1956   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gnome-session-b | mendes | 2073   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| at-spi-bus-laun | mendes | 2155   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| dbus-daemon     | mendes | 2161   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gnome-session-c | mendes | 2168   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gnome-session-b | mendes | 2180   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gnome-shell     | mendes | 2199   | 224      | 4580   | 22.40      | 458.00  | Dec 1 19:30 |
| gnome-shell-cal | mendes | 2253   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| evolution-sourc | mendes | 2261   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| dconf-service   | mendes | 2268   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| evolution-calen | mendes | 2269   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| evolution-addre | mendes | 2280   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gvfsd-trash     | mendes | 2296   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| xdg-desktop-por | mendes | 2305   | 24       | 48     | 2.40       | 4.80    | Dec 1 19:30 |
| xdg-desktop-por | mendes | 2309   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gjs             | mendes | 2314   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| at-spi2-registr | mendes | 2319   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| sh              | mendes | 2337   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gsd-a11y-settin | mendes | 2338   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| ibus-daemon     | mendes | 2340   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gsd-color       | mendes | 2341   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gsd-datetime    | mendes | 2343   | 0        | 0      | 0          | 0       | Dec 1 19:30 |
| gsd-housekeepin | mendes | 2346   | 14540    | 0      | 1454.00    | 0       | Dec 1 19:30 |
| gsd-keyboard    | mendes | 2348   | 0        | 0      | 0          | 0       | Dec 1 19:30 |

Figura 16 – Sem filtros adicionados

- `./rwstat.sh -c '^gsd.*' 5`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -c '^gsd.*' 5
```

| COMM             | USER   | PID  | READB | WRITEB | RATER | RATEW | DATE        |
|------------------|--------|------|-------|--------|-------|-------|-------------|
| gsd-a11y-settin  | mendes | 8432 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-color        | mendes | 8435 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-datetime     | mendes | 8438 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-housekeepin  | mendes | 8439 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-keyboard     | mendes | 8442 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-media-keys   | mendes | 8445 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-power        | mendes | 8448 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-print-notif  | mendes | 8451 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-rfkill       | mendes | 8452 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-screensaver  | mendes | 8453 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-sharing      | mendes | 8454 | 24    | 40     | 4.80  | 8.00  | Dez 1 19:03 |
| gsd-smartcard    | mendes | 8456 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-sound        | mendes | 8458 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-wacom        | mendes | 8459 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-xsettings    | mendes | 8464 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-disk-utillit | mendes | 8483 | 0     | 0      | 0     | 0     | Dez 1 19:03 |
| gsd-printer      | mendes | 8538 | 0     | 0      | 0     | 0     | Dez 1 19:03 |

Figura 17 – filtro “-c”

- `./rwstat.sh -u 'mendes' -c '^p.*' 8`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -u 'mendes' -c '^p.*' 8
```

| COMM            | USER   | PID  | READB | WRITEB | RATER  | RATEW  | DATE        |
|-----------------|--------|------|-------|--------|--------|--------|-------------|
| pipewire        | mendes | 7855 | 0     | 0      | 0      | 0      | Dez 1 19:03 |
| pipewire-media- | mendes | 7856 | 0     | 0      | 0      | 0      | Dez 1 19:03 |
| pulseaudio      | mendes | 7857 | 972   | 972    | 121.50 | 121.50 | Dez 1 19:03 |

Figura 18 – filtro “-u” (“-c” utilizado para aparecerem menos resultados)

