universidade de aveiro deti departamento de electrónica, telecomunicações e informática

Trabalho Prático 1

Taxas de Leitura e Escrita de processos em Bash

Relatório – Turma P3

Diogo Falcão Nº 108712

José Gameiro Nº108840

Índice

1. Introdução 3

2. Desenvolvimento 4

2.1 Inicialização de variáveis e estruturas de dados a utilizar 4

2.2 Verificação e validação do argumento obrigatório 5

2.3 Obtenção de informações relativas aos processos 5

2.4 Argumentos opcionais 7

2.5 Impressão de Dados 10

2.6 Testes 11

3. Conclusão 16

1. Introdução

Neste trabalho, foi-nos proposto o desenvolvimento de um script em bash de modo a aplicar os conteúdos lecionados nas aulas práticas e teórico-práticas. O objetivo deste é obter e imprimir estatísticas sobre leitura e escrita de processos de acordo com os argumentos usados na linha de comandos através de uma tabela.

Com isto é possível visualizar o número total de bytes de I/O que um processo leu ou escreveu e também a taxa de leitura ou escrita correspondente ao número de segundos introduzidos pelo utilizador com um parâmetro obrigatório.

Para a impressão dos dados numa tabela existem várias opções que o utilizador pode usar para puder filtrar os dados da tabela, como por exemplo, poder escolher quais os processos que pretende visualizar, ordenar os dados da tabela, entre outros.

Ao longo deste relatório iremos explicar, com detalhe, o processo para a obtenção dos dados relativos aos processos de leitura e escrita, a forma de imprimir os resultados numa tabela e as opções que o utilizador tem ao seu dispor para puder filtrar os processos que deseja visualizar na tabela.

2. Desenvolvimento

Nesta parte do relatório, descrevemos qual foi a nossa abordagem para resolver o problema proposto, bem como a explicação dos métodos usados para encontrar uma solução para os diversos problemas que surgiram ao longo da construção deste trabalho, a evolução do código e os testes efetuados para validar a nossa solução

O nosso script está dividido essencialmente em 6 partes:

- ✓ Inicialização de variáveis e estruturas de dados a utilizar;
- ✓ Verificação e validação do argumento obrigatório;
- ✓ Obtenção de informações relativas aos processos;
- ✓ Argumentos opcionais;
- ✓ Impressão de dados;
- ✓ Teste dos dados.

2.1 Inicialização de variáveis e estruturas de dados a utilizar

A partir do comando "declare -a", inicializámos arrays com os tipos de dados correspondentes de cada processo:

- "dates_seconds" - que irá guardar as datas de início dos processos em segundos;
- "process_info" - que junta todas estas informações num só array de modo a simplificar futuras operações no script;
- "information" - vai ajudar a fazer a impressão dos dados numa tabela consoante as opções escolhidas;
- "rchar_array" – que irá guardar os valores de rchar antes de ser executada a função sleep;
- "wchar_array" – que irá guardar os valores de wchar antes de ser executada a função sleep.

No que se refere a variáveis globais, inicializámos: numProcesses (o número de processos para a opção "-p", que por defeito é "null"), "write_values" para sortear os valores de wchar (por defeito esta variável encontra-se com o valor de 0), "reverse" (por defeito é definido com o valor 1, para que a ordenação seja feita por ordem inversa da taxa de leitura).

Como referido anteriormente, existe um parâmetro obrigatório, o número de segundos (a variável "seconds"), que tem de ser colocado como último argumento ao executar o programa.

2.2 Verificação e validação do argumento obrigatório

Como já referido anteriormente, o argumento "seconds" trata-se de uma opção obrigatória ao programa que recalcula as estatísticas de leituras e escritas de processos. Dado a importância, este argumento necessita de uma intensiva verificação de modo que o programa corra em consonância.

