天津大學

# 《操作系统原理》实验报告

进程管理

|  |  |
|---|---|
| 学　　号 | 3020205015 |
| 姓　　名 | 石云天 |
| 学　　院 | 未来技术学院 |
| 专　　业 | 计算机科学与技术 |
| 年　　级 | 2020 |
| 任课教师 | 李春 |

2023 年 4 月 1 日

## 一、实验内容与实现

### 1. 1 Exercise1 Shell basics

**What this exercise is about**

This exercise provides an opportunity to get to know the basic features of the Linux shell (Bash).

**What you should be able to do**

At the end of the exercise, you should be able to:

- Use wildcards for file name expansion
- Redirect standard in, standard out, and standard error
- Use pipes to provide the output of one process as input to another process
- Perform command grouping and line continuation

**Introduction**

In this exercise, you will be exploring the common file editor, Vi.

**Requirements**

- This workbook
- A workstation with Fedora, RHEL, or SLES installed

**Exercise instructions**

Preface

- All exercises in this chapter depend on the availability of specific equipment in your classroom.

**Wildcards**

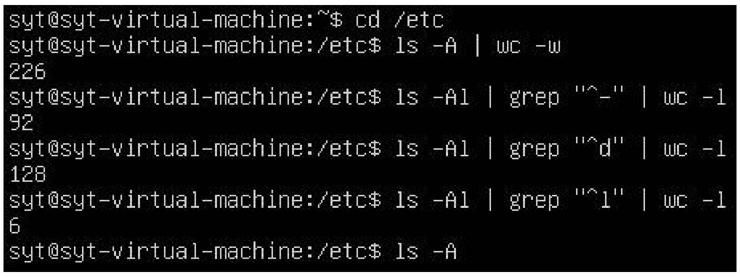
1.If you are not logged in as tux1 at tty1, log in now.

使用 `Ctrl+Alt+F1` 指令进入 tty1。

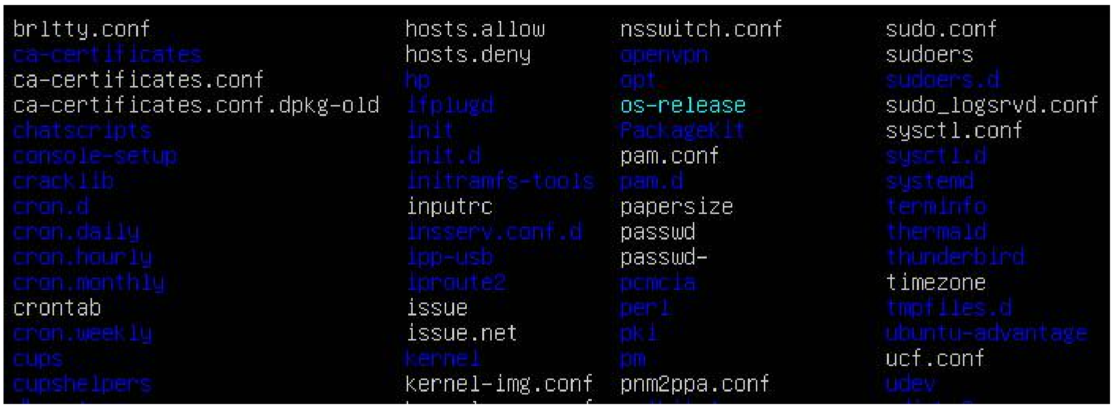
2.Go to the /etc directory and make a list of all files here.

使用 `ls -A` 命令查看当前目录下的所有文件，包括隐藏文件，但不包括当前目录与父目录，接下来使用 `ls -A | wc -w` 命令进行计数，发现/etc 文件夹下共 226 个文件，其中有 92 个文件，128 个目录，6 个链接文件。（结果未完全

展示）



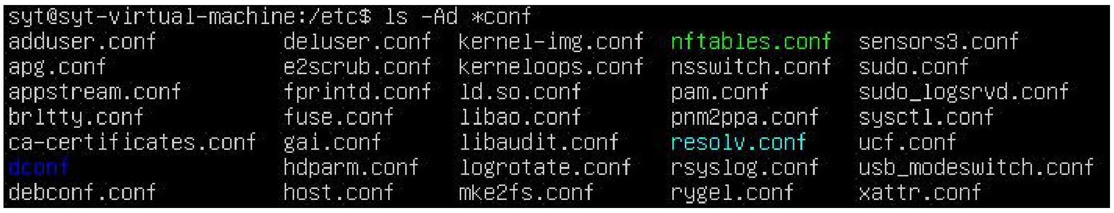
```
syt@syt-virtual-machine:~$ cd /etc
syt@syt-virtual-machine:/etc$ ls -A | wc -w
226
syt@syt-virtual-machine:/etc$ ls -Al | grep "^-" | wc -l
92
syt@syt-virtual-machine:/etc$ ls -Al | grep "^d" | wc -l
128
syt@syt-virtual-machine:/etc$ ls -Al | grep "^l" | wc -l
6
syt@syt-virtual-machine:/etc$ ls -A
```



```
brltty.conf                  hosts.allow       nsswitch.conf       sudo.conf
ca-certificates              hosts.deny        openvpn             sudoers
ca-certificates.conf         hp                opt                 sudoers.d
ca-certificates.conf.dpkg-old ifplugd          os-release          sudo_logsrvd.conf
chatscripts                  init              PackageKit          sysctl.conf
console-setup                init.d            pam.conf            sysctl.d
cracklib                     initramfs-tools   pam.d               systemd
cron.d                       inputrc           papersize           terminfo
cron.daily                   insserv.conf.d    passwd              thermald
cron.hourly                  ipp-usb           passwd-             thunderbird
cron.monthly                 iproute2          pcmcia              timezone
crontab                      issue             perl                tmpfiles.d
cron.weekly                  issue.net         pki                 ubuntu-advantage
cups                         kernel            pm                  ucf.conf
cupshelpers                  kernel-img.conf   pnm2ppa.conf        udev
```

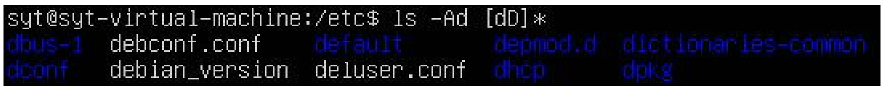
3.Use ls with wildcards to list file names:

a.That end with conf



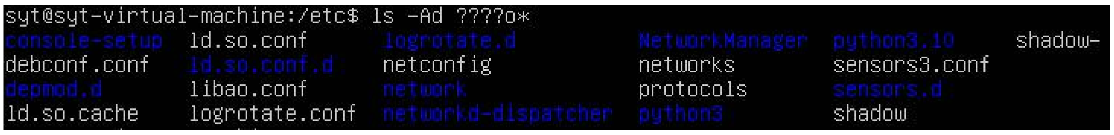
```
syt@syt-virtual-machine:/etc$ ls -Ad *conf
adduser.conf          deluser.conf   kernel-img.conf   nftables.conf   sensors3.conf
apg.conf              e2scrub.conf   kerneloops.conf   nsswitch.conf   sudo.conf
appstream.conf        fprintd.conf   ld.so.conf        pam.conf        sudo_logsrvd.conf
brltty.conf           fuse.conf      libao.conf        pnm2ppa.conf    sysctl.conf
ca-certificates.conf  gai.conf       libaudit.conf     resolv.conf     ucf.conf
dconf                 hdparm.conf    logrotate.conf    rsyslog.conf    usb_modeswitch.conf
debconf.conf          host.conf      mke2fs.conf       rygel.conf      xattr.conf
```

b.That begin with a d or D



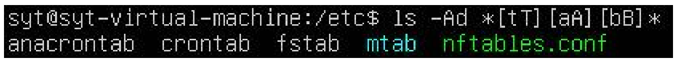
```
syt@syt-virtual-machine:/etc$ ls -Ad [dD]*
dbus-1  debconf.conf    default       depmod.d  dictionaries-common
dconf   debian_version  deluser.conf  dhcp      dpkg
```

c.That contain an o in the fifth position



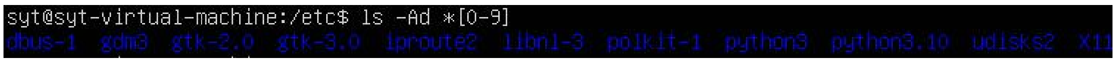
```
syt@syt-virtual-machine:/etc$ ls -Ad ????o*
console-setup  ld.so.conf    logrotate.d          NetworkManager  python3.10     shadow-
debconf.conf   ld.so.conf.d  netconfig            networks        sensors3.conf
depmod.d       libao.conf    network              protocols       sensors.d
ld.so.cache    logrotate.conf networkd-dispatcher python3         shadow
```

d.That contain the word tab (in any combination with capitals and lowercase characters)