- `./rwstat.sh -w 9`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -w 9
```

| COMM            | USER   | PID    | READB   | WRITEB | RATER     | RATEW   | DATE        |
|-----------------|--------|--------|---------|--------|-----------|---------|-------------|
| rwstat.sh       | mendes | 105533 | 6343506 | 25312  | 704834.00 | 2812.44 | Dec 1 20:12 |
| gnome-terminal- | mendes | 3819   | 0       | 9860   | 0         | 1095.55 | Dec 1 19:30 |
| gnome-shell     | mendes | 2199   | 264     | 4344   | 29.33     | 482.66  | Dec 1 19:30 |
| code            | mendes | 4064   | 7161    | 635    | 795.66    | 70.55   | Dec 1 19:30 |
| gjs             | mendes | 2805   | 184     | 296    | 20.44     | 32.88   | Dec 1 19:30 |
| nautilus        | mendes | 68971  | 0       | 224    | 0         | 24.88   | Dec 1 19:44 |
| gsd-media-keys  | mendes | 2354   | 40      | 104    | 4.44      | 11.55   | Dec 1 19:30 |
| code            | mendes | 4081   | 0       | 93     | 0         | 10.33   | Dec 1 19:31 |
| gsd-sharing     | mendes | 2365   | 48      | 80     | 5.33      | 8.88    | Dec 1 19:30 |
| pulseaudio      | mendes | 1793   | 104     | 64     | 11.55     | 7.11    | Dec 1 19:30 |
| code            | mendes | 3974   | 58      | 58     | 6.44      | 6.44    | Dec 1 19:30 |
| gnome-keyring-d | mendes | 1806   | 24      | 48     | 2.66      | 5.33    | Dec 1 19:30 |
| xdg-desktop-por | mendes | 2305   | 24      | 48     | 2.66      | 5.33    | Dec 1 19:30 |
| xdg-document-po | mendes | 1840   | 24      | 40     | 2.66      | 4.44    | Dec 1 19:30 |
| ibus-portal     | mendes | 2439   | 24      | 40     | 2.66      | 4.44    | Dec 1 19:30 |
| code            | mendes | 4049   | 24      | 24     | 2.66      | 2.66    | Dec 1 19:30 |
| firefox         | mendes | 12656  | 2860    | 14     | 317.77    | 1.55    | Dec 1 20:12 |
| pipewire        | mendes | 1791   | 8       | 8      | .88       | .88     | Dec 1 19:32 |
| Isolated_Web_Co | mendes | 41568  | 4       | 4      | .44       | .44     | Dec 1 19:30 |
| Privileged_Cont | mendes | 12833  | 2       | 2      | .22       | .22     | Dec 1 19:31 |

Figura 19 – Filtro “-w”

- `./rwstat.sh -w -p 6 10`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -w -p 6 10
```

| COMM            | USER   | PID    | READB   | WRITEB | RATER     | RATEW   | DATE        |
|-----------------|--------|--------|---------|--------|-----------|---------|-------------|
| rwstat.sh       | mendes | 102321 | 6350674 | 25284  | 635067.40 | 2528.40 | Dec 1 20:06 |
| gnome-terminal- | mendes | 3819   | 0       | 10840  | 0         | 1084.00 | Dec 1 19:30 |
| gnome-shell     | mendes | 2199   | 168     | 4160   | 16.80     | 416.00  | Dec 1 19:30 |
| code            | mendes | 4064   | 7174    | 645    | 717.40    | 64.50   | Dec 1 19:30 |
| code            | mendes | 3974   | 87      | 87     | 8.70      | 8.70    | Dec 1 19:30 |
| code            | mendes | 4081   | 0       | 62     | 0         | 6.20    | Dec 1 19:30 |

Figura 22 – Filtro “-p” (com ajuda de “-w” para mostrar o seu funcionamento)

- `./rwstat.sh -m 10000 4`

```
mendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -m 10000 4
```

| COMM            | USER   | PID   | READB    | WRITEB | RATER       | RATEW   | DATE        |
|-----------------|--------|-------|----------|--------|-------------|---------|-------------|
| firefox         | mendes | 12656 | 1427     | 8      | 356.75      | 2.00    | Dec 1 19:31 |
| Socket_Process  | mendes | 12808 | 0        | 0      | 0           | 0       | Dec 1 19:31 |
| Privileged_Cont | mendes | 12833 | 0        | 0      | 0           | 0       | Dec 1 19:31 |
| snap            | mendes | 12871 | 0        | 0      | 0           | 0       | Dec 1 19:31 |
| WebExtensions   | mendes | 13170 | 0        | 0      | 0           | 0       | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13483 | 11       | 11     | 2.75        | 2.75    | Dec 1 19:31 |
| deja-dup-monito | mendes | 13632 | 0        | 0      | 0           | 0       | Dec 1 19:32 |
| deja-dup        | mendes | 23937 | 0        | 0      | 0           | 0       | Dec 1 19:34 |
| code            | mendes | 27418 | 0        | 0      | 0           | 0       | Dec 1 19:34 |
| Web_Content     | mendes | 41520 | 0        | 0      | 0           | 0       | Dec 1 19:44 |
| Web_Content     | mendes | 41565 | 0        | 0      | 0           | 0       | Dec 1 19:44 |
| Web_Content     | mendes | 41568 | 0        | 0      | 0           | 0       | Dec 1 19:44 |
| RDD_Process     | mendes | 41616 | 0        | 0      | 0           | 0       | Dec 1 19:44 |
| Utility_Process | mendes | 41618 | 0        | 0      | 0           | 0       | Dec 1 19:44 |
| nautilus        | mendes | 68971 | 0        | 0      | 0           | 0       | Dec 1 19:52 |
| rwstat.sh       | mendes | 75030 | 59169394 | 38762  | 14792348.50 | 9690.50 | Dec 1 19:54 |