Ao observar a figura 2 podemos observar que existe uma primeira validação em que, é verificado se o utilizador inseriu pelo menos o tempo de execução. "$#" é uma variável da bash, pré-definida, que guarda o número de argumentos que são introduzidos ao executar o programa, ao verificar se o valor desta variável é 0 estamos a verificar se o utilizador inseriu o tempo de execução. Se este não inseriu, irá aparecer a mensagem presente no comando "echo", se não avança para a próxima condição.

Na condição seguinte, verifica se o argumento introduzido é um número inteiro positivo através da expressão regular presente na figura 2. Se esta condição for verdadeira irá aparecer no terminal a mensagem de erro presente no comando "echo", se não continua na execução do código.

2.3 Obtenção de informações relativas aos processos

Em primeiro lugar iremos obter o número total de bytes de I/O e de taxa de leitura ou escrita, ou seja, os valores de rchar e wchar, antes de o comando sleep $seconds ser executado para depois com estes valores podermos calcular os valores de rater e ratew. Para isto utilizámos um ciclo for em que este percorre todos os processos e com a condição decisão na linha 41 irá filtrar os processos para aqueles em que temos permissão.

O comando sleep $seconds irá fazer com que o programa espere que passe o número de segundos presente na variável $seconds. Depois, para a obtenção das informações dos processos, implementámos um ciclo for que percorre todos os processos existentes, através do seu PID. Dentro deste ciclo, incluímos três condições de decisão (if statements):

- a primeira para verificar se o processo existe;
- a segunda para verificar se temos permissão para aceder ao processo;
- a terceira e última, para verificar se existe as informações de rchar e wchar.

```
38
39  #----------------- Indexação dos dados em arrays  ----------------- #
40  for entry in $(ps -eo pid | tail -n +2); do
41      if [[ -r "/proc/$entry/io" ]] ; then
42          pid_new=$entry     # Obtenção do PID
43          rchar=$(cat /proc/$entry/io | grep rchar | cut -d " " -f 2)  # Obtenção do valor de rchar inicial
44          wchar=$(cat /proc/$entry/io | grep wchar | cut -d " " -f 2)  # Obtenção do valor de wchar inicial
45
46          rchar_array[$pid_new]=$rchar
47          wchar_array[$pid_new]=$wchar
48      fi
49  done
50
54  #----------------- Indexação dos dados em arrays  ----------------- #
55
56  for pid in $(ps -eo pid | tail -n +2); do   # Percorre todos os processos
57      # verifica se o processo existe
58      if ps -p $pid > /dev/null; then
59          # Verifica se temos permissão para aceder ao processo
60          if [[ -r "/proc/$pid/io" ]] ; then
61              # Verifica se existem as informações rchar e wchar
62              if $(cat /proc/$pid/io | grep -q 'rchar\|wchar'); then
63
64                  # command
65                  comm=$(ps -p $pid -o comm | tail -n +2)
66
67                  # user
68                  user=$(ps -p $pid -o user | tail -n +2)
69
70                  # pid
71                  processID=$pid
72
73                  # readb
74                  readb=$(cat /proc/$pid/io | grep rchar | cut -d " " -f 2)
75
76                  # writeb
77                  writeb=$(cat /proc/$pid/io | grep wchar | cut -d " " -f 2)
78
79                  # rater
80                  var1=$(cat /proc/$pid/io | grep 'rchar' | tr -dc '0-9')
81                  rchar_new=${var1//[!0-9]/}
82                  sub_rchar=$(($rchar_new-${rchar_array[$pid]}))
83                  rater=$( echo "scale=2;$sub_rchar/$seconds" | bc -l)
84
85                  # ratew
86                  var2=$(cat /proc/$pid/io | grep 'wchar')
87                  wchar_new=${var2//[!0-9]/}
88                  sub_wchar=$(($wchar_new - ${wchar_array[$pid]}))
89                  ratew=$( echo "scale=2;$sub_wchar/$seconds" | bc -l)
90
91                  # date
92                  sdate=$(ps -p $pid -o lstart | tail -n +2)
93                  dates_seconds[$count]=$(date -d"$sdate" +%s) # Guarda a data em segundos
94                  date=$(date -d "$(ps -p $pid -o lstart | tail -1 | awk '{print $1, $2, $3, $4}')" +"%b %d %H:%M" )
95
96                  # Adicionar a um array as informações todas
97                  process_info[$count]="$comm $user $processID $readb $writeb $rater $ratew $date"
98                  count=$(($count+1))
99              fi
100         fi
101     fi
102 done
```

