

CS 340 – Operating Systems

HOP04 - Input, Output and Throughput

7/13/2019 Developed by Kevin Wang
06/17/2020 Reviewed by Kim Nguyen
03/29/2021 Updated by Matt Raio
02/14/2022 Reviewed by Ken Ling



School of Technology & Computing (STC) @ City University of Seattle (CityU)

Before You Start

- This exercise assumes that the user is working with a Linux virtual machine in virtualbox.
- All commands and code discussed in this exercise will run in the Ubuntu console.
- The directory path shown in screenshots may be different from yours.
- Some steps are not explained in the tutorial. If you are not sure what to do:
 1. Consult the resources listed below and experiment in the Ubuntu console and try to solve the problem yourself.
 2. If you cannot solve the problem after a few tries, ask a TA for help.

Resources

- Linux command line: bash + utilities <https://ss64.com/bash/>
- Nano/Basics Guide https://wiki.gentoo.org/wiki/Nano/Basics_Guide

Preparation

Connect to your Ubuntu instance Using parameters and variables

1. Type the following command to reuse HelloWorld.sh file:

```
nano HelloWorld.sh
```

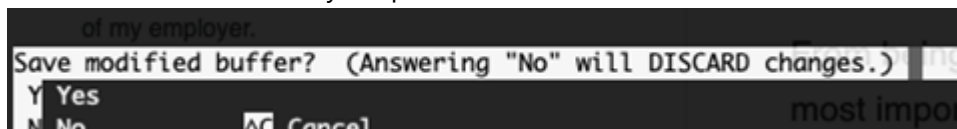
- Update the file content as below:

```
#!/bin/bash
printf "Hello %s!\n" "$1"
printf "Value of all the positional parameters: %s\n" "$*"
printf "The number of positional parameters: %s\n" "$#"
printf "The name of the current running script: %s\n" "$0"
printf "The process identification number (PID): %s\n" "$$"
printf "The exit code of the last-executed command: %s\n" "$?"

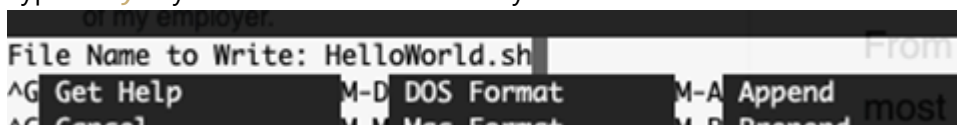
# Define a variable and use it
newName="Arthur Wang"
printf "I have a cool name which is %s\n" "$newName"
```

- Note: A positional parameter is an argument specified on the command line. Positional parameter values are stored in a special set of variables that can be accessed via positional parameters. For example, the first, second, and third positional parameter values can be retrieved via the positional parameter `$1`, `$2`, and `$3`, respectively. Parameters greater than 9 can be accessed by using curly braces around the number; for instance, `${10}` would be the `tenth parameter`. Besides positional parameters, there are other special parameters to retrieve other information, such as PID and exit code of the last executed command, as shown in the first part of the script.

- Click the `control + x` key to quit



- Type the `y` key and then hit the `enter` key to save the file



- Type the following command to see the result (please change the name to your name) Note: the command below has 3 parts: `interpreter` (i.e. bash shell); `command name` (i.e. `HelloWorld.sh`); and `arguments` (i.e. `Kevin Wang`). `bash HelloWorld.sh YourName`

```
ubuntu@ip-172-31-25-112:~/CS120/Module2$ bash HelloWorld.sh Kevin Wang
Hello Kevin!
Value of all the positional parameters: Kevin Wang
The number of positional parameters: 2
The path to the current running script: HelloWorld.sh
The process identification number (PID): 15611
The exit code of the last-executed command: 0
I have a cool name which is Arthur Wang
```

Format and print data with printf command

The basic syntax for printf command is: `printf format arg1 arg2 ...` You can find all escape codes and format specifiers here: <https://wiki-dev.bash-hackers.org/commands/builtin/printf>

