

实验 07 进程作业与 shell 基础编程实验

班级：数据科学与大数据技术 2 班

学号：202026203039

姓名：赖丽婷

用户名：llt

一、实验目的

1. 练习进程与作业管理
2. 练习 shell 脚本编程

二、实验要求

1. 填写实验报告，请将关键命令及其结果进行截图(请确保截图中的文字清晰可见)
2. 导出为 pdf 文件，文件名为用户名-姓名-lab07.pdf，在规定截止时间之前上传作业)
3. 以下步骤中所有 s01 请换成你自己的用户名，01 请换成你自己用户名中的序号。

三、实验步骤

1. 进程管理

Ctrl +alt+f2 退出虚拟控制台

Ctrl + alt +f4 f3 f5 登录

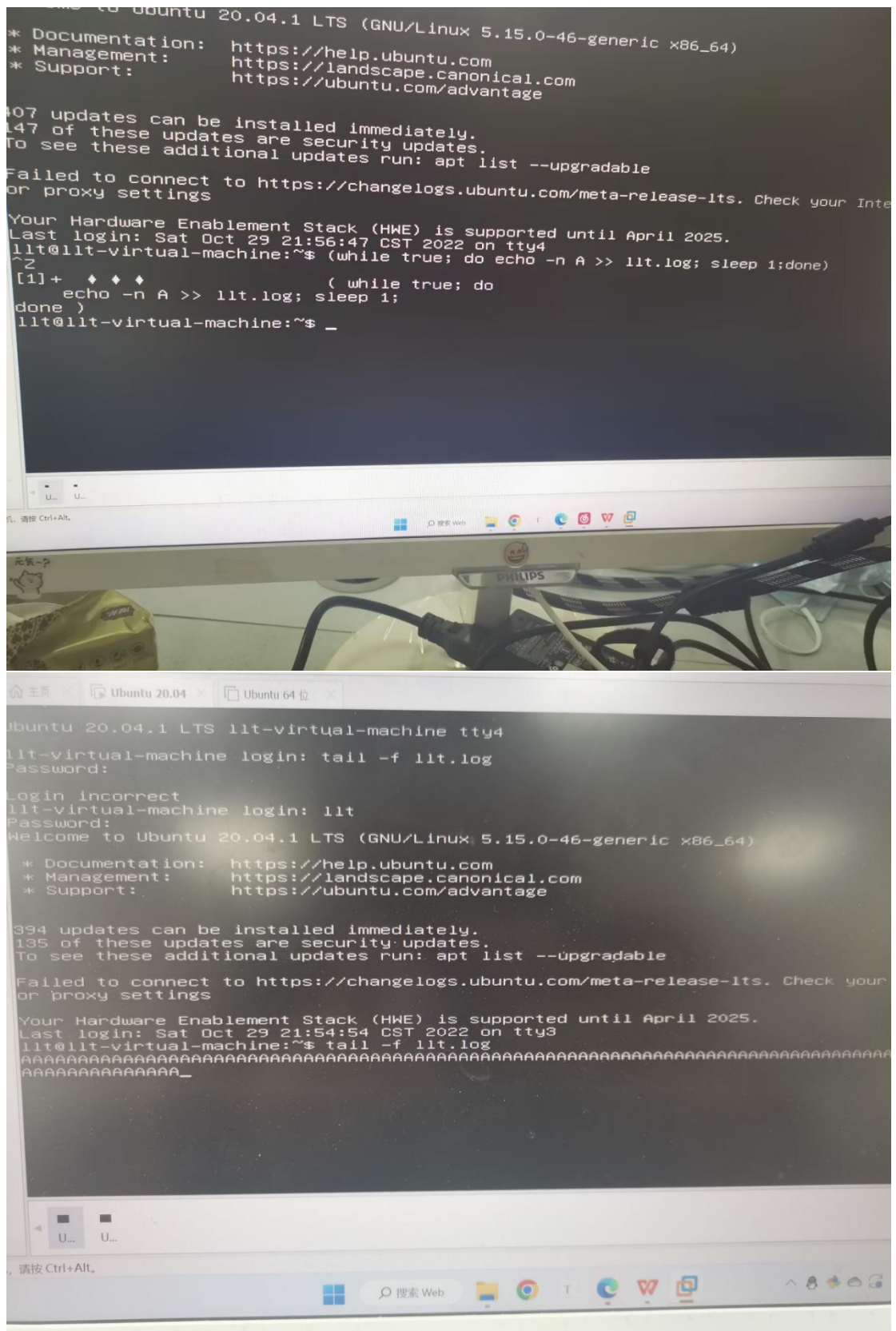
(1) 使用你的普通用户身份在本地虚拟控制台 tty2、tty3 上登陆，在 tty2 上运行以下命令：

```
(while true; do echo -n A >> s01.log; sleep 1;done)
```

