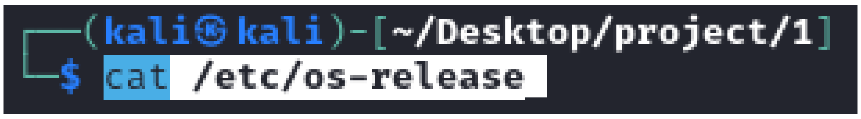
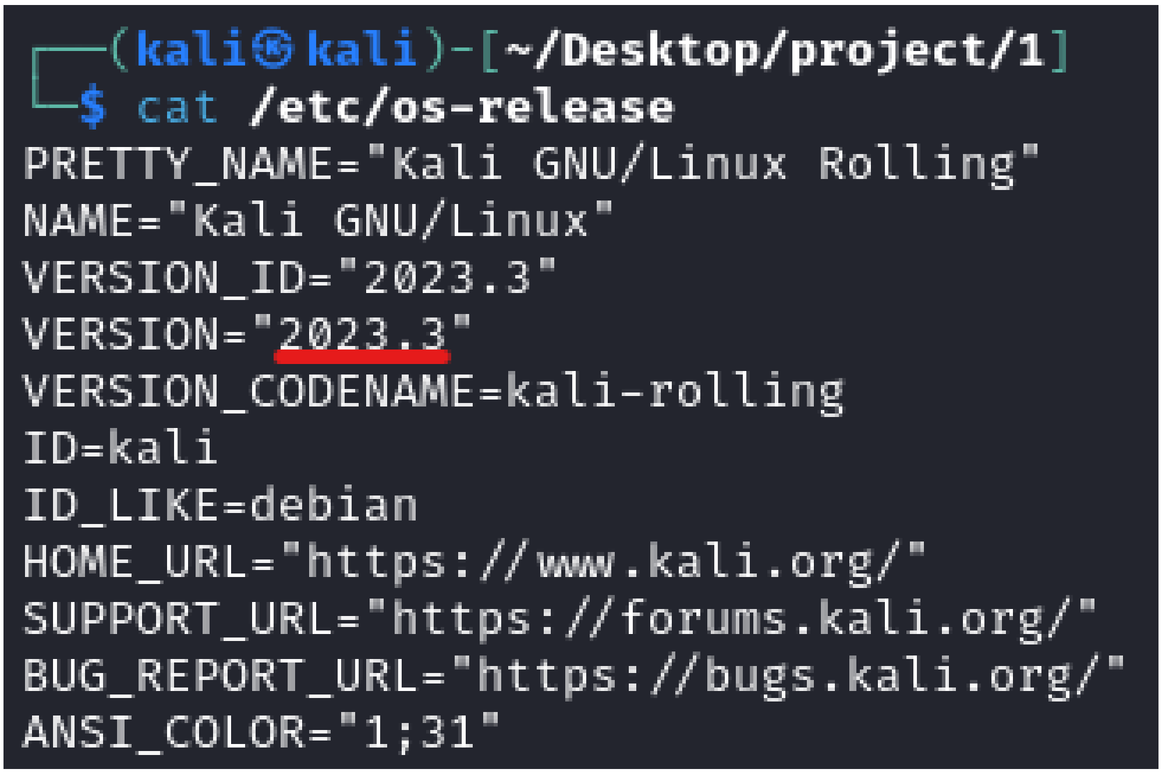
#1. Display the Linux version.

To find the version of the Linux machine, the information can be found using cat /etc/os-release



```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ cat /etc/os-release
```

The Version is "2023.3"

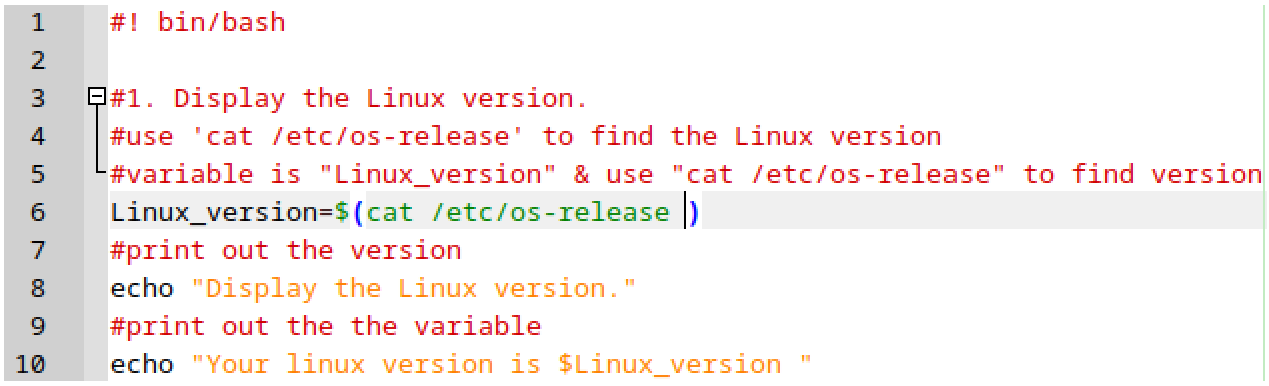


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ cat /etc/os-release
PRETTY_NAME="Kali GNU/Linux Rolling"
NAME="Kali GNU/Linux"
VERSION_ID="2023.3"
VERSION="2023.3"
VERSION_CODENAME=kali-rolling
ID=kali
ID_LIKE=debian
HOME_URL="https://www.kali.org/"
SUPPORT_URL="https://forums.kali.org/"
BUG_REPORT_URL="https://bugs.kali.org/"
ANSI_COLOR="1;31"
```

Next we open up geany to start moving over cat /etc/os-release

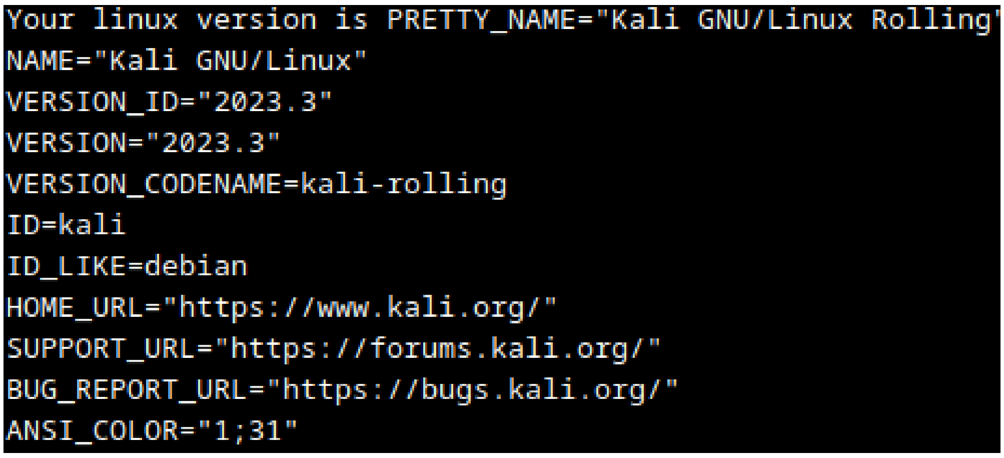
Added echo "Display the Linux version."

Tired to set the variable as cat /etc/os-release



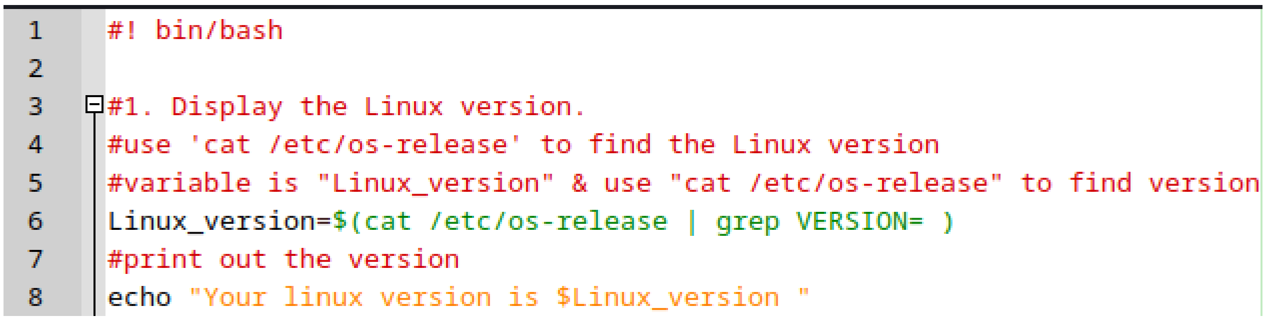
```
#! bin/bash