Fig.1 – Ciclo for para a indexação dos dados

Se para cada processo, estas três condições forem cumpridas, o programa adiciona ao array process_info os dados guardados nas variáveis “comm”, “user”, “processID”, “readb”, “writeb”, “rater”, “ratew” e “date”, como podemos observar na imagem abaixo.

A informação "comm" refere-se ao nome do processo, "user" ao utilizador que se encontre registado no computador e "processID" ao ID do processo. As informações do nome do processo e do utilizador obtém-se através das linhas 52 e 53 através de um comando semelhante presente na figura 3 nas linhas 52 e 55. Como o ciclo for utiliza o PID do processo para percorrer todos os processos existentes logo o comando a executar para obter o PID encontra-se na linha 43.

As informações "readb" e "writeb" referem-se ao número total de bytes lidos e escritos, respetivamente. Para as obter utilizámos os comandos nas linhas 61 e 64 e guardámo-las nas variáveis rchar e wchar, respetivamente.

As informações de "rater" e "ratew" referem-se aos valores de taxa de leitura e escrita de processos, respetivamente. Para os obter é necessário subtrair os valores que se encontram nos arrays rchar_array e wchar_array aos valores de rchar_new e wchar_new, ou seja, os valores depois de executado o comando sleep $seconds. Por fim divide-se o resultado da subtração pelo tempo guardado na variável $seconds e obtém-se assim os valores de rater e ratew onde ficaram guardados nas variáveis rater e ratew.

A data de início do processo é obtida através da linha de comando na linha 92 e é guardada na variável $sDate, que depois irá ser formatada para o formato de mês dia horas:minutos e guardada novamente noutra variável que é a $date. Também é adicionado ao array dates_seconds um formato da variável $sDate em segundos para depois ser utilizado nas opções -s e -e.

Por fim os valores presentes nas variáveis $comm, $user, $processID, $readb, $writeb, $rater, $ratew e $date são guardados todos no array process_info pela ordem indicada.

2.4 Argumentos opcionais

Para integrar os argumentos opcionais na execução do ficheiro rwstat.sh utilizámos um switch. Existem nove opções e uma opção default. Cada opção está essencialmente dividida em 2 partes: verificação do comando e remoção de toda a informação, que não esteja de acordo com a opção escolhida, dos processos. Para a verificação dos comandos, é usada uma variável, "validate_args" que conta o número de argumentos inseridos pelo utilizador para cada opção.

Para percorrer todas as opções utilizámos um ciclo while com a função getopts para que se pudesse percorrer todos os argumentos introduzidos pelo utilizador.

- Opção -c

Esta opção tem como objetivo receber uma expressão regular e comparar com os nomes dos comandos presentes no array process_info para depois puder filtrar e imprimir as informações completas dos processos que estejam de acordo com a expressão regular introduzida pelo utilizador.

A nossa solução para esta opção tem por base guardar o conteúdo da variável $OPTARG (uma variável pré-definida da bash, por causa da função getopts) na variável $comm_opt. Consequentemente verifica se o utilizador

inseriu algum argumento (neste caso, uma expressão regular) após a opção "-c". Se a condição não for verdadeira então percorre-se o array com a informação de todos os processos e retira-se aqueles cujo o nome do comando não se adequa à expressão regular introduzida (com o comando unset).