Figura 23 – Filtro “-m”



- `./rwstat -M 2000 10`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -M 2000 10
```

| COMM            | USER   | PID  | READB | WRITEB | RATER | RATEW | DATE        |
|-----------------|--------|------|-------|--------|-------|-------|-------------|
| systemd         | mendes | 1784 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| pipewire        | mendes | 1791 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| pipewire-media- | mendes | 1792 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| pulseaudio      | mendes | 1793 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| dbus-daemon     | mendes | 1804 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gnome-keyring-d | mendes | 1806 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfsd           | mendes | 1816 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfsd-fuse      | mendes | 1827 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| xdg-document-po | mendes | 1840 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| xdg-permission- | mendes | 1844 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| tracker-miner-f | mendes | 1865 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfs-udisks2-vo | mendes | 1877 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfs-mtp-volume | mendes | 1882 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfs-gphoto2-vo | mendes | 1886 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfs-afc-volume | mendes | 1890 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gvfs-goa-volume | mendes | 1895 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| goa-daemon      | mendes | 1899 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| goa-identity-se | mendes | 1906 | 0     | 0      | 0     | 0     | Dec 1 19:30 |
| gdm-x-session   | mendes | 1956 | 0     | 0      | 0     | 0     | Dec 1 19:30 |

Figura 24 – Filtro “-w”

- `./rwstat.sh -m 10000 -M 20000 10`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -m 10000 -M 20000 10
```

| COMM            | USER   | PID   | READB | WRITEB | RATER | RATEW | DATE        |
|-----------------|--------|-------|-------|--------|-------|-------|-------------|
| firefox         | mendes | 12656 | 9     | 9      | .90   | .90   | Dec 1 19:31 |
| Socket_Process  | mendes | 12808 | 0     | 0      | 0     | 0     | Dec 1 19:31 |
| Privileged_Cont | mendes | 12833 | 2     | 2      | .20   | .20   | Dec 1 19:31 |
| snap            | mendes | 12871 | 0     | 0      | 0     | 0     | Dec 1 19:31 |
| WebExtensions   | mendes | 13170 | 14    | 14     | 1.40  | 1.40  | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13483 | 4     | 4      | .40   | .40   | Dec 1 19:31 |
| deja-dup-monito | mendes | 13632 | 0     | 0      | 0     | 0     | Dec 1 19:32 |

Figura 25 – Filtro “-m” e “-M”

- `./rwstat.sh -s 'Dec 1 19:31' 2`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -s "Dec 1 19:31" 2
```

| COMM            | USER   | PID   | READB    | WRITEB | RATER       | RATEW    | DATE        |
|-----------------|--------|-------|----------|--------|-------------|----------|-------------|
| firefox         | mendes | 12656 | 1431     | 12     | 715.50      | 6.00     | Dec 1 19:31 |
| Socket_Process  | mendes | 12808 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Privileged_Cont | mendes | 12833 | 2        | 2      | 1.00        | 1.00     | Dec 1 19:31 |
| snap            | mendes | 12871 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| WebExtensions   | mendes | 13170 | 2        | 2      | 1.00        | 1.00     | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13182 | 2        | 2      | 1.00        | 1.00     | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13405 | 14       | 14     | 7.00        | 7.00     | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13409 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13423 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13483 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13526 | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Isolated_Web_Co | mendes | 13610 | 134      | 134    | 67.00       | 67.00    | Dec 1 19:32 |
| deja-dup-monito | mendes | 13632 | 0        | 0      | 0           | 0        | Dec 1 19:32 |
| deja-dup        | mendes | 23937 | 0        | 0      | 0           | 0        | Dec 1 19:34 |
| code            | mendes | 27418 | 0        | 0      | 0           | 0        | Dec 1 19:34 |
| chrome_crashpad | mendes | 3908  | 0        | 0      | 0           | 0        | Dec 1 19:31 |
| Web_Content     | mendes | 41520 | 0        | 0      | 0           | 0        | Dec 1 19:44 |
| Web_Content     | mendes | 41565 | 0        | 0      | 0           | 0        | Dec 1 19:44 |
| Web_Content     | mendes | 41568 | 0        | 0      | 0           | 0        | Dec 1 19:44 |
| RDD_Process     | mendes | 41616 | 0        | 0      | 0           | 0        | Dec 1 19:44 |
| Utility_Process | mendes | 41618 | 0        | 0      | 0           | 0        | Dec 1 19:44 |
| cpptools        | mendes | 4226  | 228      | 0      | 114.00      | 0        | Dec 1 19:31 |
| rwstat.sh       | mendes | 52157 | 62700835 | 40821  | 31350417.50 | 20410.50 | Dec 1 19:47 |
| update-notifier | mendes | 9657  | 0        | 0      | 0           | 0        | Dec 1 19:31 |

Figura 26 – Filtro “-s”

- `./rwstat.sh -e 'Dec 1 19:30' -c '^c.*' 2`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -e "Dec 1 19:30" -c '^c.*' 2
```