#1. Display the Linux version.
#use 'cat /etc/os-release' to find the Linux version
#variable is "Linux_version" & use "cat /etc/os-release" to find version
Linux_version=$(cat /etc/os-release )
#print out the version
echo "Display the Linux version."
#print out the the variable
echo "Your linux version is $Linux_version "
```

But the result is too long, and not right.

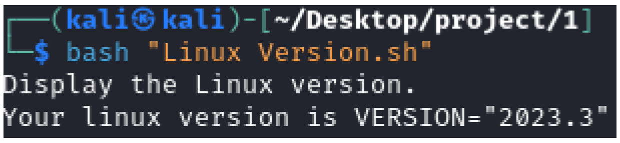


```
Your linux version is PRETTY_NAME="Kali GNU/Linux Rolling"
NAME="Kali GNU/Linux"
VERSION_ID="2023.3"
VERSION="2023.3"
VERSION_CODENAME=kali-rolling
ID=kali
ID_LIKE=debian
HOME_URL="https://www.kali.org/"
SUPPORT_URL="https://forums.kali.org/"
BUG_REPORT_URL="https://bugs.kali.org/"
ANSI_COLOR="1;31"
```

Using grep to find the version of Linux.

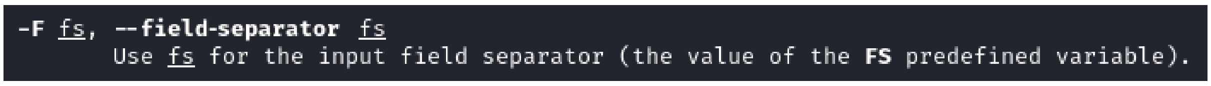


```
#! bin/bash

#1. Display the Linux version.
#use 'cat /etc/os-release' to find the Linux version
#variable is "Linux_version" & use "cat /etc/os-release" to find version
Linux_version=$(cat /etc/os-release | grep VERSION= )
#print out the version
echo "Your linux version is $Linux_version "
```



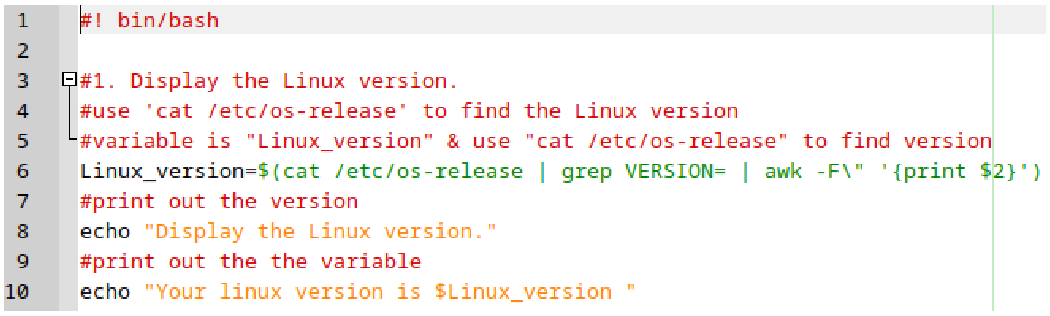
```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ bash "Linux Version.sh"
Display the Linux version.
Your linux version is VERSION="2023.3"
```

'grep' the VERSION="2023.3"

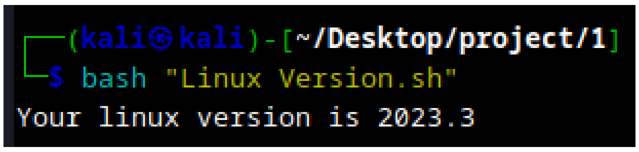


```
-F fs, --field-separator fs
       Use fs for the input field separator (the value of the FS predefined variable).
```

Add a field separator to remove the " then print $2



```
#! bin/bash

#1. Display the Linux version.
#use 'cat /etc/os-release' to find the Linux version
#variable is "Linux_version" & use "cat /etc/os-release" to find version
Linux_version=$(cat /etc/os-release | grep VERSION= | awk -F\" '{print $2}')
#print out the version
echo "Display the Linux version."
#print out the the variable
echo "Your linux version is $Linux_version "
```

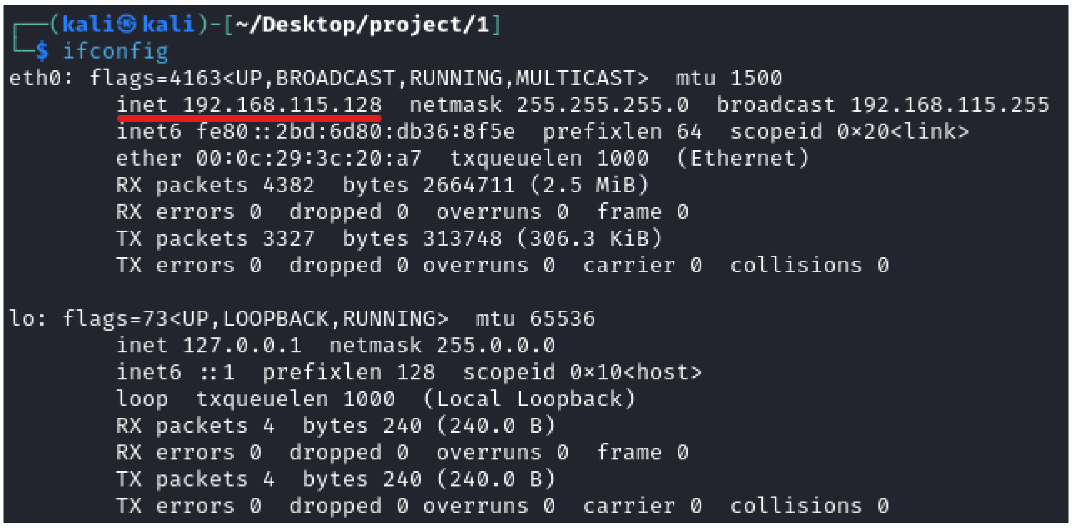


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ bash "Linux Version.sh"
Your linux version is 2023.3
```

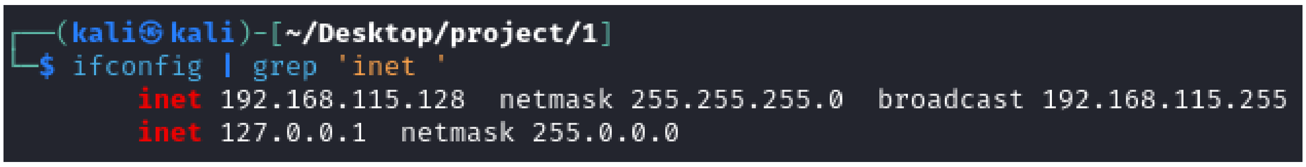
The version of kali linux is 2023.3

#2. Display the private IP address, public IP address, and the default gateway.

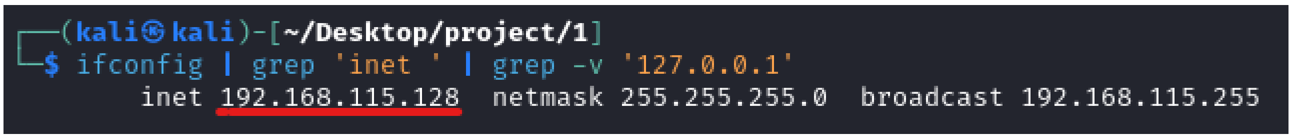
1. Find the private IP address using ifconfig


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.115.128  netmask 255.255.255.0  broadcast 192.168.115.255
        inet6 fe80::2bd:6d80:db36:8f5e  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:3c:20:a7  txqueuelen 1000  (Ethernet)
        RX packets 4382  bytes 2664711 (2.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3327  bytes 313748 (306.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

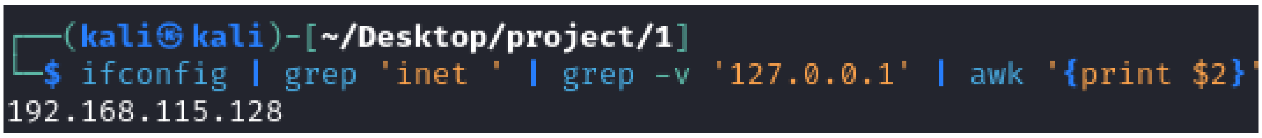
What we want is to grep the IP address after the "inet "


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ ifconfig | grep 'inet '
        inet 192.168.115.128  netmask 255.255.255.0  broadcast 192.168.115.255
        inet 127.0.0.1  netmask 255.0.0.0
```

But we get error, so we need to apply -v to remove the loopback address/localhost.

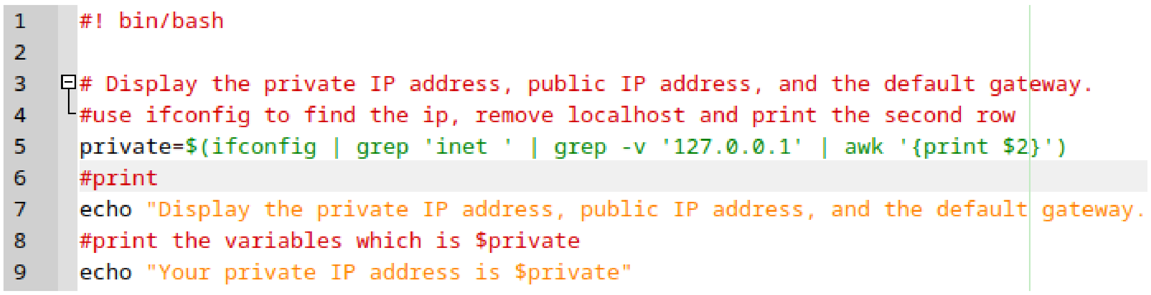