- Opção -u

Nesta opção é inserido o nome de um utilizador e, com base nesse nome, é efetuado uma filtração dos dados do array process_info tal que o nome do utilizador seja igual ao nome do utilizador dos processos.

A solução que encontrámos para este problema foi tal como na opção anterior guardar o valor que se encontra na variável $OPTARG noutra variável chamada $user_opt, depois fazer uma validação semelhante à da opção -c e se se não verificar nenhum problema é retirado do array process_info todos os processos que não tenham como utilizador o mesmo que o $user_opt.

- Opção -s

Esta opção tem o objetivo de dado uma data introduzida pelo utilizador, os processos presentes no array com todas as informações retirar aqueles cuja data de início seja menor do que a data introduzida pelo utilizador.

Semelhante às outras opções o valor de $OPTARG é validado e, se a validação estiver correta, é colocado na variável $min_date. É efetuada uma segunda validação, para verificar se a data introduzida é válida e se esta for válida são removidos os processos, presentes no array process_info, que têm uma data de início menor do que a data guardada em $min_date.

- Opção -e

Esta opção tem um objetivo semelhante ao da opção -s só que as datas dos processos presentes no array com todas as informações que forem maiores do que uma certa data introduzida pelo utilizador serão retiradas.

O algoritmo que encontrámos para esta opção foi semelhante ao da opção -s pois ambas envolvem a filtração dos dados do array process_info através de uma determinada data, porém neste caso iremos retirar do array process_info os processos mais as suas informações cujas datas forem maiores do que a data inserida pelo utilizador.

- Opção -m

Nesta opção o utilizador insere um determinado PID e depois irá se retirar os processos do array com todas as informações aqueles que tenham um PID menor do que o inserido.

Para resolvermos este problema, nós fizemos uma validação semelhante às restantes opções e depois na filtração do array, comparámos os PID's dos processos com o PID guardado na variável $minPID (a variável com o valor do $OPTARG) e retirámos aqueles cujos PID's fossem menores do o $minPID.

- Opção -M

Esta opção é bastante semelhante à opção -m, no entanto a diferença é que nesta opção são retirados do array com todas as informações os processos que tiverem um PID maior do que um PID introduzido pelo utilizador.

A resolução para este que encontrámos foi, também, muito parecido com a opção -m, em que guardámos o valor de $OPTARG na variável $maxPID, consequentemente, percorremos o array process_info e retirámos os processos que tinham maior PID do que o guardado em $maxPID.

- Opção -p

Nesta opção o utilizador insere o número de processo que pretende visualizar.

Encontrámos uma solução para este problema em que verificávamos se o utilizador tinha inserido um número, tal como implementamos para as outras opções e verificámos também se o que o utilizador inseriu era um número inteiro positivo. Depois para imprimir o número de processos presente na variável $numProcesses, criámos uma condição, na impressão dos dados, para que esta parasse de imprimir quando tivesse imprimido o número de valores desejados pelo utilizador.

Opção -r e -w

Estas opções são bastantes semelhante e é necessário apenas introduzir -r ou -w ou ambos. Elas servem para ordenar os dados do array com todas as informações dependendo da opção que escolherem. Por defeito os dados encontram-se ordenados por ordem decrescente dos valores de rater, no entanto se a opção escolhida for -w irá os dados por ordem decrescente dos valores de ratew.

Para este problema nós sorteamos os dados de forma a que os valores de rater ficassem por ordem decrescente, sendo esta é a opção default, através da função sort em que selecionámos a coluna que deveria de ordenar (que neste caso é a coluna 6) e por fim a adicionámos uma parte para ordenar array process_info de acordo com os dados de rater. Caso a opção -w seja introduzida como um argumento a forma de ordenar o array será semelhante à de -r, mudando apenas a coluna a ordenar que neste caso seria a coluna 7.

Opção *)

Esta opção não é uma que utilizador insere ao correr o programa, mas uma que avisa o utilizador caso este tenha colocado um argumento errado.