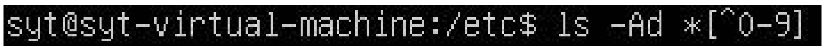
```
syt@syt-virtual-machine:/etc$ ls -Ad *[tT][aA][bB]*
anacrontab  crontab  fstab  mtab  nftables.conf
```

e.That end with a number

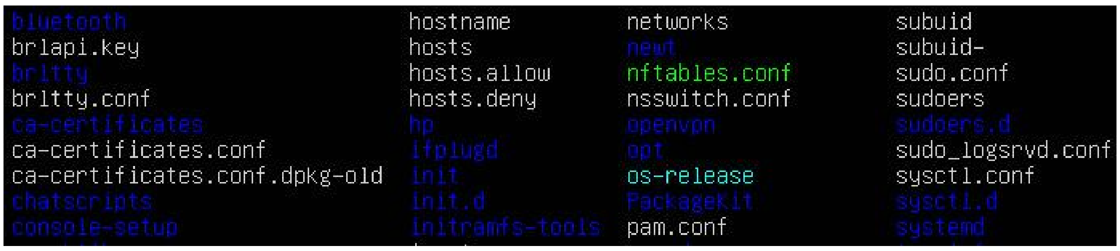


```
syt@syt-virtual-machine:/etc$ ls -Ad *[0-9]
dbus-1  gdm3  gtk-2.0  gtk-3.0  iproute2  libnl-3  polkit-1  python3  python3.10  udisks2  X11
```

f.That do not end with a number



```
syt@syt-virtual-machine:/etc$ ls -Ad *[^0-9]
```

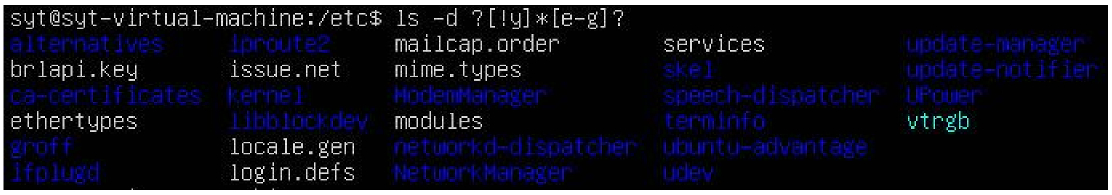


```
bluetooth                      hostname          networks          subuid
brlapi.key                     hosts             newt              subuid-
brltty                         hosts.allow       nftables.conf     sudo.conf
brltty.conf                    hosts.deny        nsswitch.conf     sudoers
ca-certificates                hp                openvpn           sudoers.d
ca-certificates.conf           ifplugd           opt               sudo_logsrvd.conf
ca-certificates.conf.dpkg-old  init              os-release        sysctl.conf
chatscripts                    init.d            PackageKit        sysctl.d
console-setup                  initramfs-tools   pam.conf          systemd
```

结果未完全展示

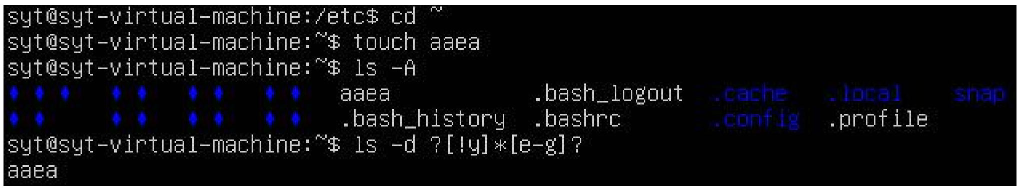
(Note that wildcard expansion is done by the shell. If one of the file names that matches is a directory name, then ls by default lists the contents of that directory instead of the file name itself. To prevent this, use the -d option.)

4.What happens if you execute the command ls -d ?[!y]*[e-g]? What would the shortest file name be that can match? Execute this command to verify your answer.



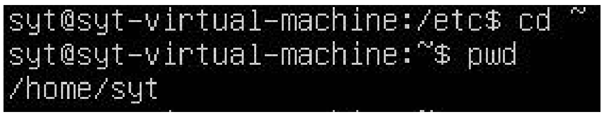
```
syt@syt-virtual-machine:/etc$ ls -d ?[!y]*[e-g]?
alternatives     iproute2     mailcap.order        services           update-manager
brlapi.key       issue.net    mime.types           skel               update-notifier
ca-certificates  kernel       ModemManager         speech-dispatcher  UPower
ethertypes       libblockdev  modules              terminfo           vtrgb
groff            locale.gen   networkd-dispatcher  ubuntu-advantage
ifplugd          login.defs   NetworkManager       udev
```

最短可匹配的文件名长度为 4，第一个字符任意，第二个字符不是 y，中间*可以匹配 0-n 个字符，倒数第二个字符为 efg 中之一，最后一个字符任意，故最少需要四个字符，按字母顺序最短的文件名为 aaea。为进行验证，需要创建名为 aaea 的文件，但由于没有/etc 目录下的写权限，故需要先返回家目录，再进行创建，最后进行验证。



```
syt@syt-virtual-machine:/etc$ cd ~
syt@syt-virtual-machine:~$ touch aaea
syt@syt-virtual-machine:~$ ls -A
♦ ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦   aaea            .bash_logout  .cache   .local    snap
♦ ♦     ♦ ♦   ♦ ♦   ♦ ♦   .bash_history   .bashrc       .config  .profile
syt@syt-virtual-machine:~$ ls -d ?[!y]*[e-g]?
aaea
```

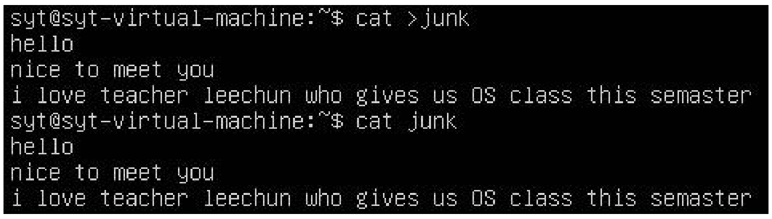
5.Return to your home directory.



```
syt@syt-virtual-machine:/etc$ cd ~
syt@syt-virtual-machine:~$ pwd
/home/syt
```

## Redirection

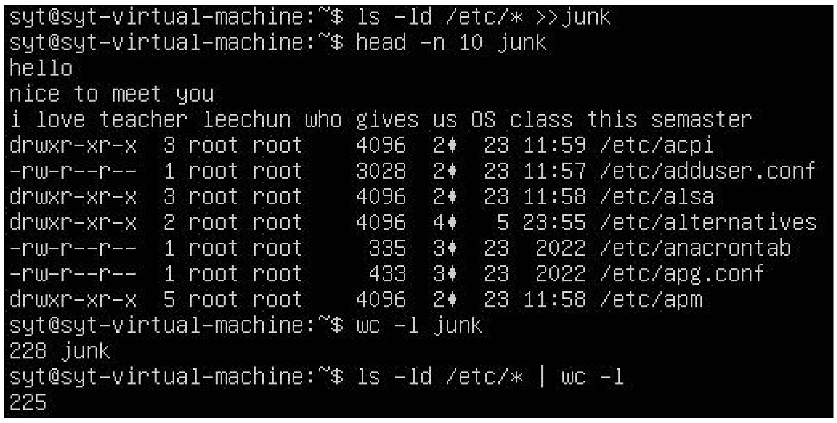
6.Use the cat command and redirection to create a file called junk containing a few lines of text. When you have typed a few lines, end your input to the cat command and return to the shell prompt. Then view the contents of the file you just created.



```
syt@syt-virtual-machine:~$ cat >junk
hello
nice to meet you
i love teacher leechun who gives us OS class this semaster
syt@syt-virtual-machine:~$ cat junk
hello
nice to meet you
i love teacher leechun who gives us OS class this semaster
```

在输入结束后利用 `ctrl+d` 指令结束文本输入，返回 shell。

7.Append more lines to the junk file using redirection. Then view the contents of the file junk and check if all the lines you saved in this file are there.