在 tty3 上运行命令：

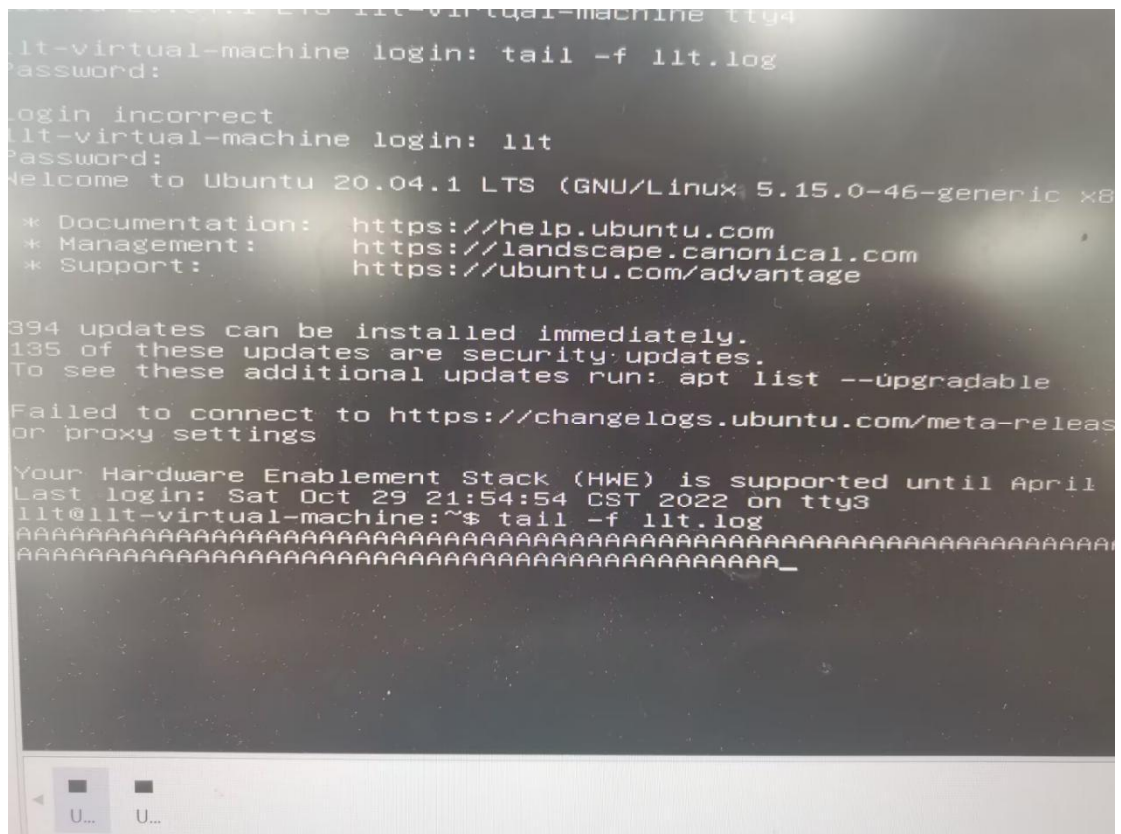
```
tail -f s01.log
```

请解释所看到的现象。



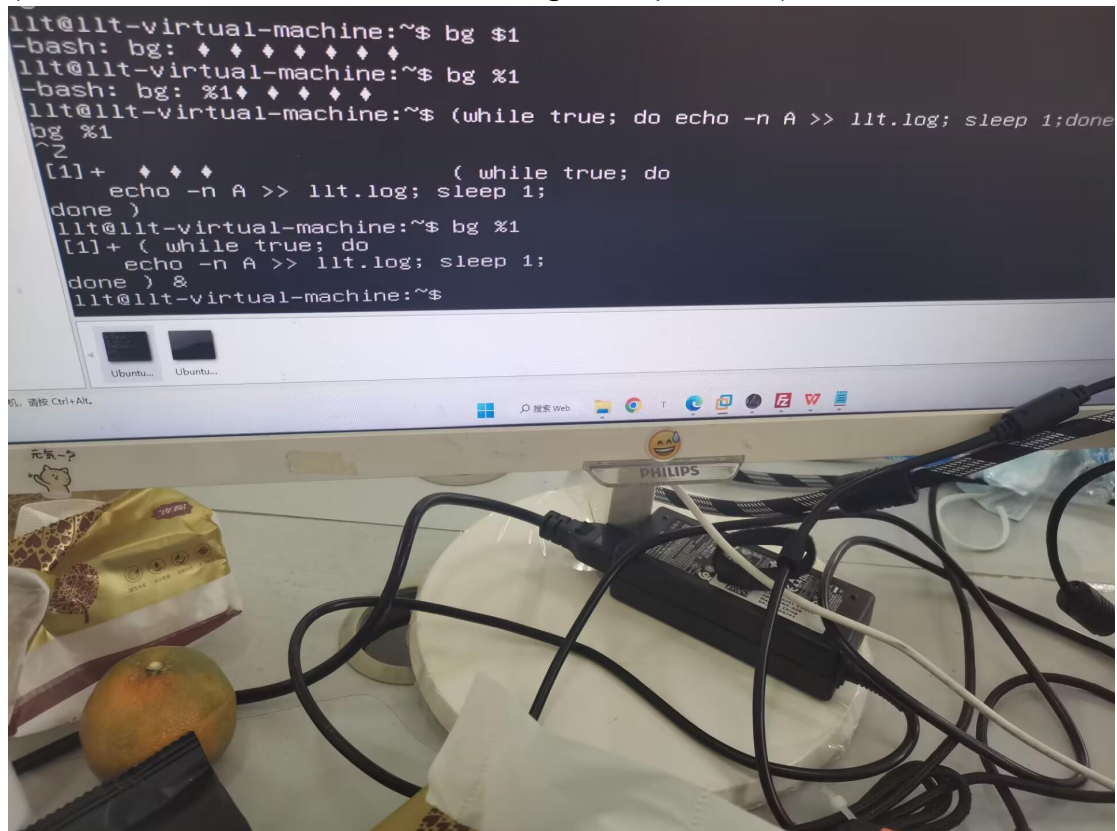
都是登录的同一个用户，所以在 **tty3** 开启进程 **tty4** 也能看到

(2) 切换回控制台 **tty2**，暂停当前进程的执行，并切换回控制台 **tty3**，查看 **tail** 命令的输出。



(3) 回到 `tty2`，将进程恢复到后台执行，然后执行以下命令：

```
(while true; do echo -n B >> s01.log; sleep 1;done) &  
(while true; do echo -n C >> s01.log; sleep 1;done) &
```