2.5 Impressão de Dados

Para a impressão de dados, averigua-se em primeiro lugar se o número de processos seja diferente de zero. Depois, se as variáveis globais não mantiverem os mesmos valores e foram alteradas no switch case, isto é, "write_values" tiver o valor de 1 e "reverse" tiver o valor de 0, então a informação é impressa em conformidade. Se estas variáveis globais não foram mudadas, então o programa vai imprimir todas as informações dos processos por ordem decrescente dos valores de "rater". Em ambas as situações, quando o número de processos impressos atingir o mesmo valor de "numProcesses", da opção "-p", faz break da impressão

```
270 #------------------------------- Impressão de dados --------------------------------#
271 if [[ $numProcesses != 0 ]] ; then
272     printf "%-40s %-20s %-10s %-20s %-10s %-15s %-15s %-10s \n" "COMM" "USER" "PID" "READB" "WRITEB" "RATER" "RATEW" "DATE"  # Impressão do cabeçalho
273     if [[ $write_values -eq 1 && $reverse -eq 0 ]] ; then
274         for ((i=0; i<=$count; i++)) ; do
275             information=(${process_info[i]})
276             # se o valor do comando for null, não imprime nada
277             if [[ ${information[0]} == "" ]] ; then
278                 continue
279             fi
280             printf "%-40s %-20s %-10s %-20s %-10s %-15s %-15s %3s %3s %5s \n" ${information[0]} ${information[1]} ${information[2]} ${information[3]} ${informatio
281             # Para a opção -p parar de imprimir quandp chegar ao número de processos inserido pelo utilizador
282             if [[ $(($i+1)) -eq $numProcesses ]] ; then
283                 break
284             fi
285         done
286     else
287         for ((i=0; i<=$count; i++)) ; do
288             information=${process_info[i]}
289             # se o valor do comando for null, não imprime nada
290             if [[ ${information[0]} == "" ]] ; then
291                 continue
292             fi
293             printf "%-40s %-20s %-10s %-20s %-10s %-15s %-15s %3s %3s %5s \n" ${information[0]} ${information[1]} ${information[2]} ${information[3]} ${informatio
294             # Para a opção -p parar de imprimir quandp chegar ao número de processos inserido pelo utilizador
295             if [[ $(($i+1)) -eq $numProcesses ]] ; then
296                 break
297             fi
298         done
299     fi
300 else
301     echo "AVISO: Nenhum processo válido encontrado"
302     exit 1
303 fi
304
305 exit 0
```

Fig.2 – Impressão de dados

2.6 Testes

Durante a realização do projeto, o programa foi alvo de testes básicos, como o argumento obrigatório segundos, até mais complexos, com todos os argumentos opcionais.

- Argumento obrigatório item de existir

Dado que uma das principais funções do programa é calcular, em "s" segundos, a taxa de leitura/escrita (em bytes por segundo) dos processos selecionados nesse intervalo de tempo, o programa dá erro ao não colocar este argumento, isto é, o número de argumentos do programa tem de ser obrigatoriamente igual ou superior a 2.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh
ERRO: é necessário introduzir pelo menos um argumento obrigatório que é o tempo de execução em segundos
```

Fig.3 – Teste 1

Argumento obrigatório tem de ser um número inteiro positivo.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh nove
ERRO: o último argumento tem de ser um número inteiro positivo

diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -3
ERRO: o último argumento tem de ser um número inteiro positivo
```

Fig.4 a) e b) – Teste 2

- Argumento opcional "–c"