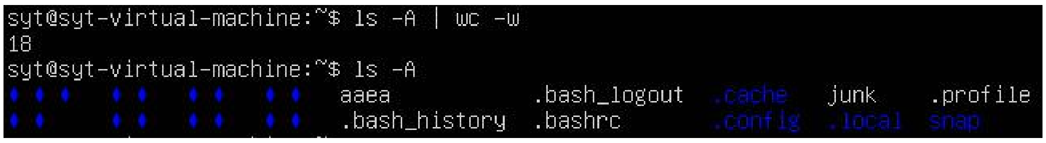
```
syt@syt-virtual-machine:~$ ls -ld /etc/* >>junk
syt@syt-virtual-machine:~$ head -n 10 junk
hello
nice to meet you
i love teacher leechun who gives us OS class this semaster
drwxr-xr-x  3 root root    4096  2♦  23 11:59 /etc/acpi
-rw-r--r--  1 root root    3028  2♦  23 11:57 /etc/adduser.conf
drwxr-xr-x  3 root root    4096  2♦  23 11:58 /etc/alsa
drwxr-xr-x  2 root root    4096  4♦   5 23:55 /etc/alternatives
-rw-r--r--  1 root root     335  3♦  23  2022 /etc/anacrontab
-rw-r--r--  1 root root     433  3♦  23  2022 /etc/apg.conf
drwxr-xr-x  5 root root    4096  2♦  23 11:58 /etc/apm
syt@syt-virtual-machine:~$ wc -l junk
228 junk
syt@syt-virtual-machine:~$ ls -ld /etc/* | wc -l
225
```

将/etc 目录下的非隐藏文件的具体信息添加到 junk 文件中，使用 `ls -ld /etc/* >> junk` 命令，查看 junk 文件前十行，判断是否在原文件下新增行，而非替换源文件内容，最后根据行数进行校验，junk 文件原有 3 行，新增 225 行，最终 junk 文件下有 228 行，通过校验。

**Pipes, tees, and filters**

8.Count the number of files in your current directory. Use a pipe. Do not count the files manually.

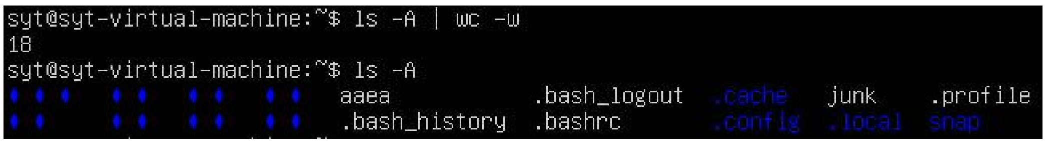
使用命令 `ls -A | wc -w` 计算所有文件，包括隐藏文件并进行校验。



```
syt@syt-virtual-machine:~$ ls -A | wc -w
18
syt@syt-virtual-machine:~$ ls -A
♦ ♦ ♦    ♦ ♦    ♦ ♦    ♦ ♦    aaea           .bash_logout  .cache   junk    .profile
♦ ♦      ♦ ♦    ♦ ♦    ♦ ♦    .bash_history  .bashrc       .config  .local  snap
```

9.Does ls > tempfile ; wc -l tempfile ; rm tempfile do the same thing

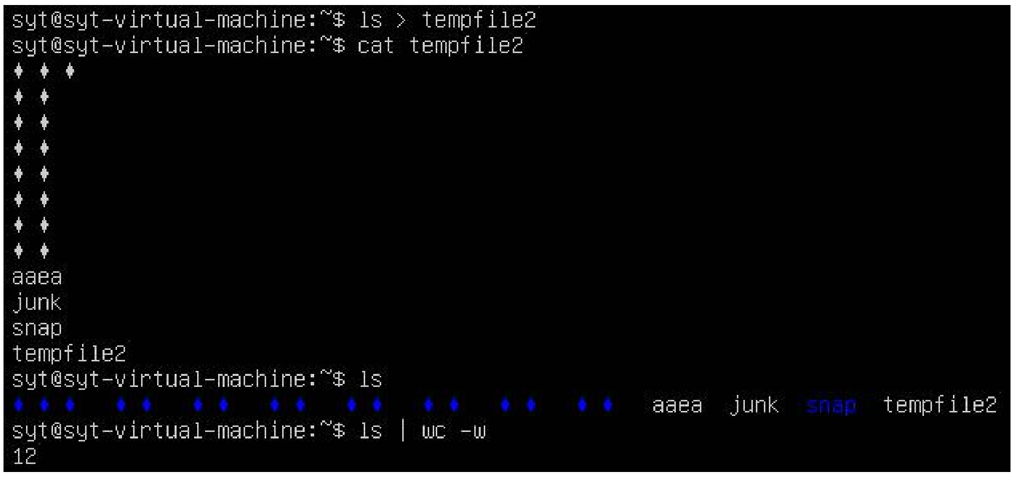
as the pipe you made in the previous command? Why or why not?



```
syt@syt-virtual-machine:~$ ls -A | wc -w
18
syt@syt-virtual-machine:~$ ls -A
♦ ♦ ♦    ♦ ♦    ♦ ♦    ♦ ♦    aaea           .bash_logout  .cache   junk    .profile
♦ ♦      ♦ ♦    ♦ ♦    ♦ ♦    .bash_history  .bashrc       .config  .local  snap
```

从结果上看，该命令也能统计 tempfile 下所有文件数量，但其并未使用管道，而是将所有文件名写入一个新的文件，统计该文件行数得出结果，最后删除该文件。此过程的运行速度与效率远高于使用管道，管道将前一部分的运行结果进行第二部分处理，优势在于无需创建新的文件。

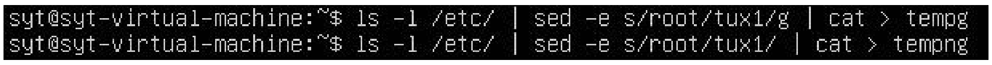
10.Use the ls command and save the output in a file called tempfile2 before you count the files.



```
syt@syt-virtual-machine:~$ ls > tempfile2
syt@syt-virtual-machine:~$ cat tempfile2
♦ ♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
aaea
junk
snap
tempfile2
syt@syt-virtual-machine:~$ ls
♦ ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦   ♦ ♦    aaea  junk  snap  tempfile2
syt@syt-virtual-machine:~$ ls | wc -w
12
```

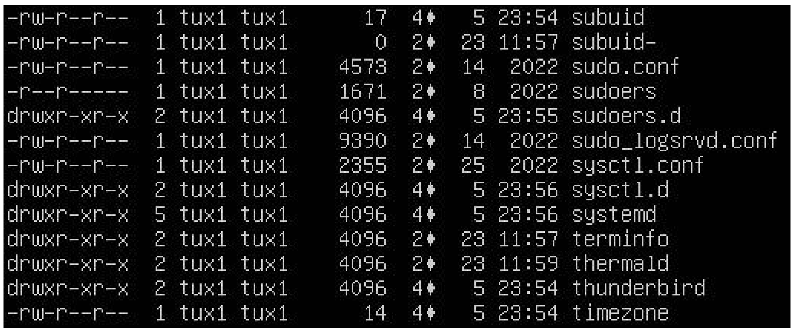
11.Use the sed command to alter the output of the ls -l /etc/ command so that it looks like you own all files in /etc. Execute this both with and without the global option. What is the difference?

首先执行如下两条指令，将使用了 global 的结果存入文件 tempg 中，将未使用 global 的结果存入文件 tempng 中，由于 sed 指令一行一行读取，不使用 global 选项，则只会替换每行第一个匹配项，而使用 global 选项会替换所有匹配项。



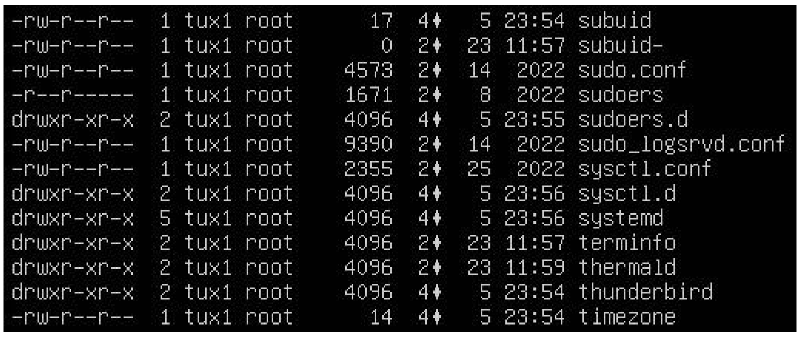
```
syt@syt-virtual-machine:~$ ls -l /etc/ | sed -e s/root/tux1/g | cat > tempg
syt@syt-virtual-machine:~$ ls -l /etc/ | sed -e s/root/tux1/ | cat > tempng
```