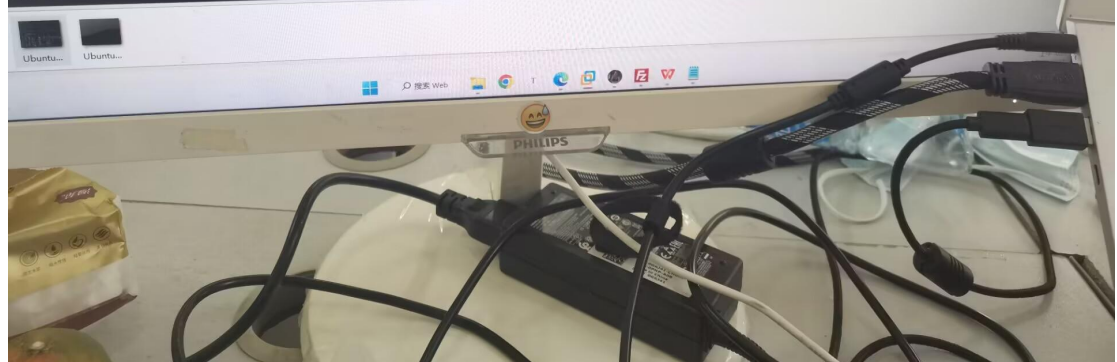


(4) 查看后台进程列表，切换至 `tty3`，查看 `tail` 命令的输出。


```

lt-virtual-machine:~$ bg %1
( while true; do
echo -n A >> llt.log; sleep 1;
) &
lt-virtual-machine:~$ (while true; do echo -n B >> llt.log; sleep 1;done) & (while true; do
22681 C >> llt.log; sleep 1;done) &
22682
lt-virtual-machine:~$ kill -STOP %2
lt-virtual-machine:~$ kill -CONT %2
lt-virtual-machine:~$ ps J
♦♦♦♦♦
lt-virtual-machine:~$ ps j
PPID    PID    PGID    SID    TTY        TPGID  STAT   UID    TIME  COMMAND
1291     1352    1352    1352    tty2        1352  Ssl+   1000    0:00  /usr/lib/gdm3/gdm-x-session --r
1352     1357    1352    1352    tty2        1352  Sl+    1000    0:10  /usr/lib/xorg/Xorg vt2 -display
1352     1414    1352    1352    tty2        1352  Sl+    1000    0:00  /usr/libexec/gnome-session-binar
19014    19148   19148   19014   tty3        24981  S      1000    0:00  -bash
19282    19425   19425   19282   tty4        20065  S      1000    0:00  -bash
19425    20065   20065   19282   tty4        20065  S+     1000    0:00  tail -f llt.log
19148    21082   21082   19014   tty3        24981  S      1000    0:00  -bash
19148    22681   22681   19014   tty3        24981  S      1000    0:00  -bash
19148    22682   22682   19014   tty3        24981  S      1000    0:00  -bash
21082    24968   21082   19014   tty3        24981  S      1000    0:00  sleep 1
22682    24969   22682   19014   tty3        24981  S      1000    0:00  sleep 1
22681    24979   22681   19014   tty3        24981  S      1000    0:00  ps j
19148    24981   24981   19014   tty3        24981  R+     1000    0:00  ps j
lt@lt-virtual-machine:~$ -