1. Create a `TestPrintf.sh` by typing the following command in the console command line:

```
nano TestPrintf.sh
```

2. Type scripts in the file as below:

```
# ! /bin/bash
printf "Two tabs after me\t\tA new line after me\nThe hexadecimal digits X42 represents \x42\n"

printf "%s\n" Print in separated lines "Print in one line"
printf "%b\n" "%b tells printf to escape sequences. Two tabs after me\t\t ."
printf "%s%d\n%s%f\n%s%e\n" "%d prints integer such as: " 25 "%f prints float numbers such as: " 25.5 \ "%e prints with the exponential notation such as: " 25.5
printf "%s#%02x%02x%02x\n" "%x prints in hexadecimal. Here is an example to convert a RGB color 82 185 225 to a hex notation: " \ 82 185 255

# Use width specification
header="\n %-10s %-18s %8s\n"
format=" %-10d %-18s %8.2f\n"
printf "$header" Id Name "Order price"
printf "=====\n"
printf "$format" 1 "Kevin Wang" 234.30 2 "Arthur B" 332.23 3 "Evan A" 525.32
```

```
#!/bin/bash
printf "Two tabs after me\t\tA new line after me\nThe hexadecimal digits X42 represents \x42\n"

printf "%s\n" Print in separated limes "Print in one line"
printf "%b\n" "%b tells printf to escape sequences. Two tabs after me\t\t ."
printf "%s%d\n%s%f\n%s%e\n" "%d prints integer such as: " 25 "%f prints float numbers such as: " 25.5 \ "%e prints with the exponential notation such as: " 25.5
printf "%s#%02x%02x%02x\n" "%x prints in hexadecimal. Here is an example to convert a RGB color 82 185 225 to a hex notation: " \ 82 185 255

#Use width specification
header="\n %-10s %-18s %8s\n"
format=" %-10d %-18s %8.2f\n"
printf "$header" Id Name "Order price"
printf "=====\n"
printf "$format" 1 "Kevin Wang" 234.30 2 "Arthur B" 332.23 3 "Evan A" 525.32
```

3. Hit the `control + x` key to quit and save the file
4. Execute the file to see the result by typing following command:

```
bash TestPrintf.sh
```

Input and Output

1. Create a file by using the standard output operation >

Type the following command in the console command line:

```
printf "%s\n" "My text in the file" > test
```

2. Check the file content by typing the following command:

```
cat test
```

```
ubuntu@ip-172-31-25-112:~/CS120/Module2$ printf "%s\n" "My text in the file" > test
ubuntu@ip-172-31-25-112:~/CS120/Module2$ cat test
My text in the file
```

3. Read data from a file to a variable by using the standard input operation <

```
read content < test
```

4. Check the variable value by typing the following command:

```
echo $content
```

```
ubuntu@ip-172-31-25-112:~/CS120/Module2$ read content < test
ubuntu@ip-172-31-25-112:~/CS120/Module2$ echo $content
My text in the file
```

Using pipelines and command substitution

A pipeline can help users to take one command's output as another command's input

1. Save top command result to a file by typing the following command in the console command line:

```
top -n 1 | tee result
# You will see the result from top command
```

2. Clear the screen by typing the following command:

```
clear
```

3. Check the result file:

```
cat result
```

```
Tasks: 86 total, 1 running, 52 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 1007520 total, 179192 free, 108544 used, 719784 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 739320 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	159848	9132	6708	S	0.0	0.9	0:04.69	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworker/0:0H
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_percpu_wq
7	root	20	0	0	0	0	S	0.0	0.0	0:00.26	ksoftirqd/0
8	root	20	0	0	0	0	I	0.0	0.0	0:00.62	rcu_sched
9	root	20	0	0	0	0	I	0.0	0.0	0:00.00	rcu_bh
10	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
11	root	rt	0	0	0	0	S	0.0	0.0	0:00.56	watchdog/0

4. We also can use the command substitution to store the command output to a variable:

```
lsResult=$(ls)
# We are saving the ls command's output to the lsResult variable
```

5. Check the variable value by typing the bash command:

```
echo $lsResult
```

Take a screenshot of your `printf` commands as well as your pipelines and substitutions.

What commands did you have to substitute? Were there any challenges you faced with the instructions vs. what your OS had in its environment?

I didn't have any issues besides my shared folder not sharing with my instance otherwise it was fine.

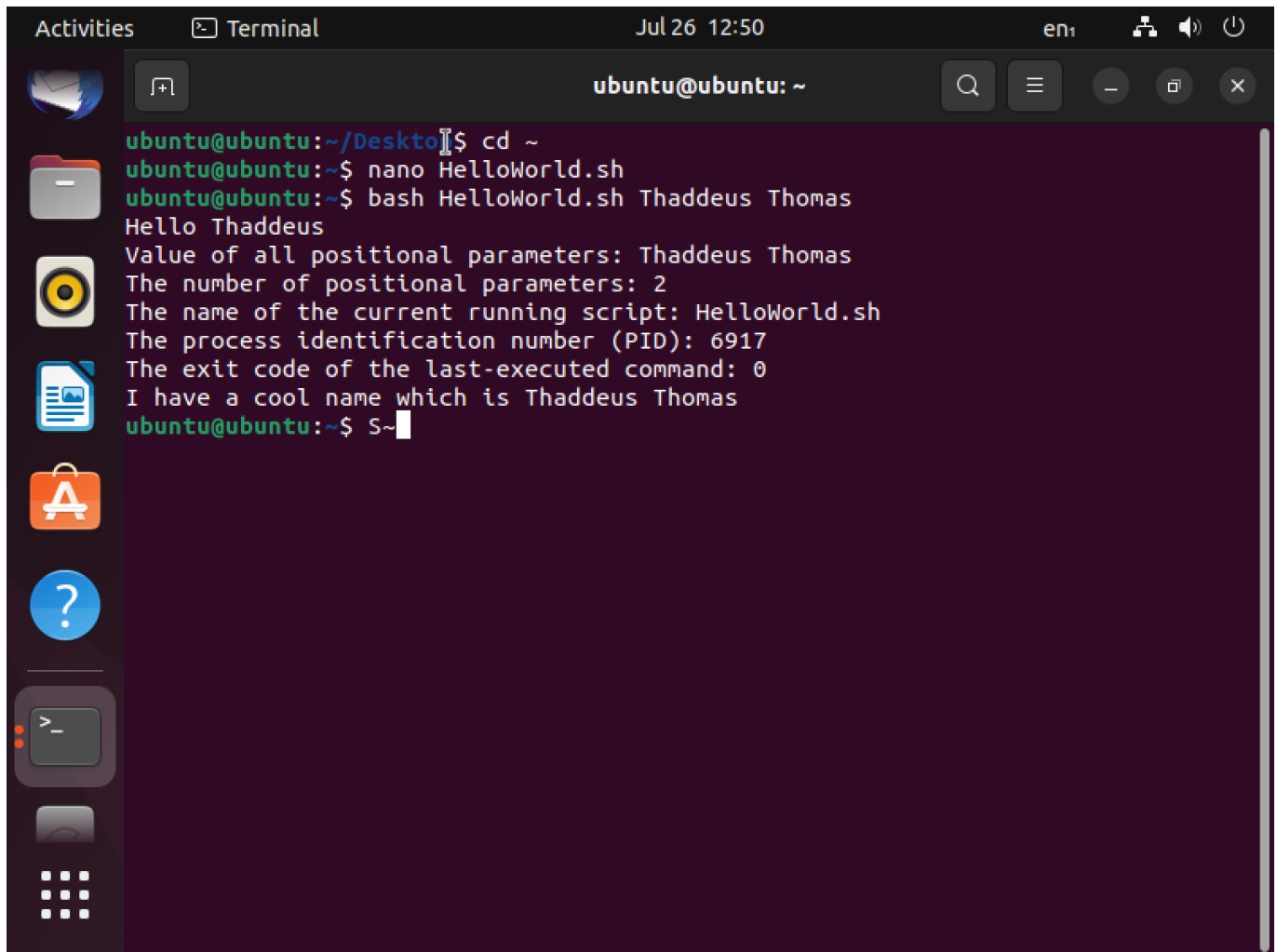
Submit your Work to Brightspace

Please upload all your files for this hands-on practice to the HOP assignment on Brightspace.

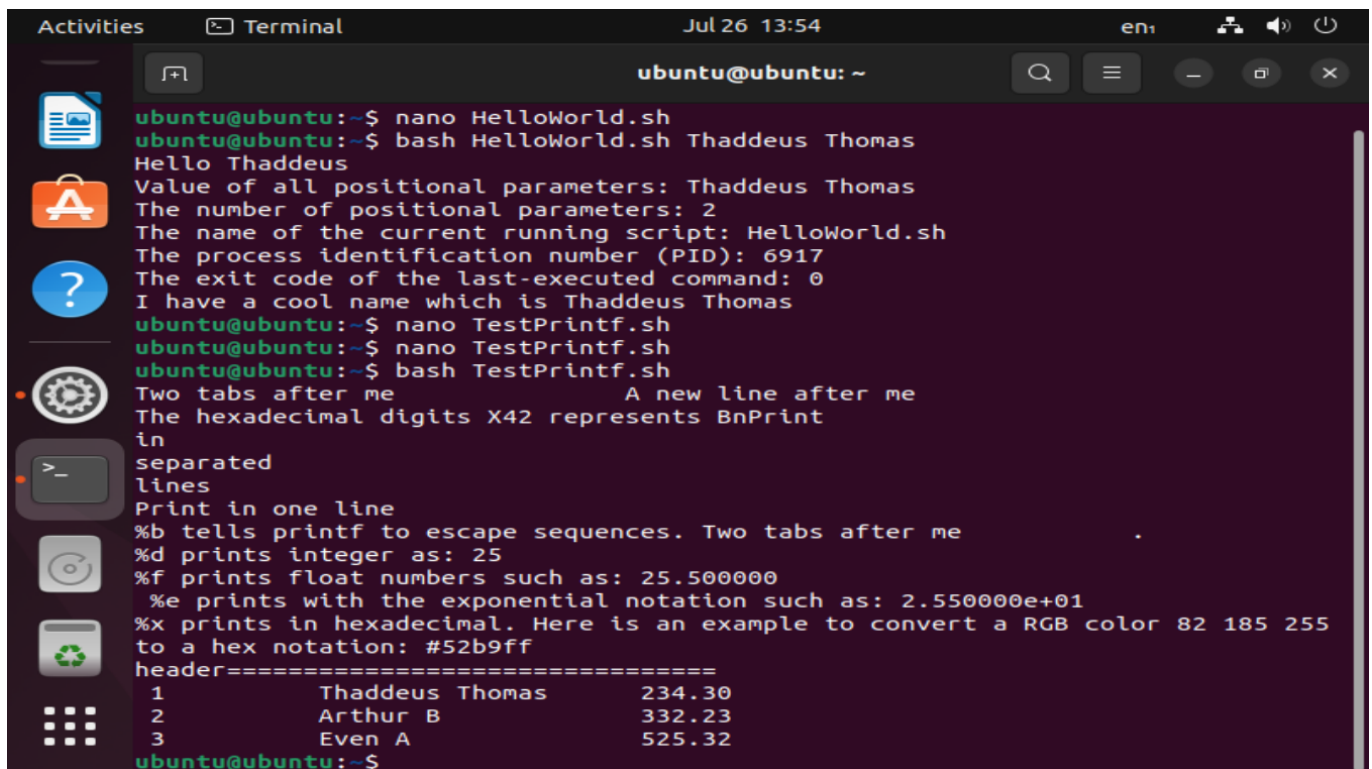
Shell Scripts

 TestPrintf.sh  HelloWorld.sh

Screenshots