使用 global 选项所得结果如下（未完全显示）：



```
-rw-r--r--  1 tux1 tux1     17  4♦   5 23:54 subuid
-rw-r--r--  1 tux1 tux1      0  2♦  23 11:57 subuid-
-rw-r--r--  1 tux1 tux1   4573  2♦  14  2022 sudo.conf
-r--r-----  1 tux1 tux1   1671  2♦   8  2022 sudoers
drwxr-xr-x  2 tux1 tux1   4096  4♦   5 23:55 sudoers.d
-rw-r--r--  1 tux1 tux1   9390  2♦  14  2022 sudo_logsrvd.conf
-rw-r--r--  1 tux1 tux1   2355  2♦  25  2022 sysctl.conf
drwxr-xr-x  2 tux1 tux1   4096  4♦   5 23:56 sysctl.d
drwxr-xr-x  5 tux1 tux1   4096  4♦   5 23:56 systemd
drwxr-xr-x  2 tux1 tux1   4096  2♦  23 11:57 terminfo
drwxr-xr-x  2 tux1 tux1   4096  2♦  23 11:59 thermald
drwxr-xr-x  2 tux1 tux1   4096  4♦   5 23:54 thunderbird
-rw-r--r--  1 tux1 tux1     14  4♦   5 23:54 timezone
```

不使用 global 选项所得结果如下（未完全显示）：

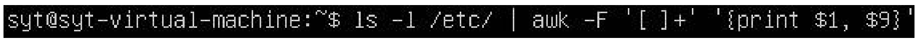


```
-rw-r--r--  1 tux1 root     17  4♦   5 23:54 subuid
-rw-r--r--  1 tux1 root      0  2♦  23 11:57 subuid-
-rw-r--r--  1 tux1 root   4573  2♦  14  2022 sudo.conf
-r--r-----  1 tux1 root   1671  2♦   8  2022 sudoers
drwxr-xr-x  2 tux1 root   4096  4♦   5 23:55 sudoers.d
-rw-r--r--  1 tux1 root   9390  2♦  14  2022 sudo_logsrvd.conf
-rw-r--r--  1 tux1 root   2355  2♦  25  2022 sysctl.conf
drwxr-xr-x  2 tux1 root   4096  4♦   5 23:56 sysctl.d
drwxr-xr-x  5 tux1 root   4096  4♦   5 23:56 systemd
drwxr-xr-x  2 tux1 root   4096  2♦  23 11:57 terminfo
drwxr-xr-x  2 tux1 root   4096  2♦  23 11:59 thermald
drwxr-xr-x  2 tux1 root   4096  4♦   5 23:54 thunderbird
-rw-r--r--  1 tux1 root     14  4♦   5 23:54 timezone
```

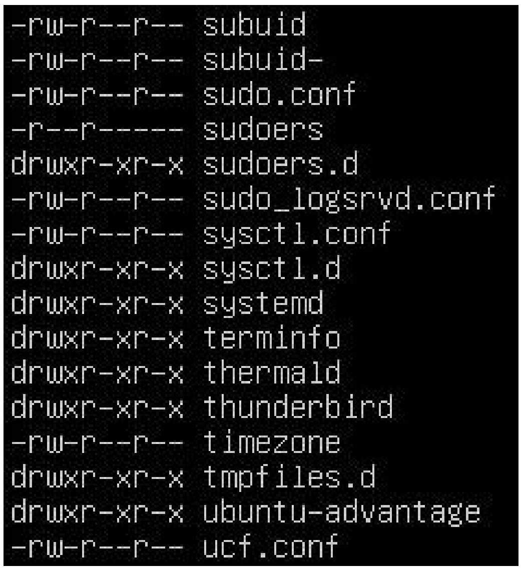
12.Use the awk command to display the first and ninth column of the output of the ls

-l /etc/ command.

结果如下（未完全展示）：



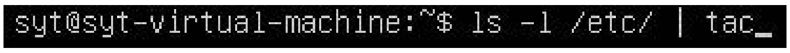
```
syt@syt-virtual-machine:~$ ls -l /etc/ | awk -F '[ ]+' '{print $1, $9}'
```



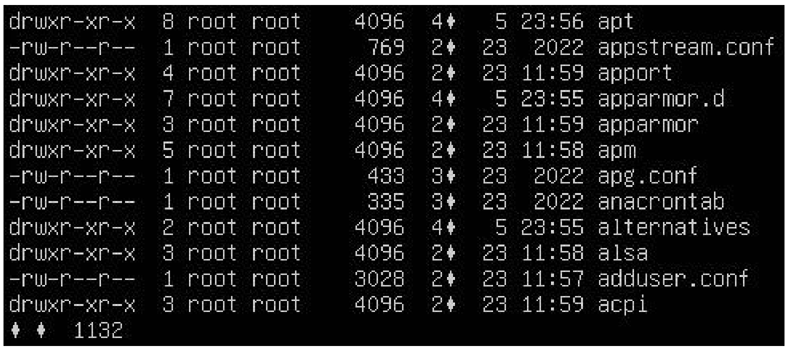
```
-rw-r--r-- subuid
-rw-r--r-- subuid-
-rw-r--r-- sudo.conf
-r--r----- sudoers
drwxr-xr-x sudoers.d
-rw-r--r-- sudo_logsrvd.conf
-rw-r--r-- sysctl.conf
drwxr-xr-x sysctl.d
drwxr-xr-x systemd
drwxr-xr-x terminfo
drwxr-xr-x thermald
drwxr-xr-x thunderbird
-rw-r--r-- timezone
drwxr-xr-x tmpfiles.d
drwxr-xr-x ubuntu-advantage
-rw-r--r-- ucf.conf
```

13.Use the tac command to display the output of the ls command in reverse order.

结果如下（未完全展示）：

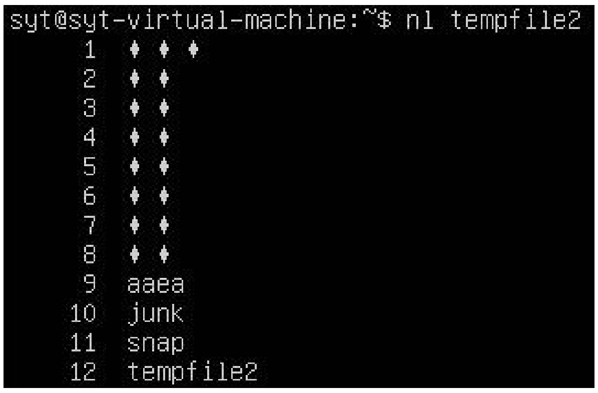


```
syt@syt-virtual-machine:~$ ls -l /etc/ | tac_
```



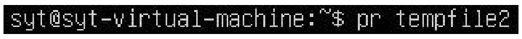
```
drwxr-xr-x  8 root root    4096  4♦   5 23:56 apt
-rw-r--r--  1 root root     769  2♦  23  2022 appstream.conf
drwxr-xr-x  4 root root    4096  2♦  23 11:59 apport
drwxr-xr-x  7 root root    4096  4♦   5 23:55 apparmor.d
drwxr-xr-x  3 root root    4096  2♦  23 11:59 apparmor
drwxr-xr-x  5 root root    4096  2♦  23 11:58 apm
-rw-r--r--  1 root root     433  3♦  23  2022 apg.conf
-rw-r--r--  1 root root     335  3♦  23  2022 anacrontab
drwxr-xr-x  2 root root    4096  4♦   5 23:55 alternatives
drwxr-xr-x  3 root root    4096  2♦  23 11:58 alsa
-rw-r--r--  1 root root    3028  2♦  23 11:57 adduser.conf
drwxr-xr-x  3 root root    4096  2♦  23 11:59 acpi
♦ ♦  1132
```

14.Use the nl command to number the lines of tempfile2.

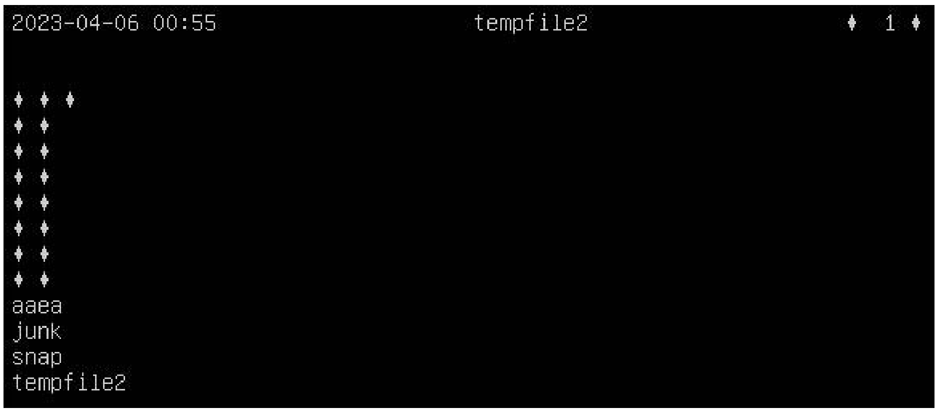


```
syt@syt-virtual-machine:~$ nl tempfile2
     1  ♦ ♦ ♦
     2  ♦ ♦
     3  ♦ ♦
     4  ♦ ♦
     5  ♦ ♦
     6  ♦ ♦
     7  ♦ ♦
     8  ♦ ♦
     9  aaea
    10  junk
    11  snap
    12  tempfile2
```

15.Use the pr command to format tempfile2 for the printer.