```



```

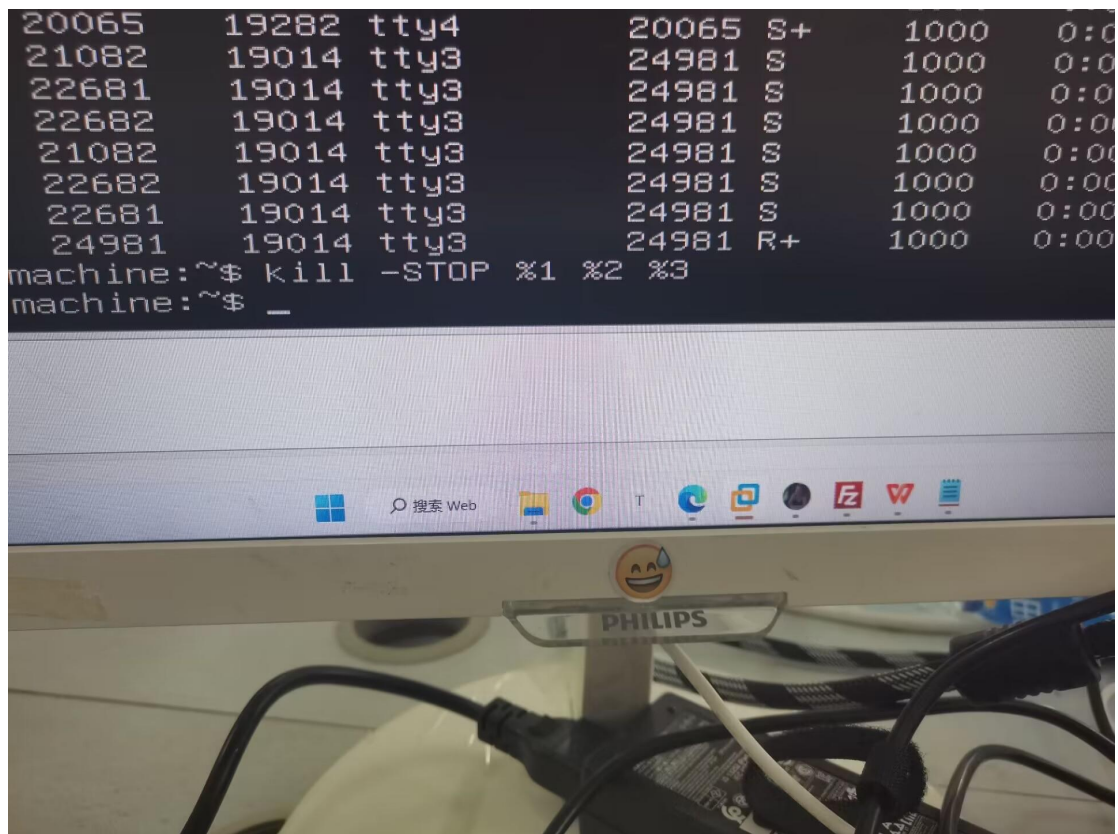
supported until April 2025.
on tty3
g
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
BCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBCBC
CCCCCCCCCAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
CABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABC
ACBACBACACACACACACACACACACACACACACACACACACACACACACACACACACACACACAC
ABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABCABC
BACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBAC
ACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBACBAC

```

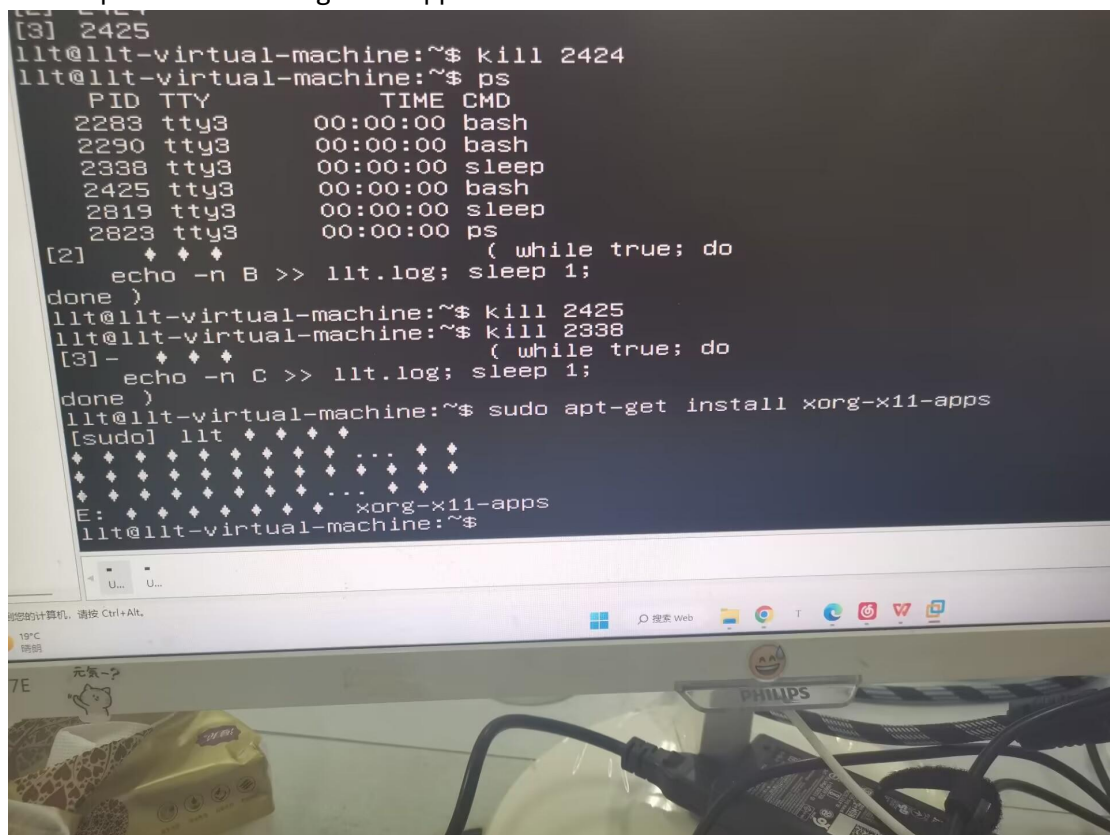


恢复了

(7) 用 kill 命令结束 tty2 上的所有三个进程，查看后台进程列表，并切换至 tty3 查看 tail 命令的输出是否停止。然后用快捷键结束 tail 命令的执行。



(8) 设置好虚拟机网络，使其能上网，然后以管理员身份安装 xorg-x11-apps 软件包。
 Sudo apt install xorg-x11-apps



(8) 从 MobaXterm 后台启动 4 次 o'clock 程序，令其 nice 值分别为 0, 10, 15, -5，然后用 ps 命令查看它们的 o'clock 值和优先级，执行什么操作时需要切换至 root 用户？

调为-5 需要 root 身份

(9) 用 renice 命令将 4 个 oclock 进程的 nice 值调整为 5, 执行什么操作时需要切换至 root 用户?

执行将高的 nice 值降低时需要切换至 root 用户

(11) 用 killall 命令结束所有 oclock 进程, 打印进程树, 同时显示各进程的 PID

```
graph LR
    xdg_document_po["xdg-document-po(1673)"] --- xdg_document_po_children["{xdg-document-po}(1674)  
{xdg-document-po}(1675)  
{xdg-document-po}(1682)  
{xdg-document-po}(1683)  
{xdg-document-po}(1684)"]
    xdg_document_po --- xdg_permission["xdg-permission-(1676)"]
    xdg_permission --- xdg_permission_children["{xdg-permission-}(1677)  
{xdg-permission-}(1679)"]
    systemd_journal["systemd-journal(277)"]
    systemd_logind["systemd-logind(592)"]
    systemd_resolve["systemd-resolve(499)"]
    systemd_timesyn["systemd-timesyn(500)"] --- systemd_timesyn_children["{systemd-timesyn}(543)"]
    systemd_udev["systemd-udev(306)"]
    udisksd["udiskd(598)"] --- udisksd_children["{udiskd}(604)  
{udiskd}(635)  
{udiskd}(667)  
{udiskd}(692)"]
    unattended_upgr["unattended-upgr(672)"] --- unattended_upgr_children["{unattended-upgr}(715)"]
    upowerd["upowerd(1075)"] --- upowerd_children["{upowerd}(1102)  
{upowerd}(1103)"]
    whoopsie["whoopsie(726)"] --- whoopsie_children["{whoopsie}(741)  
{whoopsie}(742)"]
    wpa_supplicant["wpa_supplicant(603)"]
```

[1]+ 已终止 nice --5 oclock

2. 编写脚本 s01-multi 打印九九乘法表如下:

```
1x1=1
2x1=2 2x2=4
3x1=3 3x2=6 3x3=9
.....
9x1=1 9x2=18 ..... 9x9=81
```

(请在下面贴出源代码)

```
#!/bin/bash
for j in {1..9}
do
for i in `seq $j`
do
echo -e -n "${i}*${j}=${i*j}\t"
done
echo
done
```