```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ ifconfig | grep 'inet ' | grep -v '127.0.0.1'
        inet 192.168.115.128  netmask 255.255.255.0  broadcast 192.168.115.255
```

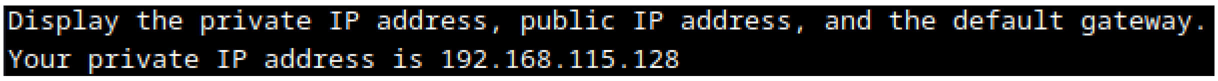
Now we manage to remove 127.0.0.1 but we only wanted to display the private IP.


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}'
192.168.115.128
```

Using awk to print the second row, we manage to display the private IP address (192.168.115.128)

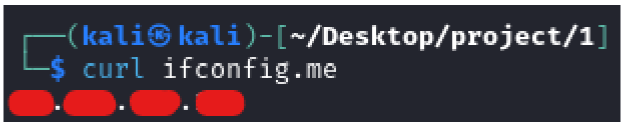
Now transferring to the geany and write the script.


```
#! bin/bash

# Display the private IP address, public IP address, and the default gateway.
#use ifconfig to find the ip, remove localhost and print the second row
private=$(ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}')
#print
echo "Display the private IP address, public IP address, and the default gateway."
#print the variables which is $private
echo "Your private IP address is $private"
```


```
Display the private IP address, public IP address, and the default gateway.
Your private IP address is 192.168.115.128
```

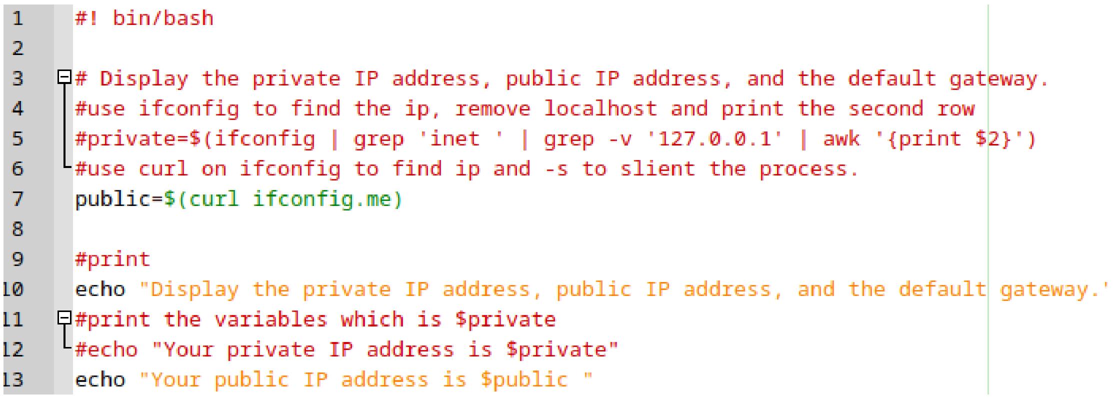
The private IP address printed correctly.

2. Find the public IP address using curl ifconfig.me

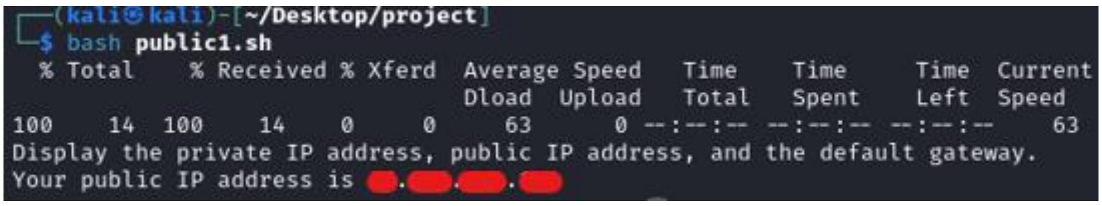


```
┌──(kali㉿kali)-[~/Desktop/project/1]
└─$ curl ifconfig.me
[redacted].[redacted].[redacted].[redacted]
```

No issue in the terminal so I process to transfer it to geany.

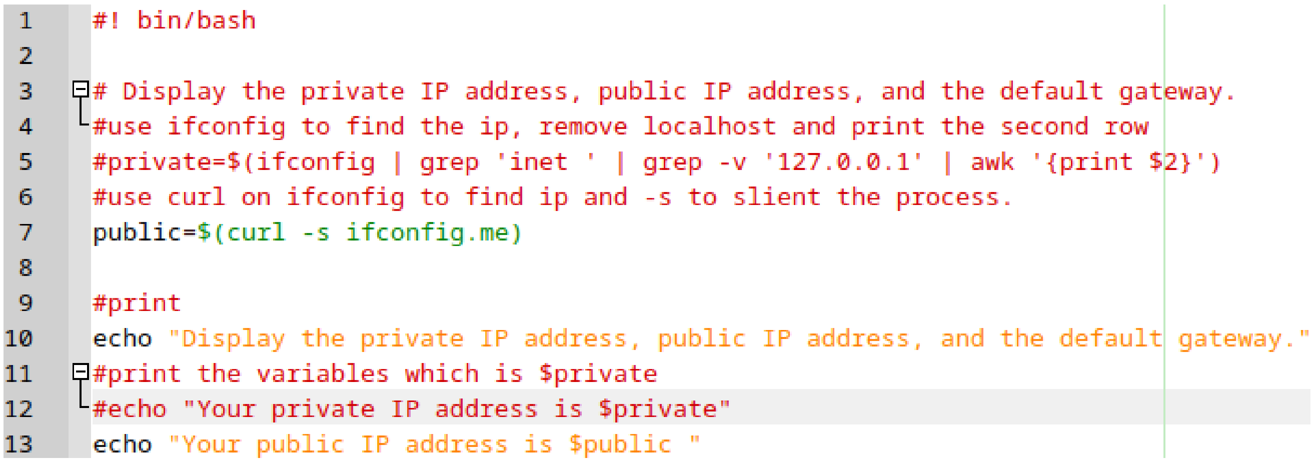


```
1   #! bin/bash
2
3   # Display the private IP address, public IP address, and the default gateway.
4   #use ifconfig to find the ip, remove localhost and print the second row
5   #private=$(ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}')
6   #use curl on ifconfig to find ip and -s to slient the process.
7   public=$(curl ifconfig.me)
8
9   #print
10  echo "Display the private IP address, public IP address, and the default gateway."
11  #print the variables which is $private
12  #echo "Your private IP address is $private"
13  echo "Your public IP address is $public "
```



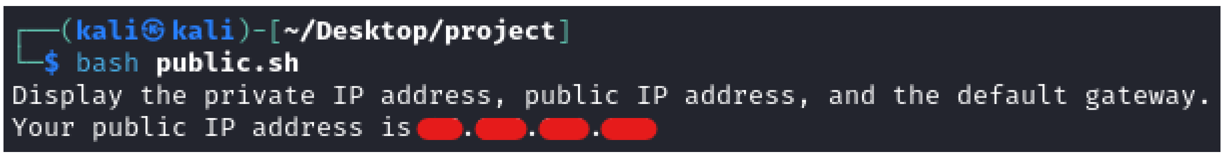
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ bash public1.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    14  100    14    0     0     63      0 --:--:-- --:--:-- --:--:--    63
Display the private IP address, public IP address, and the default gateway.
Your public IP address is [redacted].[redacted].[redacted].[redacted]
```

The output is not displaying correctly, I realise I dint include the -s.