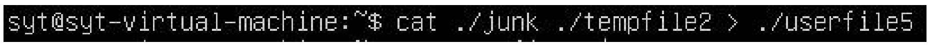
```
syt@syt-virtual-machine:~$ pr tempfile2
```



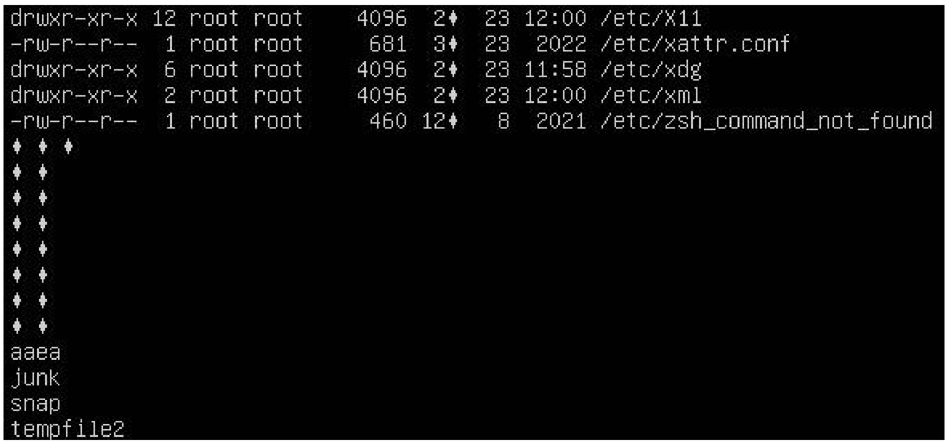
```
2023-04-06 00:55                    tempfile2                    ♦  1 ♦


♦ ♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
aaea
junk
snap
tempfile2
```

16.Combine all usersfile parts from the file and directory permissions exercise into one big file, called usersfile5. Check to see if this file is identical to the original usersfile.

将 junk 文件与 tempfile5 文件拼接在一起，重命名为 userfile5，两者完全相同。



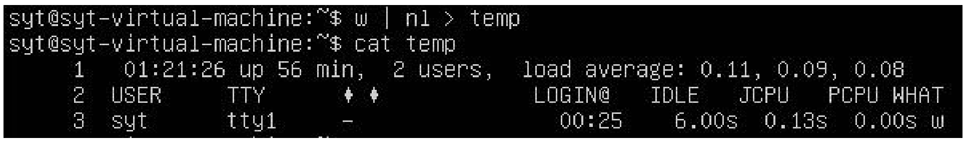
```
syt@syt-virtual-machine:~$ cat ./junk ./tempfile2 > ./userfile5
```



```
drwxr-xr-x 12 root root     4096  2♦  23 12:00 /etc/X11
-rw-r--r--  1 root root      681  3♦  23  2022 /etc/xattr.conf
drwxr-xr-x  6 root root     4096  2♦  23 11:58 /etc/xdg
drwxr-xr-x  2 root root     4096  2♦  23 12:00 /etc/xml
-rw-r--r--  1 root root      460 12♦   8  2021 /etc/zsh_command_not_found
♦ ♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
♦ ♦
aaea
junk
snap
tempfile2
```

**Command grouping**

17.On the same command line, display the current system date and all the users that are logged in, together with some explaining comments, and save all this to one file after numbering the lines. Check your output.

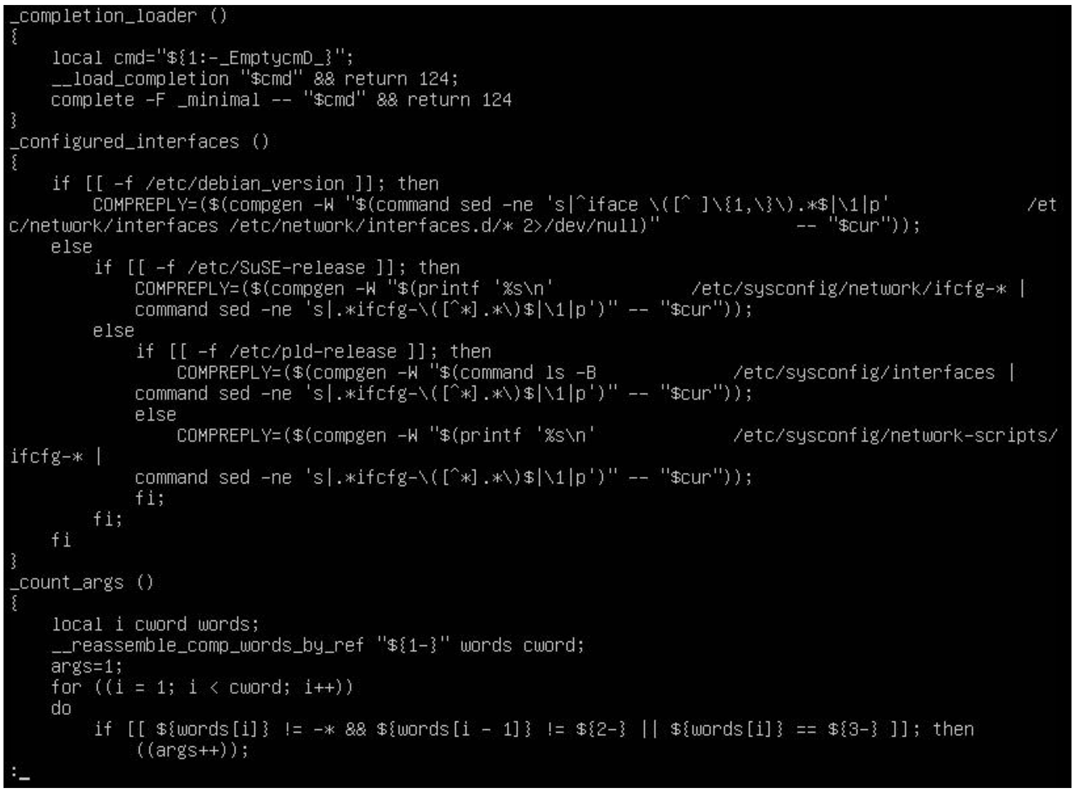


```
syt@syt-virtual-machine:~$ w | nl > temp
syt@syt-virtual-machine:~$ cat temp
     1   01:21:26 up 56 min,  2 users,  load average: 0.11, 0.09, 0.08
     2  USER     TTY      ♦ ♦             LOGIN@   IDLE   JCPU   PCPU WHAT
     3  syt      tty1     -                00:25   6.00s  0.13s  0.00s w
```

**Process environment**

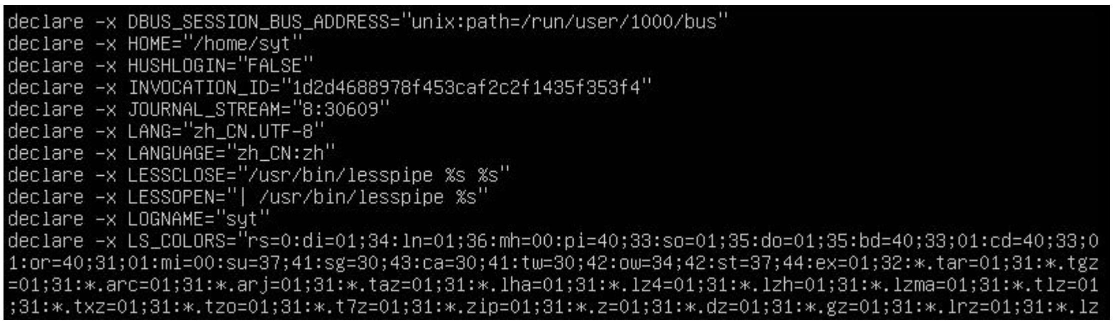
18.Display all your variables that are defined in your current process environment. Also
display all variables that are currently exported.