```
ubuntu@ubuntu: ~/Desktop$ cd ~
ubuntu@ubuntu:~$ nano HelloWorld.sh
ubuntu@ubuntu:~$ bash HelloWorld.sh Thaddeus Thomas
Hello Thaddeus
Value of all positional parameters: Thaddeus Thomas
The number of positional parameters: 2
The name of the current running script: HelloWorld.sh
The process identification number (PID): 6917
The exit code of the last-executed command: 0
I have a cool name which is Thaddeus Thomas
ubuntu@ubuntu:~$ S~
```



```
ubuntu@ubuntu:~$ nano HelloWorld.sh
ubuntu@ubuntu:~$ bash HelloWorld.sh Thaddeus Thomas
Hello Thaddeus
Value of all positional parameters: Thaddeus Thomas
The number of positional parameters: 2
The name of the current running script: HelloWorld.sh
The process identification number (PID): 6917
The exit code of the last-executed command: 0
I have a cool name which is Thaddeus Thomas
ubuntu@ubuntu:~$ nano TestPrintf.sh
ubuntu@ubuntu:~$ nano TestPrintf.sh
ubuntu@ubuntu:~$ bash TestPrintf.sh
Two tabs after me           A new line after me
The hexadecimal digits X42 represents BnPrint
in
separated
lines
Print in one line
%b tells printf to escape sequences. Two tabs after me      .
%d prints integer as: 25
%f prints float numbers such as: 25.500000
%e prints with the exponential notation such as: 2.550000e+01
%x prints in hexadecimal. Here is an example to convert a RGB color 82 185 255
to a hex notation: #52b9ff
header=====
1      Thaddeus Thomas      234.30
2      Arthur B             332.23
3      Even A                525.32
ubuntu@ubuntu:~$
```