```
1   #! bin/bash
2
3   # Display the private IP address, public IP address, and the default gateway.
4   #use ifconfig to find the ip, remove localhost and print the second row
5   #private=$(ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}')
6   #use curl on ifconfig to find ip and -s to slient the process.
7   public=$(curl -s ifconfig.me)
8
9   #print
10  echo "Display the private IP address, public IP address, and the default gateway."
11  #print the variables which is $private
12  #echo "Your private IP address is $private"
13  echo "Your public IP address is $public "
```

And I added #to prevent my confusing myself when the output is out. (line 5 & 11)

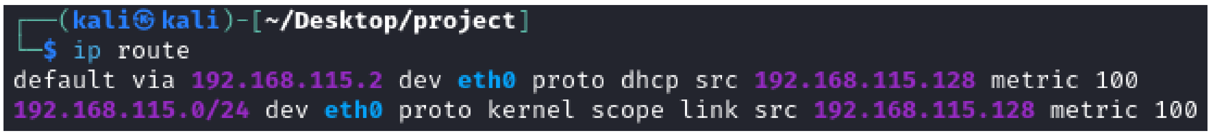


```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ bash public.sh
Display the private IP address, public IP address, and the default gateway.
Your public IP address is [redacted].[redacted].[redacted].[redacted]
```

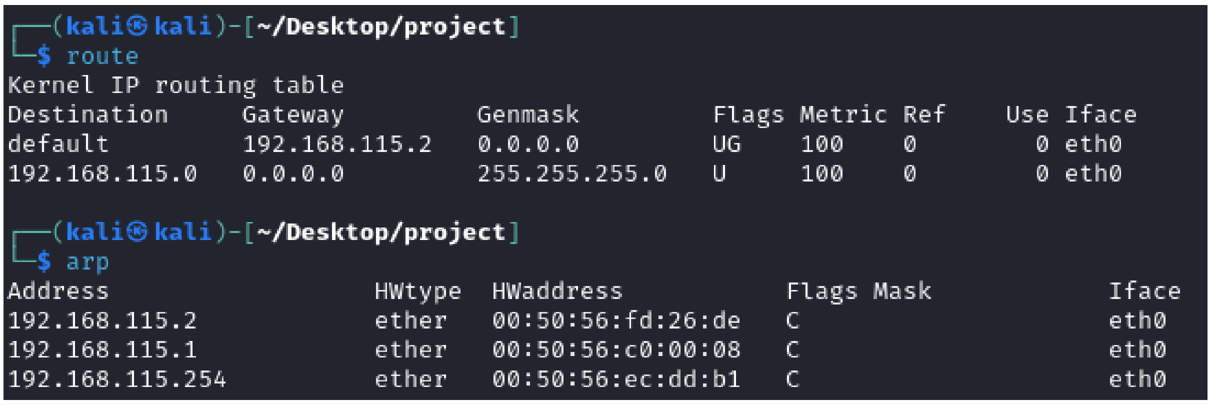
After adding “-s” the output is displaying the correct IP address.

3. Find the default gateway.

Using “ip route” I manage to find the default gateway in the terminal.

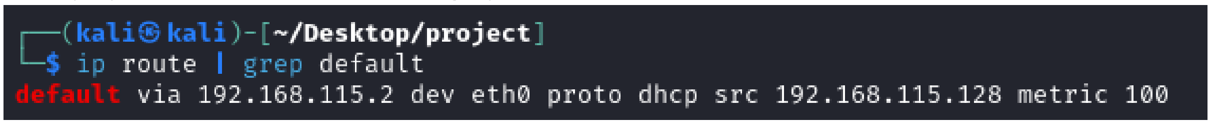


```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ ip route
default via 192.168.115.2 dev eth0 proto dhcp src 192.168.115.128 metric 100
192.168.115.0/24 dev eth0 proto kernel scope link src 192.168.115.128 metric 100
```



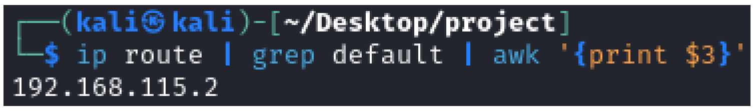
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.115.2   0.0.0.0         UG    100    0        0 eth0
192.168.115.0   0.0.0.0         255.255.255.0   U     100    0        0 eth0

┌──(kali㉿kali)-[~/Desktop/project]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.115.2            ether   00:50:56:fd:26:de   C                     eth0
192.168.115.1            ether   00:50:56:c0:00:08   C                     eth0
192.168.115.254          ether   00:50:56:ec:dd:b1   C                     eth0
```

To display default row, the commend grep is used on “default”



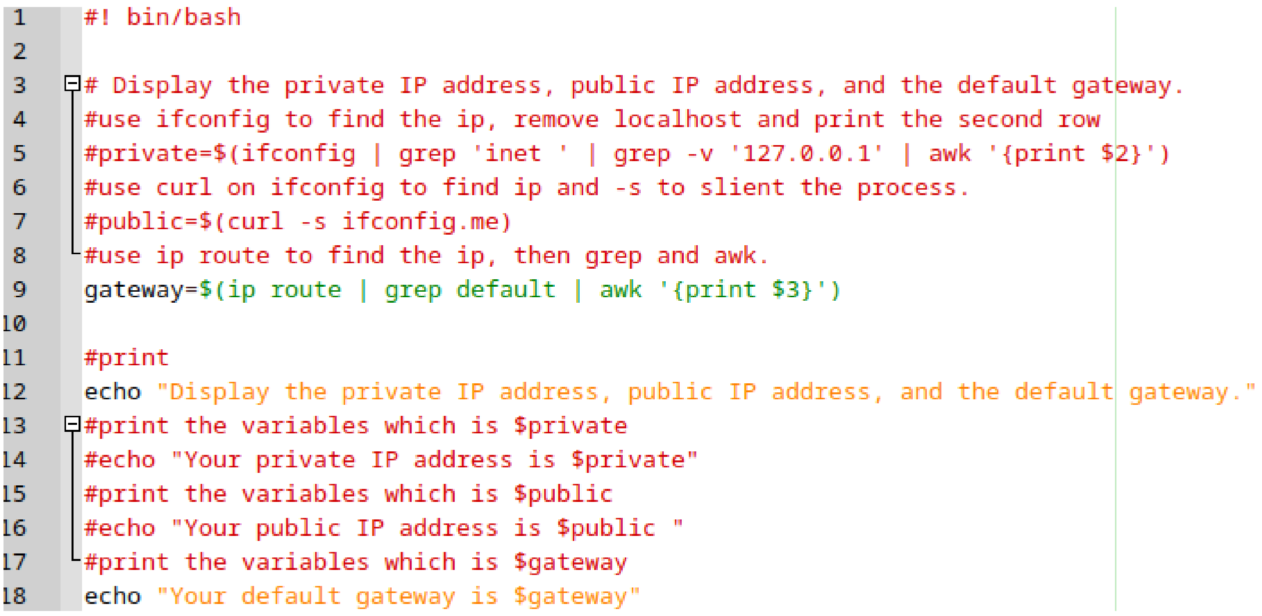
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ ip route | grep default
default via 192.168.115.2 dev eth0 proto dhcp src 192.168.115.128 metric 100
```

To display only the IP address, the commend awk is used on column 3

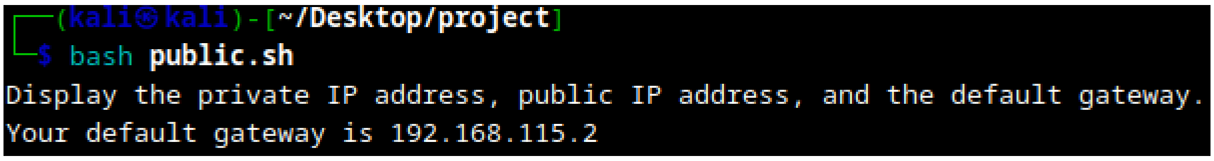


```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ ip route | grep default | awk '{print $3}'
192.168.115.2
```

After everything is working accordingly, I process on geany.

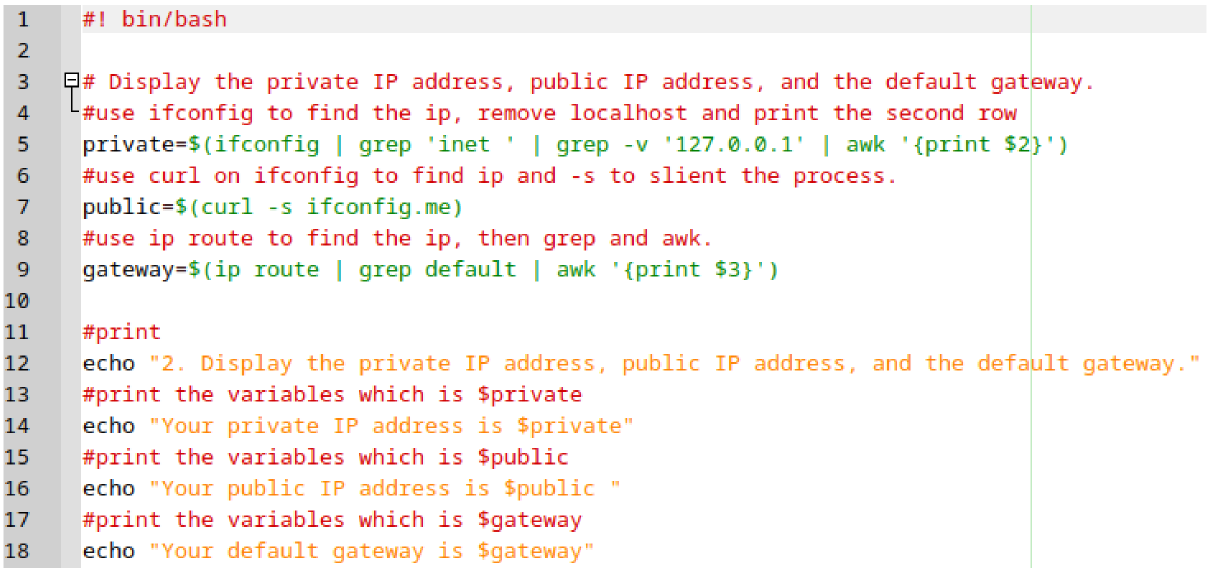