首先利用 set 指令展示已定义的所有变量（未完全显示）：



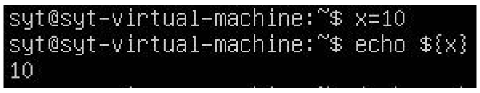
```
_completion_loader ()
{
    local cmd="${1:-_EmptycmD_}";
    __load_completion "$cmd" && return 124;
    complete -F _minimal -- "$cmd" && return 124
}
_configured_interfaces ()
{
    if [[ -f /etc/debian_version ]]; then
        COMPREPLY=($(compgen -W "$(command sed -ne 's|^iface \([^ ]\{1,\}\).*$|\1|p'             /et
c/network/interfaces /etc/network/interfaces.d/* 2>/dev/null)"             -- "$cur"));
    else
        if [[ -f /etc/SuSE-release ]]; then
            COMPREPLY=($(compgen -W "$(printf '%s\n'             /etc/sysconfig/network/ifcfg-* |
            command sed -ne 's|.*ifcfg-\([^*].*\)$|\1|p')" -- "$cur"));
        else
            if [[ -f /etc/pld-release ]]; then
                COMPREPLY=($(compgen -W "$(command ls -B             /etc/sysconfig/interfaces |
            command sed -ne 's|.*ifcfg-\([^*].*\)$|\1|p')" -- "$cur"));
            else
                COMPREPLY=($(compgen -W "$(printf '%s\n'             /etc/sysconfig/network-scripts/
ifcfg-* |
            command sed -ne 's|.*ifcfg-\([^*].*\)$|\1|p')" -- "$cur"));
            fi;
        fi;
    fi
}
_count_args ()
{
    local i cword words;
    __reassemble_comp_words_by_ref "${1-}" words cword;
    args=1;
    for ((i = 1; i < cword; i++))
    do
        if [[ ${words[i]} != -* && ${words[i - 1]} != ${2-} || ${words[i]} == ${3-} ]]; then
            ((args++));
:_
```

再利用 `export -p` 指令展示已 export 的所有变量（未完全显示）：



```
declare -x DBUS_SESSION_BUS_ADDRESS="unix:path=/run/user/1000/bus"
declare -x HOME="/home/syt"
declare -x HUSHLOGIN="FALSE"
declare -x INVOCATION_ID="1d2d4688978f453caf2c2f1435f353f4"
declare -x JOURNAL_STREAM="8:30609"
declare -x LANG="zh_CN.UTF-8"
declare -x LANGUAGE="zh_CN:zh"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="syt"
declare -x LS_COLORS="rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;0
1:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz
=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01
;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz
```

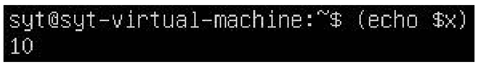
19.Create a variable x and set its value to 10. Check the value of the variable. Again, display all your current variables.



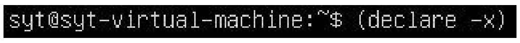
```
syt@syt-virtual-machine:~$ x=10
syt@syt-virtual-machine:~$ echo ${x}
10
```

20.Create a subshell. Check to see what value variable x holds in the subshell. What is the value of x? List the subshell's current variables. Do you see a listing for x?

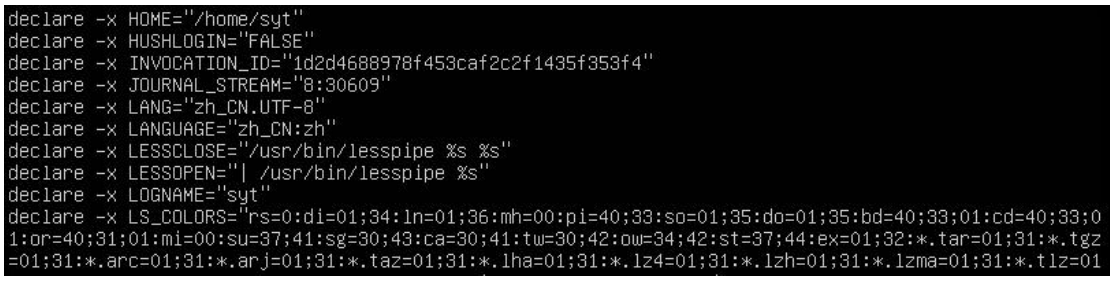
在子进程中，x 仍为 10：



```
syt@syt-virtual-machine:~$ (echo $x)
10
```



```
syt@syt-virtual-machine:~$ (declare -x)
```

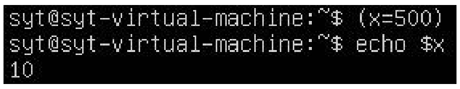


```
declare -x HOME="/home/syt"
declare -x HUSHLOGIN="FALSE"
declare -x INVOCATION_ID="1d2d4688978f453caf2c2f1435f353f4"
declare -x JOURNAL_STREAM="8:30609"
declare -x LANG="zh_CN.UTF-8"
declare -x LANGUAGE="zh_CN:zh"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="syt"
declare -x LS_COLORS="rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;0
1:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz
=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01
```

从结果中未看到 x 的列表。

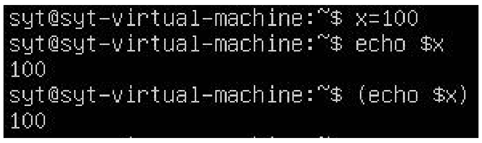
21.Set the value of x to 500 and go back to your parent process. What is the current value of x? Why?

x 仍为 10，因为子进程会继承父进程的所有变量，因此子进程中存在父进程定义的变量，但子进程创建后便独立于父进程运行，因此子进程中定义的变量不会影响到父进程。



```
syt@syt-virtual-machine:~$ (x=500)
syt@syt-virtual-machine:~$ echo $x
10
```

22.Make sure that child processes inherit the variable x. Verify this by creating a subshell and checking the value of variable x. After this, exit your subshell.



```
syt@syt-virtual-machine:~$ x=100
syt@syt-virtual-machine:~$ echo $x
100
syt@syt-virtual-machine:~$ (echo $x)
100
```

## 1. 2 Exercise2 Working with processes

### What this exercise is about

This exercise familiarizes you with process manipulation and process control.

### What you should be able to do

At the end of the exercise, you should be able to:

- Monitor processes
- Change and understand the process environment
- Control jobs
- Terminate processes

### Introduction

In this exercise, you will be executing and manipulating Linux processes.

### Requirements

• This workbook
• A workstation with Fedora, RHEL, or SLES installed

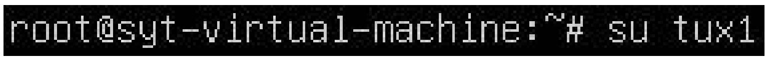
**Exercise instructions**

Preface

• All exercises in this chapter depend on the availability of specific equipment in your classroom.

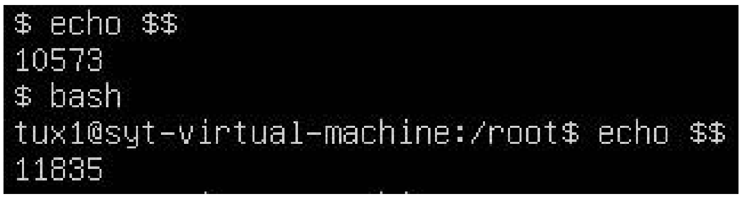
**Listing processes**

1.Log in at tty1 as tux1.



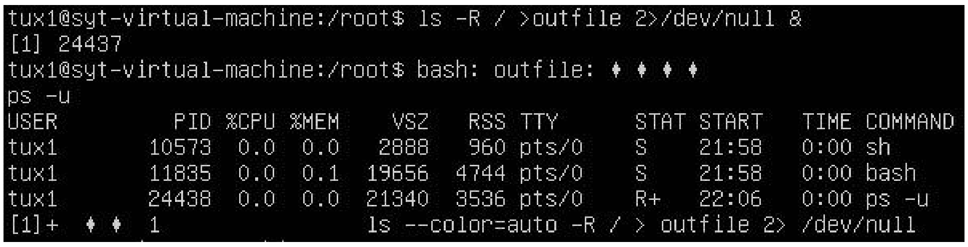
```
root@syt-virtual-machine:~# su tux1
```

2.Check the PID of your log in environment and then create a subshell by entering bash. What is the process ID of the subshell? Is it different from your login process?



```
$ echo $$
10573
$ bash
tux1@syt-virtual-machine:/root$ echo $$
11835
```

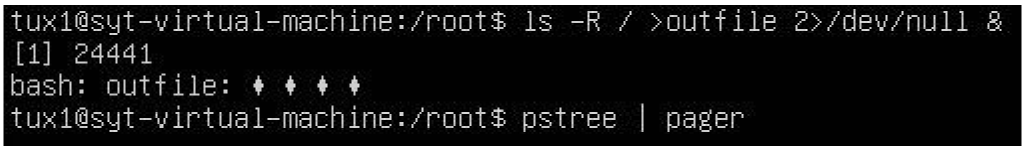
3.Enter the command ls -R / >outfile 2>/dev/null & and then show the processes that you are running in the system. Which processes are running?