É através da opção "-c" que se pode visualizar no terminal todos os processos que comecem com a expressão regular introduzida. No entanto, para a opção selecionar processos, é necessário introduzir a expressão regular. Assim, se o utilizador apenas colocar no terminal "-c" mais o número de segundos, o programa identifica o erro de que um argumento está em falta.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -c 5
ERRO: não se pode utilizar o argumento dos segundos para a opção -c
```

Fig.5 – Teste 3

- Argumento opcional "-u"

Ao introduzir "-u" no terminal, é possível filtrar processos cujo nome do utilizador comece com o argumento seguinte escrito no terminal. Mais uma vez, é necessária a introdução de uma parte do nome do utilizador. Assim:

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -u 5
ERRO: não se pode utilizar o argumento dos segundos para a opção -u
```

Fig.6 – Teste 4

- Argumento opcional "-s"

Este argumento tem como função selecionar uma data mínima. Igualmente às opções anteriores, é necessária a introdução de um argumento, data_mínima.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -s 5
ERRO: não se pode utilizar o argumento dos segundos para a opção -s
```

Fig.7 a) – Teste 5

Para além do utilizador ao usar esta opção ter de escrever o argumento da data_mínima, esta tem de ser uma data válida.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -s "Nov 16 27:00"  5
date: data inválida "Nov 16 27:00"
ERRO: a data mínima inserida não é válida
```

Fig.7 b) – Teste 6

- Argumento opcional "-e"

Da mesma forma do argumento "-s", esta opção seleciona todos os processos que tenham como data máxima o argumento a seguir da opção "-e", necessitando do argumento seguinte.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -e 5
ERRO: não se pode utilizar o argumento dos segundos para a opção -e
```

Fig.8 a) – Teste 7

Para além do utilizador ao usar esta opção ter de escrever o argumento da data_máxima, esta tem de ser uma data válida.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -e "Fev 30 20:00"  5
date: data inválida "Fev 30 20:00"
ERRO: A data máxima não é válida
```

Fig.8 b) – Teste 8

- Argumento opcional "-m"

Esta opção requer um argumento seguinte para a seleção de processos.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -m 2
ERRO: não se pode utilizar o argumento dos segundos para a opção -m
```

Fig.9 a) – Teste 9

O argumento seguinte tem de ser um número inteiro positivo.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -m -2 5
ERRO: o argumento da opção -m não é um número inteiro positivo
```

Fig.9 b) – Teste 10

- Argumento opcional "-M"

Esta opção requer um argumento seguinte para a seleção de processos.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -M 2
ERRO: não se pode utilizar o argumento dos segundos para a opção -M
```

Fig.10 a) – Teste 11

O argumento seguinte tem de ser um número inteiro positivo.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -M 4.5 5
ERRO: o argumento da opção -m não é um número inteiro positivo
```

Fig.10 b) – Teste 12

- Argumento opcional "-p"

Finalmente, a opção "-p" controla o número de processos a imprimir no terminal. Mais uma vez, para o uso desta opção é necessário escrever um argumento que represente o número de processos.

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -p 5
ERRO: não se pode utilizar o argumento dos segundos para a opção -p
```

Fig.11 a) – Teste 13

Mais uma vez, verifica-se se o argumento introduzido é um número inteiro positivo

```
diogofalcao@diogofalcao-Surface-Laptop-4:~/Desktop/SO/SO_A01$ ./rwstat.sh -p -22 5
ERRO: o número de processos tem de ser maior que 0
```

Fig.11 b) – Teste 14

- Argumento "-s" e "-e"

Ao utilizar estas opções, verificamos na imagem abaixo, que durante 10 segundos, os processos apresentam uma data maior ou igual à data mínima inserida e menor ou igual à data máxima inserida.