```

Activities  Terminal  Jul 26 13:56  en1
ubuntu@ubuntu: ~

The process identification number (PID): 6917
The exit code of the last-executed command: 0
I have a cool name which is Thaddeus Thomas
ubuntu@ubuntu:~$ nano TestPrintf.sh
ubuntu@ubuntu:~$ nano TestPrintf.sh
ubuntu@ubuntu:~$ bash TestPrintf.sh
Two tabs after me          A new line after me
The hexadecimal digits X42 represents BnPrint
in
separated
lines
Print in one line
%b tells printf to escape sequences. Two tabs after me
%d prints integer as: 25
%f prints float numbers such as: 25.500000
%e prints with the exponential notation such as: 2.550000e+01
%x prints in hexadecimal. Here is an example to convert a RGB color 82 185 255
to a hex notation: #52b9ff
header=====
1          Thaddeus Thomas      234.30
2          Arthur B            332.23
3          Even A              525.32
ubuntu@ubuntu:~$ printf "%s\n" "My text in the file" > test
ubuntu@ubuntu:~$ cat test
My text in the file
ubuntu@ubuntu:~$ read content < test
ubuntu@ubuntu:~$ echo $content
My text in the file
ubuntu@ubuntu:~$

```

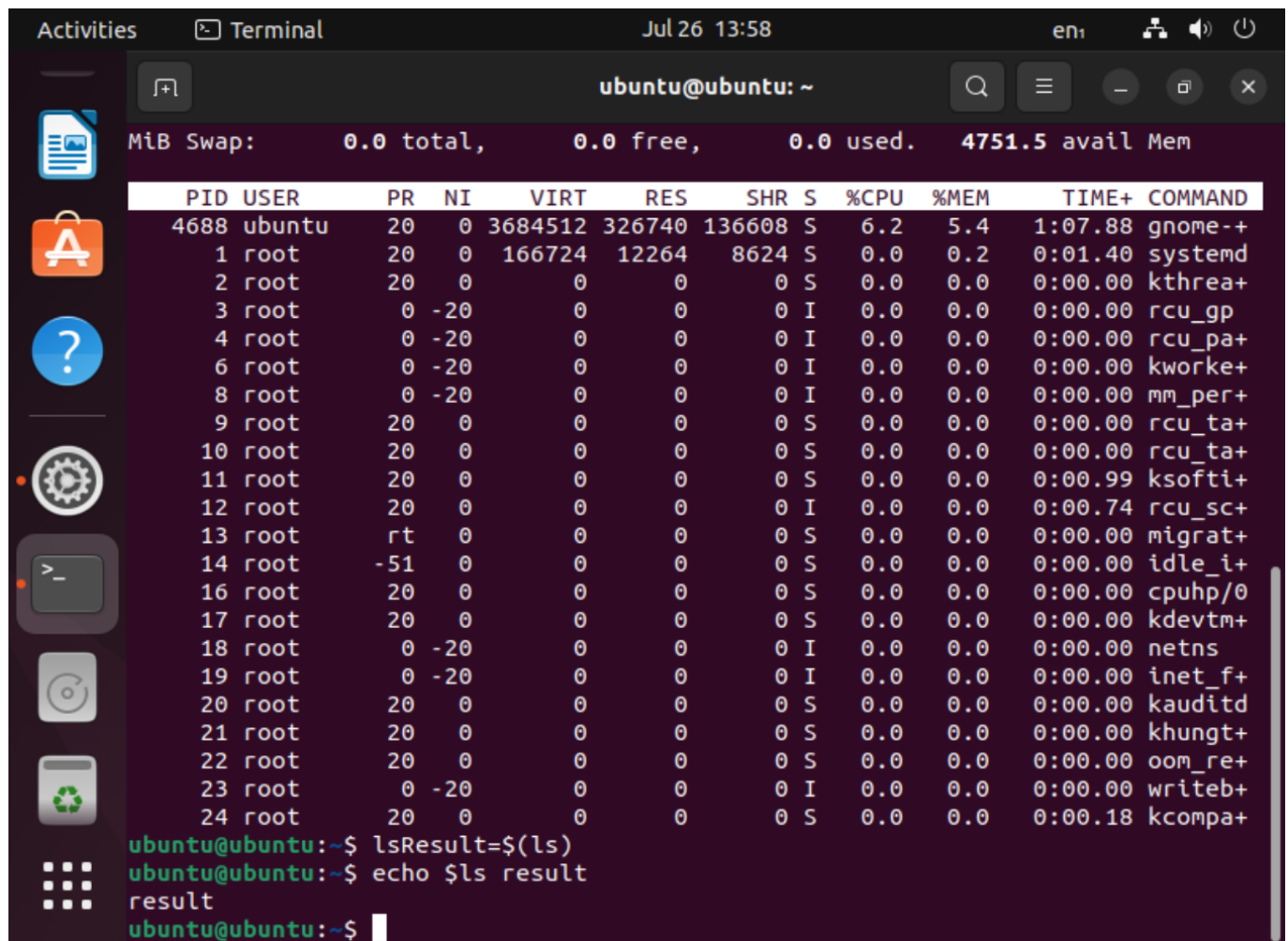
```