```
#! bin/bash

# Display the private IP address, public IP address, and the default gateway.
#use ifconfig to find the ip, remove localhost and print the second row
#private=$(ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}')
#use curl on ifconfig to find ip and -s to slient the process.
#public=$(curl -s ifconfig.me)
#use ip route to find the ip, then grep and awk.
gateway=$(ip route | grep default | awk '{print $3}')

#print
echo "Display the private IP address, public IP address, and the default gateway."
#print the variables which is $private
#echo "Your private IP address is $private"
#print the variables which is $public
#echo "Your public IP address is $public "
#print the variables which is $gateway
echo "Your default gateway is $gateway"
```



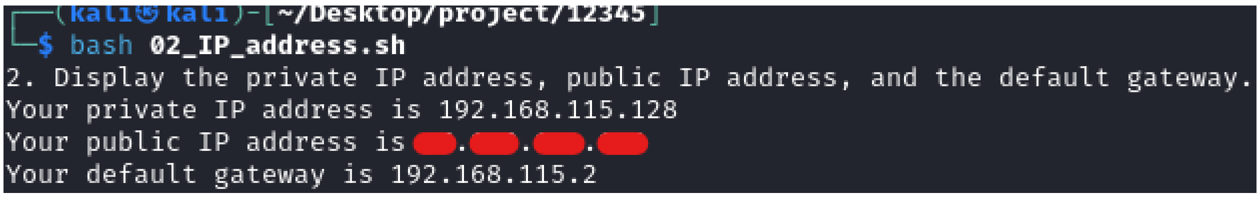
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ bash public.sh
Display the private IP address, public IP address, and the default gateway.
Your default gateway is 192.168.115.2
```

The default gateway is displaying the correct IP address.



```
1   #! bin/bash
2
3   # Display the private IP address, public IP address, and the default gateway.
4   #use ifconfig to find the ip, remove localhost and print the second row
5   private=$(ifconfig | grep 'inet ' | grep -v '127.0.0.1' | awk '{print $2}')
6   #use curl on ifconfig to find ip and -s to slient the process.
7   public=$(curl -s ifconfig.me)
8   #use ip route to find the ip, then grep and awk.
9   gateway=$(ip route | grep default | awk '{print $3}')
10
11  #print
12  echo "2. Display the private IP address, public IP address, and the default gateway."
13  #print the variables which is $private
14  echo "Your private IP address is $private"
15  #print the variables which is $public
16  echo "Your public IP address is $public "
17  #print the variables which is $gateway
18  echo "Your default gateway is $gateway"
```

Remove the # on line 5,7,14,16.



```
┌──(kali㉿kali)-[~/Desktop/project/12345]
└─$ bash 02_IP_address.sh
2. Display the private IP address, public IP address, and the default gateway.
Your private IP address is 192.168.115.128
Your public IP address is [redacted].[redacted].[redacted].[redacted]
Your default gateway is 192.168.115.2
```

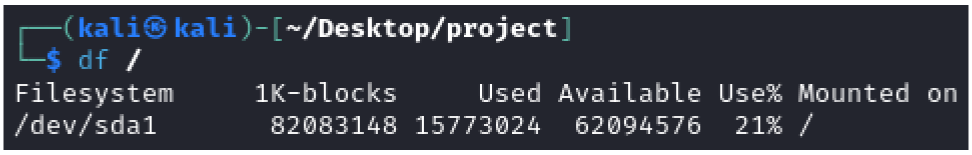
The bash script is displaying the correct information.

#3. Display the hard disk size, free and used space.

1. Display the hard disk size

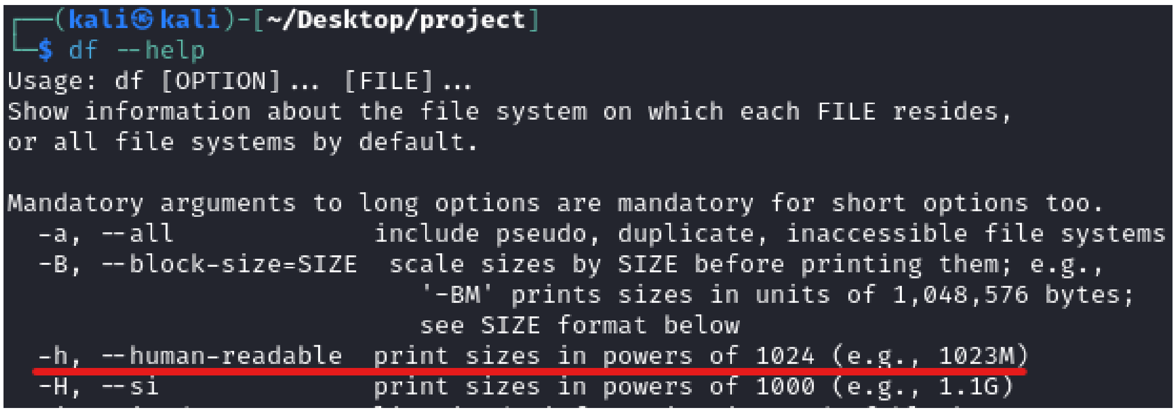
2. Display the hard disk free space

3. Display the hard disk used space



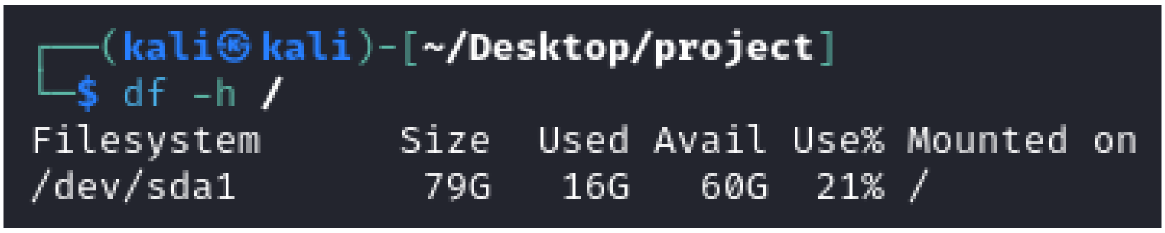
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ df /
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/sda1       82083148 15773024  62094576  21% /
```

The commend “df /” displayed the size but the number is too long.



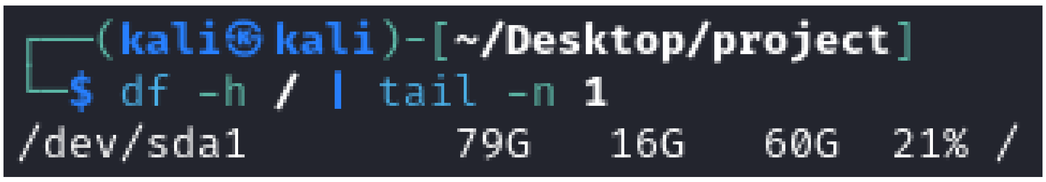
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ df --help
Usage: df [OPTION]... [FILE]...
Show information about the file system on which each FILE resides,
or all file systems by default.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all             include pseudo, duplicate, inaccessible file systems
  -B, --block-size=SIZE  scale sizes by SIZE before printing them; e.g.,
                           '-BM' prints sizes in units of 1,048,576 bytes;
                           see SIZE format below
  -h, --human-readable  print sizes in powers of 1024 (e.g., 1023M)
  -H, --si              print sizes in powers of 1000 (e.g., 1.1G)
```

Using ‘-h’ will print the sizes in powers of 1024



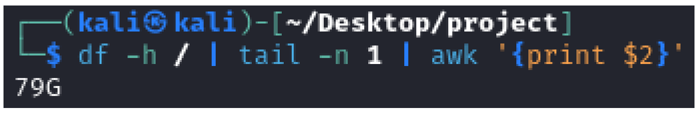
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ df -h /
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        79G   16G   60G  21% /
```

Using the commend “df -h /” in the terminal, it displayed the Size, Used and Avail space of the hard disk.