```
jose@jose-VivoBook-ASUSLaptop-X580GD-N580GD:~/LEI/2ºano/1ºSemestre/SO/Práticas/Projeto_git/SO_A01$ ./rwstat.sh -s "Dec 02 23:00" -e "Dec 02 23:13" 10
COMM                USER            PID     READB           WRITEB      RATER       RATEW       DATE
gnome-shell         jose            1905    129664107       25620891    50738.60    721.20      dez  02 23:12
pulseaudio          jose            1503    18426291        19452992    15289.10    16126.00    dez  02 23:12
Xorg                jose            1514    14392749        126983728   2503.30     42706.10    dez  02 23:12
chrome              jose            3753    478707590       259313750   506.30      1285.60     dez  02 23:12
chrome              jose            3798    60576290        53489723    393.20      42750.10    dez  02 23:12
goa-identity-se     jose            1572    187272          49297       72.00       38.40       dez  02 23:12
gsd-sharing         jose            2089    56413           28203       12.00       20.00       dez  02 23:12
xdg-permission-     jose            1969    20895           939         0           0           dez  02 23:12
xdg-document-po     jose            2232    43463           13778       0           0           dez  02 23:12
xdg-desktop-por     jose            2859    1942776         26542       0           0           dez  02 23:12
xdg-desktop-por     jose            2855    61375           22122       0           0           dez  02 23:12
tracker-miner-f     jose            1505    2108082         6720        0           0           dez  02 23:12
systemd             jose            1496    10996773        19940186    0           0           dez  02 23:12
snap-store          jose            2192    2633441         170104      0           0           dez  02 23:12
nacl_helper         jose            3771    10209           120         0           0           dez  02 23:12
ibus-x11            jose            1941    292742          6540        0           0           dez  02 23:12
ibus-portal         jose            1943    46771           16649       0           0           dez  02 23:12
ibus-memconf        jose            1938    37347           888         0           0           dez  02 23:12
ibus-extension-     jose            1939    901158          47509       0           0           dez  02 23:12
ibus-engine-sim     jose            2204    73495           51440       0           0           dez  02 23:12
ibus-daemon         jose            1934    206586          220190      0           0           dez  02 23:12
gvfs-udisks2-vo     jose            1546    4007494         4467        0           0           dez  02 23:12
gvfs-mtp-volume     jose            1557    35531           1958        0           0           dez  02 23:12
gvfs-gphoto2-vo     jose            1552    39763           1986        0           0           dez  02 23:12
gvfs-goa-volume     jose            1561    21727           2254        0           0           dez  02 23:12
gvfsd-trash         jose            2062    153301          4304        0           0           dez  02 23:12
gvfsd               jose            1523    198745          8953        0           0           dez  02 23:12
gvfsd-fuse          jose            1528    47798           1176        0           0           dez  02 23:12
gvfs-afc-volume     jose            1577    31671           1966        0           0           dez  02 23:12
gsd-xsettings       jose            2104    477107          11865       0           0           dez  02 23:12
gsd-wwan            jose            2101    43429           3576        0           0           dez  02 23:12
gsd-wacom           jose            2099    446363          8192        0           0           dez  02 23:12
gsd-usb-protect     jose            2096    41221           4106        0           0           dez  02 23:12
gsd-sound           jose            2093    57565           3569        0           0           dez  02 23:12
gsd-smartcard       jose            2091    50124           3493        0           0           dez  02 23:12
gsd-screensaver     jose            2087    22895           2980        0           0           dez  02 23:12
gsd-rfkill          jose            2086    25753           4945        0           0           dez  02 23:12
gsd-print-notif     jose            2085    55724           3364        0           0           dez  02 23:12
gsd-printer         jose            2134    77450           912         0           0           dez  02 23:12
gsd-power           jose            2084    305872          12358       0           0           dez  02 23:12
gsd-media-keys      jose            2082    321545          26486       0           0           dez  02 23:12
gsd-keyboard        jose            2080    301988          9148        0           0           dez  02 23:12
gsd-housekeepin     jose            2078    959901          4096        0           0           dez  02 23:12
gsd-disk-utilit     jose            2136    25370           2444        0           0           dez  02 23:12
gsd-datetime        jose            2076    96605           3364        0           0           dez  02 23:12
gsd-color           jose            2075    314539          14776       0           0           dez  02 23:12
gsd-a11y-settin     jose            2073    39213           3433        0           0           dez  02 23:12
goa-daemon          jose            1565    133932          5026        0           0           dez  02 23:12
gnome-shell-cal     jose            1973    117182          5522        0           0           dez  02 23:12
gnome-session-c     jose            1865    24503           64          0           0           dez  02 23:12
gnome-session-b     jose            1872    933750          29452       0           0           dez  02 23:12
gnome-session-b     jose            1635    5912371         29589       0           0           dez  02 23:12
gjs                 jose            2044    71266           2409        0           0           dez  02 23:12
```

Fig.12 – Teste 15

- Argumento “-m” e “-M”

É possível observar na imagem seguinte que ao usar as estas opções e introduzindo no terminal processos com PID’s entre 3000 e 4000 durante 10 segundos, os dados são impressos como esperado.