| COMM | USER   | PID  | READB | WRITEB | RATER   | RATEW  | DATE        |
|------|--------|------|-------|--------|---------|--------|-------------|
| code | mendes | 3859 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code | mendes | 3873 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code | mendes | 3874 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code | mendes | 3935 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code | mendes | 3974 | 29    | 29     | 14.50   | 14.50  | Dec 1 19:30 |
| code | mendes | 4065 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code | mendes | 4049 | 8     | 8      | 4.00    | 4.00   | Dec 1 19:30 |
| code | mendes | 4064 | 2867  | 256    | 1433.50 | 128.00 | Dec 1 19:30 |
| code | mendes | 4081 | 0     | 31     | 0       | 15.50  | Dec 1 19:30 |

Figura 27 – Filtro “-e” (“-c” usado para diminuir o número de elementos na tabela)

- **`./rwstat.sh -s 'Dec 1 19:30' -e 'Dec 1 19:31' -c '^c.*' 2`**

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -s "Dec 1 19:30" -e "Dec 1 19:31" -c '^c.*' 2
```

| COMM            | USER   | PID  | READB | WRITEB | RATER   | RATEW  | DATE        |
|-----------------|--------|------|-------|--------|---------|--------|-------------|
| code            | mendes | 3859 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code            | mendes | 3873 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code            | mendes | 3874 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| chrome_crashpad | mendes | 3908 | 0     | 0      | 0       | 0      | Dec 1 19:31 |
| code            | mendes | 3935 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code            | mendes | 3974 | 29    | 29     | 14.50   | 14.50  | Dec 1 19:30 |
| code            | mendes | 4005 | 0     | 0      | 0       | 0      | Dec 1 19:30 |
| code            | mendes | 4049 | 8     | 8      | 4.00    | 4.00   | Dec 1 19:30 |
| code            | mendes | 4064 | 2867  | 254    | 1433.50 | 127.00 | Dec 1 19:30 |
| code            | mendes | 4081 | 0     | 31     | 0       | 15.50  | Dec 1 19:30 |
| cpptools        | mendes | 4226 | 225   | 0      | 112.50  | 0      | Dec 1 19:31 |

Figura 28 – Filtrros “-s” e “-e” (“-c” usado para obter menos posições)

- **`./rwstat.sh -w -r -c '^c.*' 7`**

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -w -r -c '^c.*' 7
```

| COMM            | USER   | PID    | READB | WRITEB | RATER  | RATEW | DATE        |
|-----------------|--------|--------|-------|--------|--------|-------|-------------|
| cpptools        | mendes | 4226   | 456   | 0      | 65.14  | 0     | Dec 1 20:05 |
| code            | mendes | 4005   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 3935   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| chrome_crashpad | mendes | 3908   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 3874   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 3873   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 3859   | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 102297 | 0     | 0      | 0      | 0     | Dec 1 20:05 |
| code            | mendes | 4049   | 16    | 16     | 2.28   | 2.28  | Dec 1 19:30 |
| code            | mendes | 3974   | 58    | 58     | 8.28   | 8.28  | Dec 1 19:30 |
| code            | mendes | 4081   | 0     | 62     | 0      | 8.85  | Dec 1 19:30 |
| code            | mendes | 4064   | 5734  | 514    | 819.14 | 73.42 | Dec 1 19:30 |

Figura 2920 – Filtro “-r” (com “-w” e “-c” para obter um resultado mais claro do seu funcionamento)

- **`./rwstat.sh -c '^g.*' -w -r -s "Nov 23 18:00" -e "Dec 2 19:30" -u mendes -m 2300 -M 4000 -p 3 10`**