```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ df -h / | tail -n 1
/dev/sda1        79G   16G   60G  21% /
```

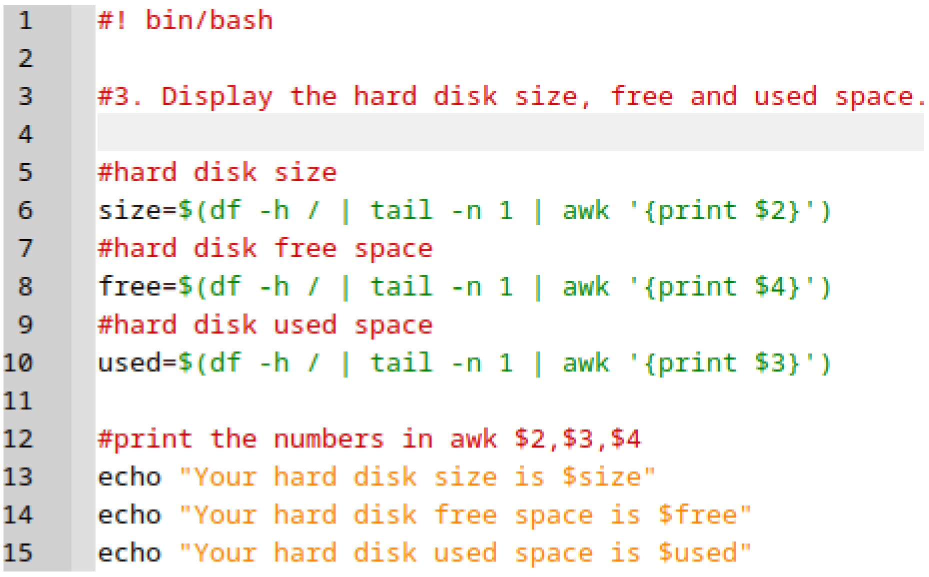
Using tail -n 1 to show the second row.



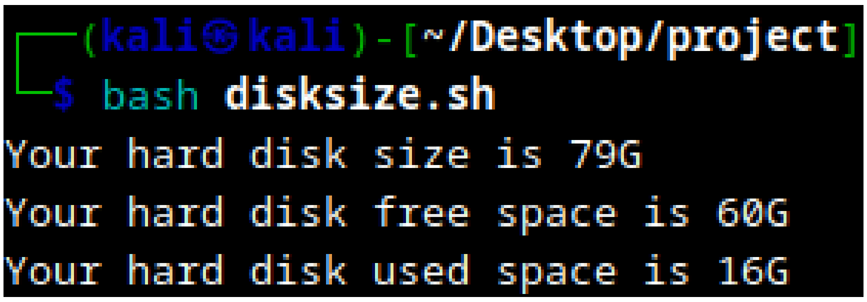
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ df -h / | tail -n 1 | awk '{print $2}'
79G
```

Using awk & print column 2, we were able to print the size.

Transferring to geany and do for column 3 & 4



```
#! bin/bash

#3. Display the hard disk size, free and used space.

#hard disk size
size=$(df -h / | tail -n 1 | awk '{print $2}')
#hard disk free space
free=$(df -h / | tail -n 1 | awk '{print $4}')
#hard disk used space
used=$(df -h / | tail -n 1 | awk '{print $3}')

#print the numbers in awk $2,$3,$4
echo "Your hard disk size is $size"
echo "Your hard disk free space is $free"
echo "Your hard disk used space is $used"
```



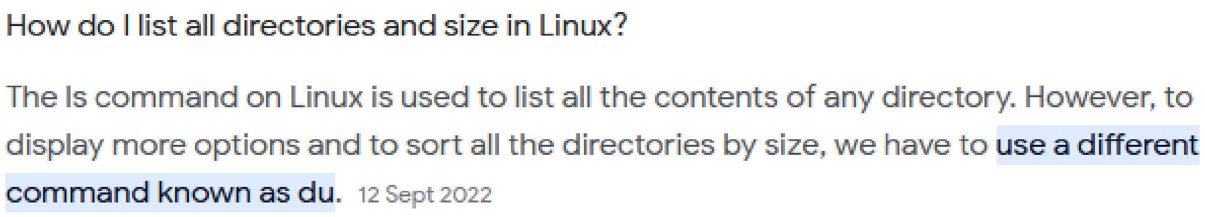
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ bash disksize.sh
Your hard disk size is 79G
Your hard disk free space is 60G
Your hard disk used space is 16G
```

The hard disk size is 79G

The hard disk free space is 60G

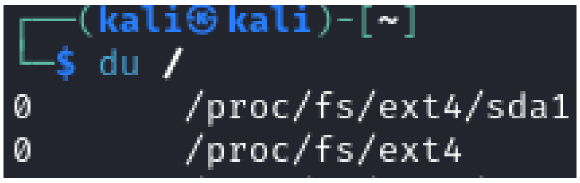
The hard disk used space is 16G

#4. Display the top five (5) directories and their size.

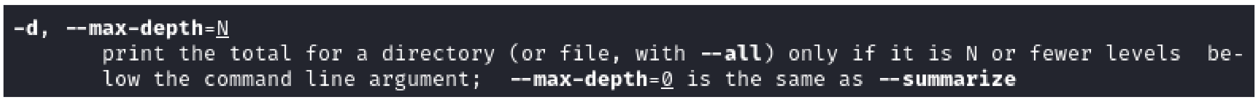

How do I list all directories and size in Linux?

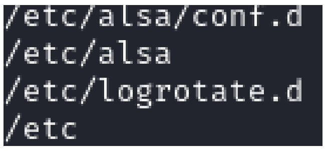
The ls command on Linux is used to list all the contents of any directory. However, to display more options and to sort all the directories by size, we have to use a different command known as du. 12 Sept 2022

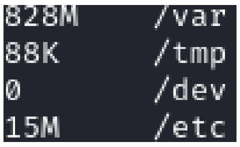
Googled and the command which is "du".


```
┌──(kali㉿kali)-[~]
└─$ du /
0       /proc/fs/ext4/sda1
0       /proc/fs/ext4
```

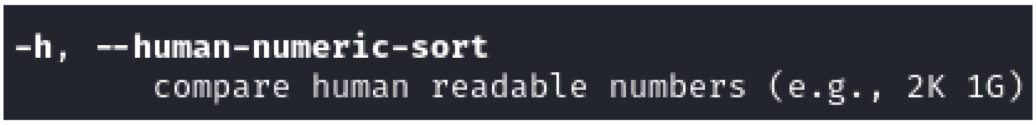
The command 'du' created a very long list a long list and error.


```
-d, --max-depth=N
       print the total for a directory (or file, with --all) only if it is N or fewer levels  be-
       low the command line argument;  --max-depth=0 is the same as --summarize
```

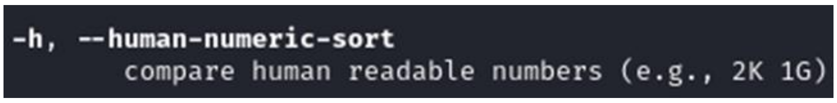

```
/etc/alsa/conf.d
/etc/alsa
/etc/logrotate.d
/etc
```

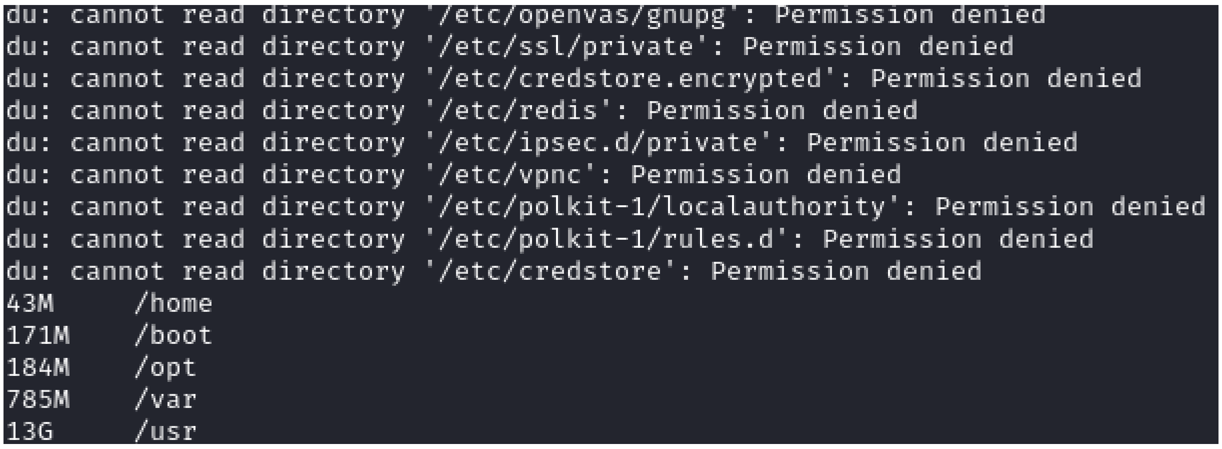

```
828M    /var
88K     /tmp
0       /dev
15M     /etc
```

Before -d 1 and after -d 1, the sub-directories excluded. This will not mess up the total calculation

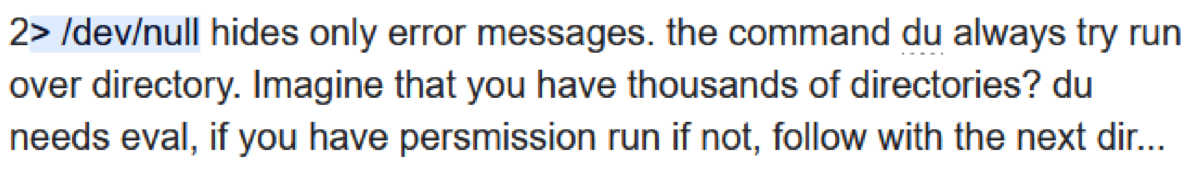

```
-h, --human-numeric-sort
       compare human readable numbers (e.g., 2K 1G)
```

Using -h to arranged according to the size.