```
jose@jose-VivoBook-ASUSLaptop-X580GD-N580GD:~/LEI/2ºano/1ºSemestre/SO/Práticas/Projeto_git/SO_A01$ ./rwstat.sh -m 3000 -M 4000 10
COMM                                USER                PID         READB               WRITEB      RATER           RATEW           DATE
chrome                              jose                3924        74067232            3737843     232821.50       1.00            dez  02 23:13
chrome                              jose                3753        479934638           260736607   107780.40       1659.20         dez  02 23:12
chrome                              jose                3992        6303935             4975713     6618.70         5296.60         dez  02 23:13
chrome                              jose                3798        62485189            57844157    1397.30         31517.40        dez  02 23:12
chrome                              jose                3796        1012429             1439641     101.70          101.20          dez  02 23:12
nacl_helper                         jose                3771        10209               120         0               0               dez  02 23:12
chrome                              jose                3828        5999391             5412548     0               0               dez  02 23:12
chrome                              jose                3774        2582837             1850498     0               0               dez  02 23:12
chrome                              jose                3770        142719              120         0               0               dez  02 23:12
chrome                              jose                3769        142705              21          0               0               dez  02 23:12
chrome_crashpad                     jose                3763        7700                0           0               0               dez  02 23:12
chrome_crashpad                     jose                3761        7732                88          0               0               dez  02 23:12
cat                                 jose                3759        4801                509         0               0               dez  02 23:12
cat                                 jose                3758        4292                0           0               0               dez  02 23:12
```

Fig.13 – Teste 16

- Argumento “-w”, “-p” e “-c”

Por fim, ao usarmos uma combinação aleatória de elementos opcionais, de ordenar por ordem os write values, imprimir 5 processos e mostrar todos aqueles que têm o comando começado por “c”, encontramos na imagem seguinte a impressão esperada.

```
jose@jose-VivoBook-ASUSLaptop-X580GD-N580GD:~/LEI/2ºano/1ºSemestre/SO/Práticas/Projeto_git/SO_A01$ ./rwstat.sh -w -p 5 -c "c.*"  10
COMM                                USER                PID         READB               WRITEB      RATER           RATEW           DATE
chrome                              jose                3798        63043119            62952512    289.90          46861.30        dez  02 23:12
chrome                              jose                3992        7394722             5849796     6690.80         5361.70         dez  02 23:13
chrome                              jose                5444        2813012             1855550     936.30          936.30          dez  02 23:16
chrome                              jose                3753        481362507           289878722   98592.80        886.50          dez  02 23:12
code                                jose                4073        39654768            7994601     0               486.40          dez  02 23:13
```

Fig.14 – Teste 17

3. Conclusão

Resumindo, para este projeto foi desenvolvido um script em bash, apto a obter estatísticas sobre leitura e escrita de processos, de acordo com os argumentos passados pelo utilizador em qualquer combinação. Todos estes argumentos passam por uma série de verificações capazes de parar o programa, caso faltarem os argumentos de opções que o utilizador selecionou. O output do programa será uma tabela com as informações relativas aos processos ocorridos num determinado período de segundos.