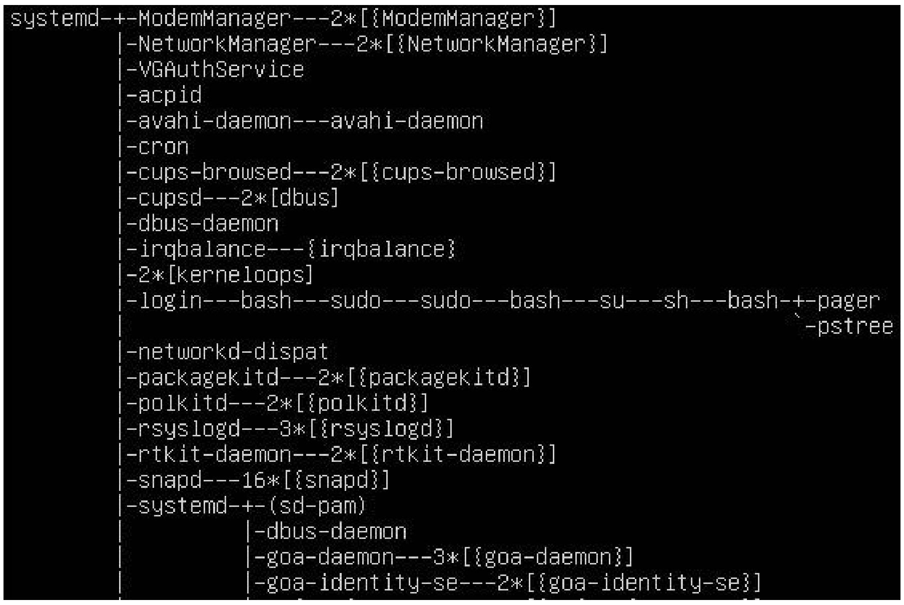
```
tux1@syt-virtual-machine:/root$ ls -R / >outfile 2>/dev/null &
[1] 24437
tux1@syt-virtual-machine:/root$ bash: outfile: ♦ ♦ ♦ ♦
ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
tux1       10573  0.0  0.0   2888   960 pts/0    S    21:58   0:00 sh
tux1       11835  0.0  0.1  19656  4744 pts/0    S    21:58   0:00 bash
tux1       24438  0.0  0.0  21340  3536 pts/0    R+   22:06   0:00 ps -u
[1]+  ♦ ♦  1                ls --color=auto -R / > outfile 2> /dev/null
```

Note: This command is explained in full in the next units.

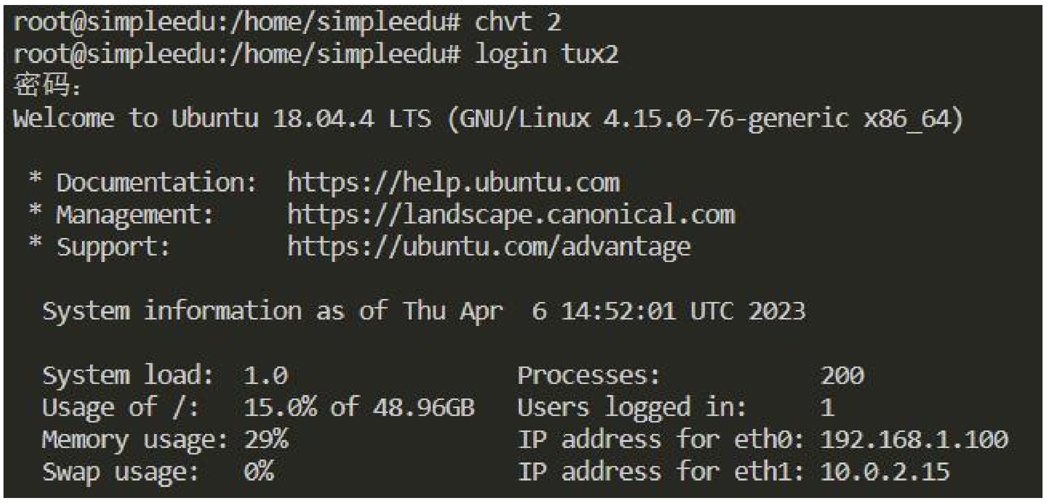
4.While the ls command is still running, run the pstree command. (It might be necessary to restart the ls command.)

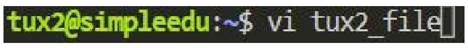


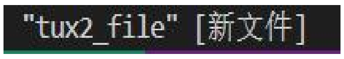
```
tux1@syt-virtual-machine:/root$ ls -R / >outfile 2>/dev/null &
[1] 24441
bash: outfile: ♦ ♦ ♦ ♦
tux1@syt-virtual-machine:/root$ pstree | pager
```


```
systemd-+-ModemManager---2*[{ModemManager}]
        |-NetworkManager---2*[{NetworkManager}]
        |-VGAuthService
        |-acpid
        |-avahi-daemon---avahi-daemon
        |-cron
        |-cups-browsed---2*[{cups-browsed}]
        |-cupsd---2*[dbus]
        |-dbus-daemon
        |-irqbalance---{irqbalance}
        |-2*[kerneloops]
        |-login---bash---sudo---sudo---bash---su---sh---bash-+-pager
        |                                                    `-pstree
        |-networkd-dispat
        |-packagekitd---2*[{packagekitd}]
        |-polkitd---2*[{polkitd}]
        |-rsyslogd---3*[{rsyslogd}]
        |-rtkit-daemon---2*[{rtkit-daemon}]
        |-snapd---16*[{snapd}]
        |-systemd-+-(sd-pam)
        |         |-dbus-daemon
        |         |-goa-daemon---3*[{goa-daemon}]
        |         |-goa-identity-se---2*[{goa-identity-se}]
```

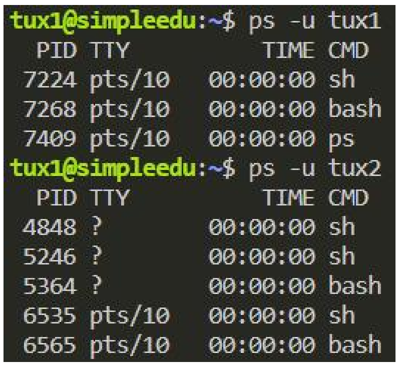
5.Log in as tux2 on tty2 and run vi tux2_file.


```
root@simpleedu:/home/simpleedu# chvt 2
root@simpleedu:/home/simpleedu# login tux2
密码:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Apr  6 14:52:01 UTC 2023

  System load:  1.0               Processes:           200
  Usage of /:   15.0% of 48.96GB  Users logged in:     1
  Memory usage: 29%               IP address for eth0: 192.168.1.100
  Swap usage:   0%                IP address for eth1: 10.0.2.15
```

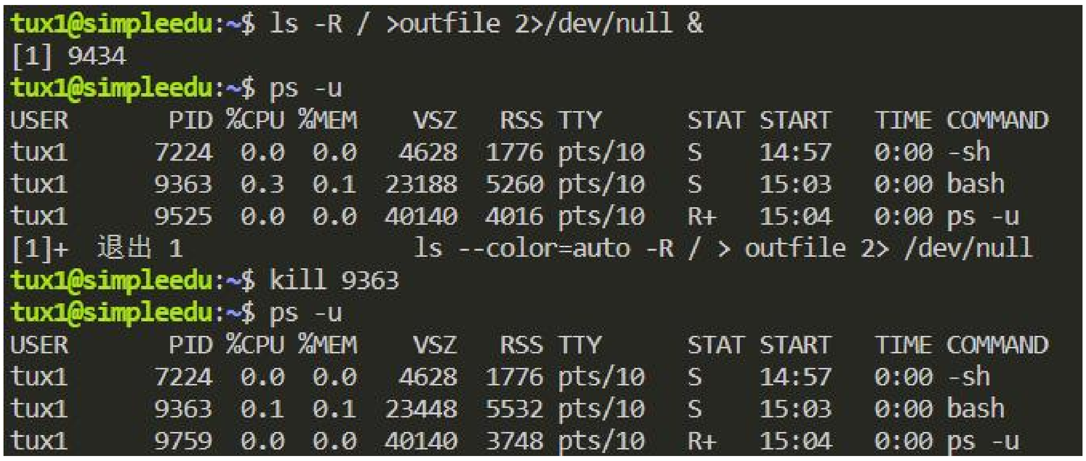

```
tux2@simpleedu:~$ vi tux2_file
```


```
"tux2_file" [新文件]
```

6.Go back to tty1 and show all the processes in your system. If necessary, look in the man pages and info to find the correct options to show all processes running in your system. Look for your own processes as well as the processes of tux2.


```
tux1@simpleedu:~$ ps -u tux1
  PID TTY          TIME CMD
 7224 pts/10   00:00:00 sh
 7268 pts/10   00:00:00 bash
 7409 pts/10   00:00:00 ps
tux1@simpleedu:~$ ps -u tux2
  PID TTY          TIME CMD
 4848 ?        00:00:00 sh
 5246 ?        00:00:00 sh
 5364 ?        00:00:00 bash
 6535 pts/10   00:00:00 sh
 6565 pts/10   00:00:00 bash
```