```
-h, --human-numeric-sort
       compare human readable numbers (e.g., 2K 1G)
```

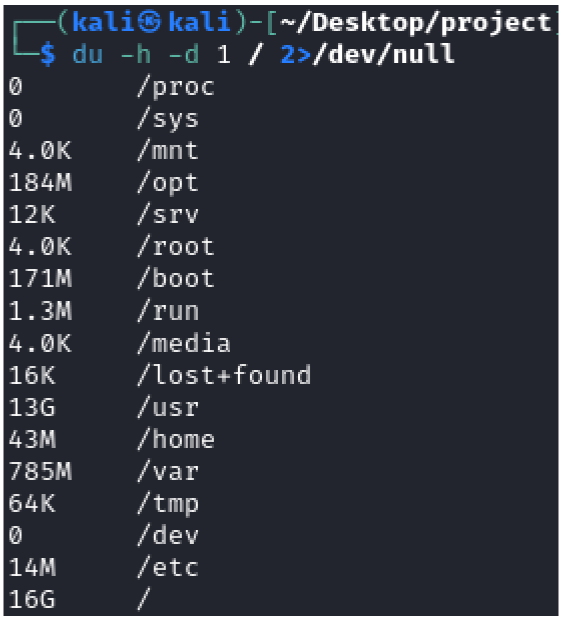

```
du: cannot read directory '/etc/openvas/gnupg': Permission denied
du: cannot read directory '/etc/ssl/private': Permission denied
du: cannot read directory '/etc/credstore.encrypted': Permission denied
du: cannot read directory '/etc/redis': Permission denied
du: cannot read directory '/etc/ipsec.d/private': Permission denied
du: cannot read directory '/etc/vpnc': Permission denied
du: cannot read directory '/etc/polkit-1/localauthority': Permission denied
du: cannot read directory '/etc/polkit-1/rules.d': Permission denied
du: cannot read directory '/etc/credstore': Permission denied
43M     /home
171M    /boot
184M    /opt
785M    /var
13G     /usr
```

Getting errors while trying to print the file and sizes.



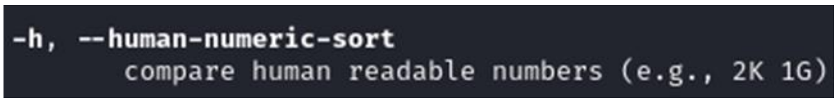
2> /dev/null hides only error messages. the command du always try run over directory. Imagine that you have thousands of directories? du needs eval, if you have persmission run if not, follow with the next dir...

Google online and found this command, to hide the errors.



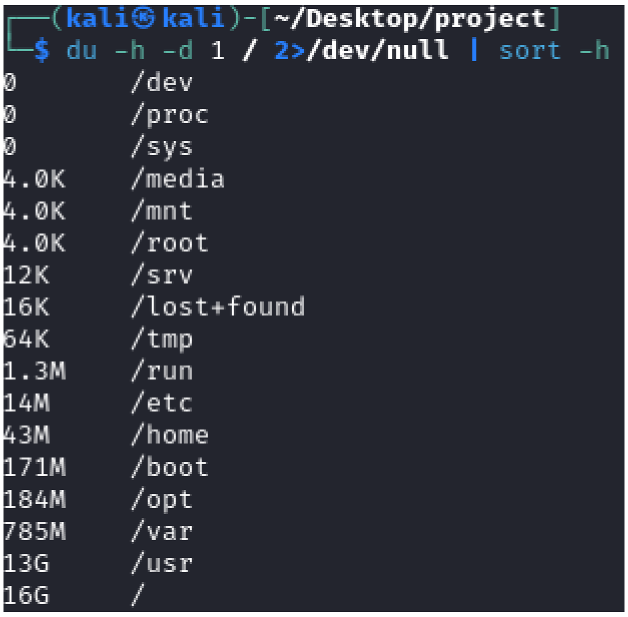
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ du -h -d 1 / 2>/dev/null
0       /proc
0       /sys
4.0K    /mnt
184M    /opt
12K     /srv
4.0K    /root
171M    /boot
1.3M    /run
4.0K    /media
16K     /lost+found
13G     /usr
43M     /home
785M    /var
64K     /tmp
0       /dev
14M     /etc
16G     /
```

No longer displaying errors. Now to sort and print them.



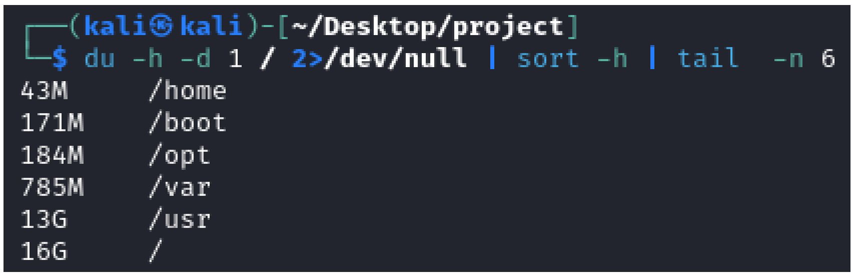
```
-h, --human-numeric-sort
       compare human readable numbers (e.g., 2K 1G)
```

sort -h is the same as above, so the number will be arranged accordingly.



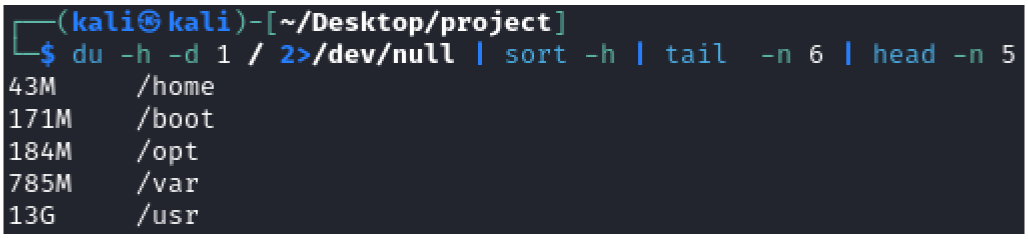
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ du -h -d 1 / 2>/dev/null | sort -h
0       /dev
0       /proc
0       /sys
4.0K    /media
4.0K    /mnt
4.0K    /root
12K     /srv
16K     /lost+found
64K     /tmp
1.3M    /run
14M     /etc
43M     /home
171M    /boot
184M    /opt
785M    /var
13G     /usr
16G     /
```

Sorted accordingly to size.



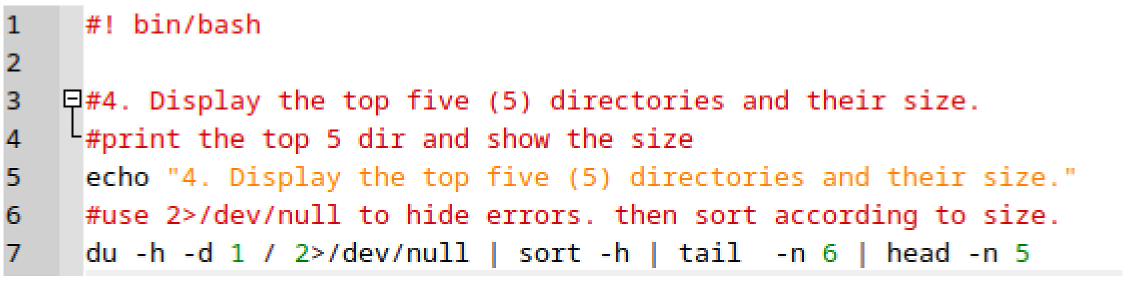
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ du -h -d 1 / 2>/dev/null | sort -h | tail  -n 6
43M     /home
171M    /boot
184M    /opt
785M    /var
13G     /usr
16G     /
```

After sorting, using tail to display the last 6, Directories and their size displayed along with root directories.