(请在下面贴出执行情况截图)


```
llt@llt-virtual-machine:~/7$ bash llt-multi
1*1=1
1*2=2      2*2=4
1*3=3      2*3=6      3*3=9
1*4=4      2*4=8      3*4=12    4*4=16
1*5=5      2*5=10     3*5=15     4*5=20     5*5=25
1*6=6      2*6=12     3*6=18     4*6=24     5*6=30     6*6=36
1*7=7      2*7=14     3*7=21     4*7=28     5*7=35     6*7=42     7*7=49
1*8=8      2*8=16     3*8=24     4*8=32     5*8=40     6*8=48     7*8=56     8*8=64
1*9=9      2*9=18     3*9=27     4*9=36     5*9=45     6*9=54     7*9=63     8*9=72     9*9=81
llt@llt-virtual-machine:~/7$
```

3. 编写脚本 **s01-primes**, 打印 n 以内所有的质数 (n 默认为 100)。
(请在下面贴出源代码)

```
#!/bin/bash
for i in `seq 100`
do
    for((j=2;j<=i-1;j++))
    do
        [ $((i%j)) -eq 0 ] && break
    done
    [ $j -eq $i ] && echo $i
done
```

"llt-primes" 9L, 125C

(请在下面贴出执行情况截图)

```
llt@llt-virtual-machine:~/7$ bash llt-primes
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
llt@llt-virtual-machine:~/7$
```

4. 编写脚本 `s01-numbers`, 提示用户输入一个整数, 脚本将分别输出该整数每个位的数字, 并输出这些数字的和, 例如, 输出整数 1234 每个位上的 1 2 3 4 以及 10, 输出整数 -5678 每个位上的 5 6 7 8 以及 26。

(请在下面贴出源代码)

```
llt@llt-virtual-machine: ~/7
#!/bin/bash
echo 'input a number'
read m
if [ $m -lt 0 ]
then
    let "m=-$m"
fi
sum=0
while [ $m -ne 0 ]
do
    let "g=$m%10"
    let "m=$m/10"
    let "sum=$sum+$g"
done
echo $sum
~
~
~
~
~
~
~
~
~
~
-- 插入 --
```

:

(请在下面贴出执行情况截图)

```
llt@llt-virtual-machine: ~/7$ vim llt-numbers
llt@llt-virtual-machine:~/7$ bash llt-numbers
input a number
1234
10
llt@llt-virtual-machine:~/7$ bash llt-numbers
input a number
-5678
26
llt@llt-virtual-machine:~/7$
```

5. 编写 `s01-toss` 脚本模拟抛掷硬币实验 n 次(默认为 10 次), 每次抛掷硬币 m 次(默认为 1000 次), 例如 `toss 5 500`, 并统计和打印出每次实验中正面和反面出现的总次数。

(请在下面贴出源代码)

```
llt@llt-virtual-machine: ~/7
#!/bin/bash
echo "input a n:"
read n
echo "input a m:"
read m
count1=0
count2=0
for i in `seq 1 $n`
do
    for j in $(seq 1 $m)
    do
        if [ $((RANDOM%2)) -eq 0 ]
        then
            let "count1=$count1+1"
        else
            let "count2=$count2+1"
        fi
    done
    echo "第 $i 次实验"
    echo "正面 $count1 次"
    echo "反面 $count2 次"
    let "count1=0"
    let "count2=0"
done
~
~
~
~
```

(请在下面贴出执行情况截图)

```
第 1 次实验
正面 510 次
反面 490 次
第 2 次实验
正面 484 次
反面 516 次
第 3 次实验
正面 485 次
反面 515 次
第 4 次实验
正面 489 次
反面 511 次
第 5 次实验
正面 518 次
反面 482 次
第 6 次实验
正面 476 次
反面 524 次
第 7 次实验
正面 498 次
反面 502 次
第 8 次实验
正面 500 次
反面 500 次
第 9 次实验
正面 504 次
反面 496 次
第 10 次实验
正面 490 次
反面 510 次
```