Activities  Terminal  Jul 26 13:57  en1
ubuntu@ubuntu: ~

Tasks: 194 total,  1 running, 193 sleeping,  0 stopped,  0 zombie
%Cpu(s):  0.0 us,  7.1 sy,  0.0 ni, 92.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  5936.0 total,  1645.6 free,  845.8 used,  3444.7 buff/cache
MiB Swap:   0.0 total,   0.0 free,   0.0 used.  4751.5 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR S  %CPU  %MEM    TIME+  COMMAND
  4688 ubuntu    20   0 3684512 326740 136608 S   6.2   5.4   1:07.88  gnome-+
    1 root      20   0  166724  12264   8624 S   0.0   0.2   0:01.40  systemd
    2 root      20   0        0        0        0 S   0.0   0.0   0:00.00  kthrea+
    3 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  rcu_gp
    4 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  rcu_pa+
    6 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  kworke+
    8 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  mm_per+
    9 root      20   0        0        0        0 S   0.0   0.0   0:00.00  rcu_ta+
   10 root      20   0        0        0        0 S   0.0   0.0   0:00.00  rcu_ta+
   11 root      20   0        0        0        0 S   0.0   0.0   0:00.99  ksofti+
   12 root      20   0        0        0        0 I   0.0   0.0   0:00.74  rcu_sc+
   13 root      rt    0        0        0        0 S   0.0   0.0   0:00.00  migrat+
   14 root     -51   0        0        0        0 S   0.0   0.0   0:00.00  idle_i+
   16 root      20   0        0        0        0 S   0.0   0.0   0:00.00  cpuhp/0
   17 root      20   0        0        0        0 S   0.0   0.0   0:00.00  kdevtm+
   18 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  netns
   19 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  inet_f+
   20 root      20   0        0        0        0 S   0.0   0.0   0:00.00  kauditd
   21 root      20   0        0        0        0 S   0.0   0.0   0:00.00  khungtd
   22 root      20   0        0        0        0 S   0.0   0.0   0:00.00  oom_re+
   23 root       0 -20        0        0        0 I   0.0   0.0   0:00.00  writeb+
   24 root      20   0        0        0        0 S   0.0   0.0   0:00.18  kcompa+
ubuntu@ubuntu:~$

```



Activities Terminal Jul 26 13:58 en1

ubuntu@ubuntu: ~

MiB Swap: 0.0 total, 0.0 free, 0.0 used. 4751.5 avail Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4688	ubuntu	20	0	3684512	326740	136608	S	6.2	5.4	1:07.88	gnome-+
1	root	20	0	166724	12264	8624	S	0.0	0.2	0:01.40	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthrea+
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_pa+
6	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	kworke+
8	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	mm_per+
9	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_ta+
10	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_ta+
11	root	20	0	0	0	0	S	0.0	0.0	0:00.99	ksofti+
12	root	20	0	0	0	0	I	0.0	0.0	0:00.74	rcu_sc+
13	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migrat+
14	root	-51	0	0	0	0	S	0.0	0.0	0:00.00	idle_i+
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuhp/0
17	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kdevtm+
18	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	netns
19	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	inet_f+
20	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kauditd
21	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khung+
22	root	20	0	0	0	0	S	0.0	0.0	0:00.00	oom_re+
23	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	writeb+
24	root	20	0	0	0	0	S	0.0	0.0	0:00.18	kcompa+

```

ubuntu@ubuntu:~$ lsResult=$(ls)
ubuntu@ubuntu:~$ echo $ls result
result
ubuntu@ubuntu:~$

```