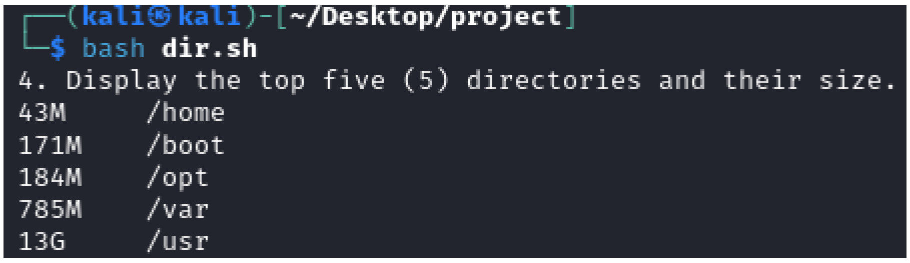
```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ du -h -d 1 / 2>/dev/null | sort -h | tail  -n 6 | head -n 5
43M     /home
171M    /boot
184M    /opt
785M    /var
13G     /usr
```

Added head -n 5 to remove the root directory.



```
#! bin/bash

#4. Display the top five (5) directories and their size.
#print the top 5 dir and show the size
echo "4. Display the top five (5) directories and their size."
#use 2>/dev/null to hide errors. then sort according to size.
du -h -d 1 / 2>/dev/null | sort -h | tail  -n 6 | head -n 5
```

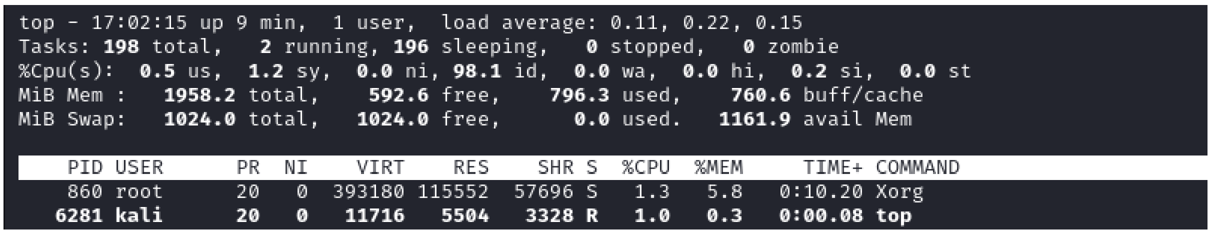
Transferred to geany.



```
┌──(kali㉿kali)-[~/Desktop/project]
└─$ bash dir.sh
4. Display the top five (5) directories and their size.
43M     /home
171M    /boot
184M    /opt
785M    /var
13G     /usr
```

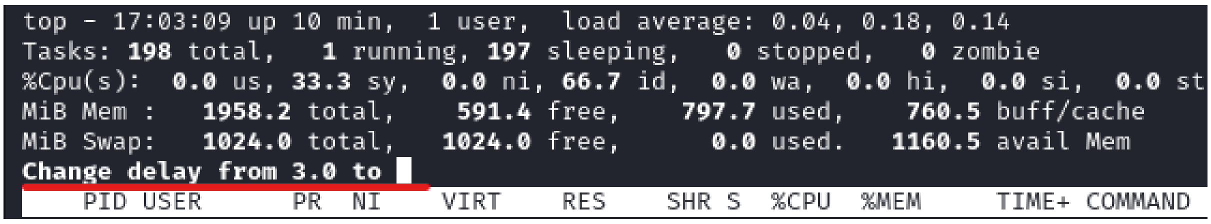
Ran the bash script and displayed the directories and the size.

#5. Display the CPU usage; refresh every 10 seconds.



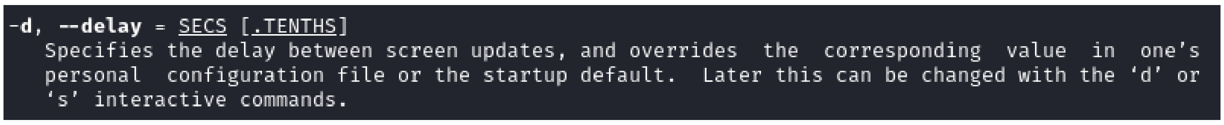
```
top - 17:02:15 up 9 min,  1 user,  load average: 0.11, 0.22, 0.15
Tasks: 198 total,   2 running, 196 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.5 us,  1.2 sy,  0.0 ni, 98.1 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   1958.2 total,    592.6 free,    796.3 used,    760.6 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1161.9 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    860 root      20   0  393180 115552  57696 S   1.3   5.8   0:10.20 Xorg
   6281 kali      20   0   11716   5504   3328 R   1.0   0.3   0:00.08 top
```

Tested the 'top' command in terminal.

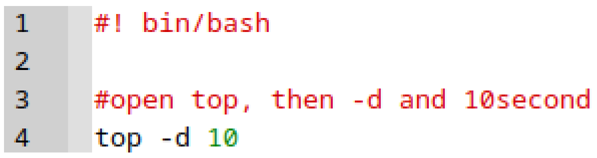


```
top - 17:03:09 up 10 min,  1 user,  load average: 0.04, 0.18, 0.14
Tasks: 198 total,   1 running, 197 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us, 33.3 sy,  0.0 ni, 66.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1958.2 total,    591.4 free,    797.7 used,    760.5 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1160.5 avail Mem
Change delay from 3.0 to
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
```

Press D then add 10 to add 10 second, q to exit top.

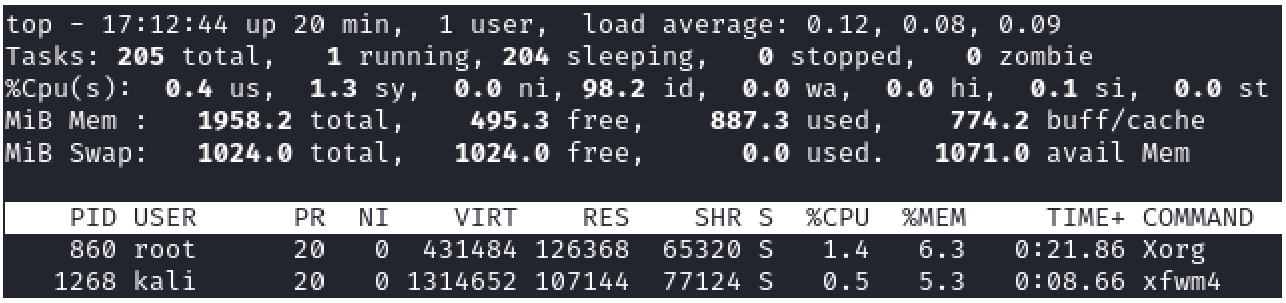


```
-d, --delay = SECS [.TENTHS]
   Specifies the delay between screen updates, and overrides  the  corresponding  value  in  one's
   personal  configuration file or the startup default.  Later this can be changed with the 'd' or
   's' interactive commands.
```



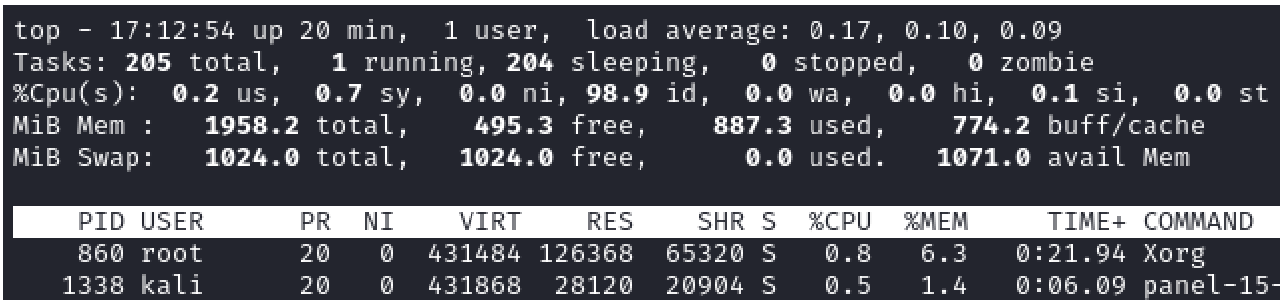
```
#! bin/bash

#open top, then -d and 10second
top -d 10
```

Copy over to geany.



```
top - 17:12:44 up 20 min,  1 user,  load average: 0.12, 0.08, 0.09
Tasks: 205 total,   1 running, 204 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.4 us,  1.3 sy,  0.0 ni, 98.2 id,  0.0 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :   1958.2 total,    495.3 free,    887.3 used,    774.2 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1071.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    860 root      20   0  431484 126368  65320 S   1.4   6.3   0:21.86 Xorg
   1268 kali      20   0 1314652 107144  77124 S   0.5   5.3   0:08.66 xfwm4
```

Running the bash task.sh



```
top - 17:12:54 up 20 min,  1 user,  load average: 0.17, 0.10, 0.09
Tasks: 205 total,   1 running, 204 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.7 sy,  0.0 ni, 98.9 id,  0.0 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :   1958.2 total,    495.3 free,    887.3 used,    774.2 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1071.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    860 root      20   0  431484 126368  65320 S   0.8   6.3   0:21.94 Xorg
   1338 kali      20   0  431868  28120  20904 S   0.5   1.4   0:06.09 panel-15-
```

"top" continue to run and print every 10 seconds.