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -c '^g.*' -w -r -s "Nov 23 18:00" -e "Dec 2 19:30" -u mendes -m 2300 -M 4000 -p 3 10
```

| COMM           | USER   | PID  | READB | WRITEB | RATER | RATEW | DATE        |
|----------------|--------|------|-------|--------|-------|-------|-------------|
| gvfsd-metadata | mendes | 3935 | 0     | 0      | 0     | 0     | Dec 2 19:27 |
| gjs            | mendes | 3666 | 0     | 0      | 0     | 0     | Dec 2 19:27 |
| gsd-printer    | mendes | 3540 | 0     | 0      | 0     | 0     | Dec 2 19:27 |

Figura 3021 – Teste de todos os filtros ao mesmo tempo

## Testes de Erro

- **./rwstat.sh**

```
nendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh
./rwstat.sh: line 70: ARRAY: bad array subscript
O último argumento deve ser um numero (tempo em segundos)

TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os itens aparecem na tabela (reverse);
-w : ordena os itens da tabela através dos valores 'escritos' por cada um (write values);
```

Figura 31 – Erro caso não seja introduzido o tempo

- **./rwstat.sh -c 10**

```
nendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -c 10
O argumento -c necessita de um parametro

TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os itens aparecem na tabela (reverse);
-w : ordena os itens da tabela através dos valores 'escritos' por cada um (write values);
```

Figura 32 – Erro caso não seja introduzido um valor a uma opção que necessite

- **./rwstat.sh -w**

```
nendes@mendesze:~/Desktop/50/Projeto_1$ ./rwstat.sh -w
O último argumento deve ser um numero (tempo em segundos)

TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os itens aparecem na tabela (reverse);
-w : ordena os itens da tabela através dos valores 'escritos' por cada um (write values);
```

Figura 33 – Erro caso seja introduzida uma opção, mas não o tempo

- `./rwstat.sh -M 10 6`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -M 10 6
Não existem processos ativos que cumpram esses filtros no espaço de tempo dado

TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROSTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os itens aparecem na tabela (reverse);
-w : ordena os itens da tabela através dos valores 'escritos' por cada um (write values);
```

Figura 34 – Erro caso não existam processos que cumpram os requisitos dos filtros introduzidos

- `./rwstat.sh -c '^gsd.*' -c '^d.*' 5`

```
mendes@mendesze:~/Desktop/S0/Projeto_1$ ./rwstat.sh -c '^gsd.*' -c '^d.*' 5
Foram escolhidas opções repetidas

TODAS AS OPÇÕES VÁLIDAS:
./rwstat.sh <FILTROSTEMPO>, em que os TEMPO é o número de segundos que serão usados para calcular as taxas de I/O,
logo o último argumento passado é obrigatoriamente em segundos.

FILTROS são por sua vez filtros de procura:
FILTROS DE PROCURA:
-c <OPTION> : filtra processos baseados no nome do comando através de uma expressão regular (regex);
-u <OPTION> : filtra processos pelo nome do utilizador;
-p <OPTION> : o número de processos que vão aparecer na tabela;
-m <OPTION> : filtro em que OPTION é o número mínimo de PID;
-M <OPTION> : filtro em que OPTION é o número máximo de PID;
FILTROS DE DATAS:
-s <DATE> : filtro em que DATE é a data mínima do início do processo;
-e <DATE> : filtro em que DATE é a data máxima do início do processo;
FILTROS DE ORDEM:
-r : inverte as posições em que os itens aparecem na tabela (reverse);
-w : ordena os itens da tabela através dos valores 'escritos' por cada um (write values);
```

Figura 35 – Erro caso seja introduzida a mesma opção repetidamente



## Conclusão

Todas as opções foram implementadas com êxito, ou seja, todos os resultados estão em conformidade com aquilo que é pedido no enunciado deste trabalho prático, e apresentados nos exemplos acima.

Este trabalho foi realizado com base em conhecimentos sobre a linguagem Bash adquiridos maioritariamente nas aulas teóricas e práticas. Com o decorrer do trabalho surgiram alguns desafios em relação a certos comandos essenciais para a realização deste trabalho. Estes foram ultrapassados através de muita pesquisa e discussão entre os participantes do trabalho.

Em suma, este trabalho teve um impacto bastante positivo no nosso conhecimento sobre a linguagem Bash, aprimorando as nossas habilidades de programação através da implementação de novas metodologias de trabalho e pesquisa.

## Bibliografia

<https://stackoverflow.com/>

<https://www.geeksforgeeks.org/bash-scripting-introduction-to-bash-and-bash-scripting/>

<https://www.gnu.org/software/bash/manual/bash.html#Lists>