7.Again, run the ls -R / >outfile 2>/dev/null & command and then exit your current process. List the processes you are running. What happens to processes if you kill their parent process?



```
tux1@simpleedu:~$ ls -R / >outfile 2>/dev/null &
[1] 9434
tux1@simpleedu:~$ ps -u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
tux1      7224  0.0  0.0   4628  1776 pts/10   S    14:57   0:00 -sh
tux1      9363  0.3  0.1  23188  5260 pts/10   S    15:03   0:00 bash
tux1      9525  0.0  0.0  40140  4016 pts/10   R+   15:04   0:00 ps -u
[1]+  退出 1                 ls --color=auto -R / > outfile 2> /dev/null
tux1@simpleedu:~$ kill 9363
tux1@simpleedu:~$ ps -u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
tux1      7224  0.0  0.0   4628  1776 pts/10   S    14:57   0:00 -sh
tux1      9363  0.1  0.1  23448  5532 pts/10   S    15:03   0:00 bash
tux1      9759  0.0  0.0  40140  3748 pts/10   R+   15:04   0:00 ps -u
```

杀死父进程，子进程也会死亡。

Job control

8.Using Vi or another editor, create the file named myclock in your bin directory with the following contents:

while true

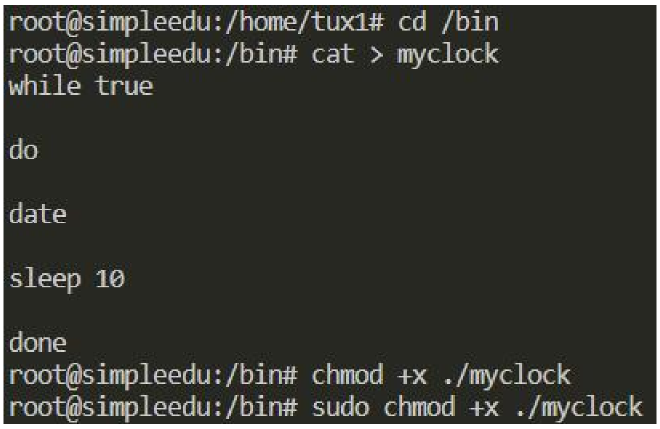
do

date

sleep 10

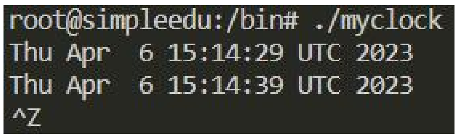
done

Make the script executable.



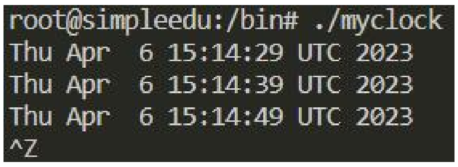
```
root@simpleedu:/home/tux1# cd /bin
root@simpleedu:/bin# cat > myclock
while true

do

date

sleep 10

done
root@simpleedu:/bin# chmod +x ./myclock
root@simpleedu:/bin# sudo chmod +x ./myclock
```

9.Run the script myclock. Run this script in the foreground.



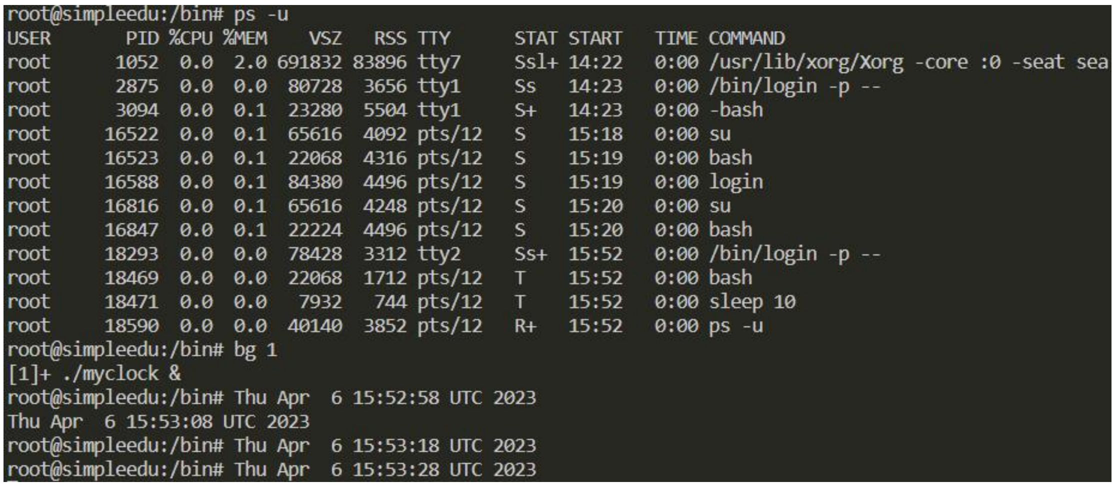
```
root@simpleedu:/bin# ./myclock
Thu Apr  6 15:14:29 UTC 2023
Thu Apr  6 15:14:39 UTC 2023
^Z
```

10.Suspend the job you just started.



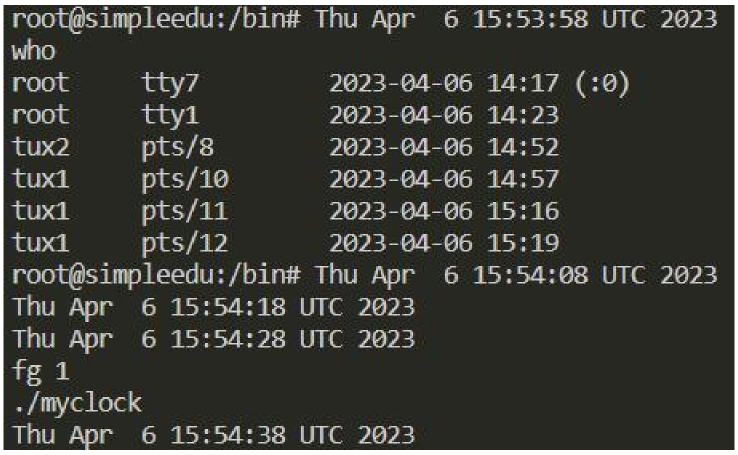
```
root@simpleedu:/bin# ./myclock
Thu Apr  6 15:14:29 UTC 2023
Thu Apr  6 15:14:39 UTC 2023
Thu Apr  6 15:14:49 UTC 2023
^Z
```

11.List all the jobs that you are running on the system and restart the above job in the background.



```
root@simpleedu:/bin# ps -u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      1052  0.0  2.0 691832 83896 tty7     Ssl+ 14:22   0:00 /usr/lib/xorg/Xorg -core :0 -seat sea
root      2875  0.0  0.0  80728  3656 tty1     Ss   14:23   0:00 /bin/login -p --
root      3094  0.0  0.1  23280  5504 tty1     S+   14:23   0:00 -bash
root     16522  0.0  0.1  65616  4092 pts/12   S    15:18   0:00 su
root     16523  0.0  0.1  22068  4316 pts/12   S    15:19   0:00 bash
root     16588  0.0  0.1  84380  4496 pts/12   S    15:19   0:00 login
root     16816  0.0  0.1  65616  4248 pts/12   S    15:20   0:00 su
root     16847  0.0  0.1  22224  4496 pts/12   S    15:20   0:00 bash
root     18293  0.0  0.0  78428  3312 tty2     Ss+  15:52   0:00 /bin/login -p --
root     18469  0.0  0.0  22068  1712 pts/12   T    15:52   0:00 bash
root     18471  0.0  0.0   7932   744 pts/12   T    15:52   0:00 sleep 10
root     18590  0.0  0.0  40140  3852 pts/12   R+   15:52   0:00 ps -u
root@simpleedu:/bin# bg 1
[1]+ ./myclock &
root@simpleedu:/bin# Thu Apr  6 15:52:58 UTC 2023
Thu Apr  6 15:53:08 UTC 2023
root@simpleedu:/bin# Thu Apr  6 15:53:18 UTC 2023
root@simpleedu:/bin# Thu Apr  6 15:53:28 UTC 2023
```

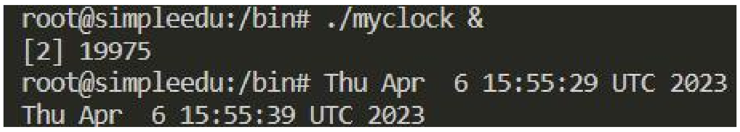
12.List all users that are logged in. Bring the job back to the foreground, wait until you
get a timestamp, and then exit the job.



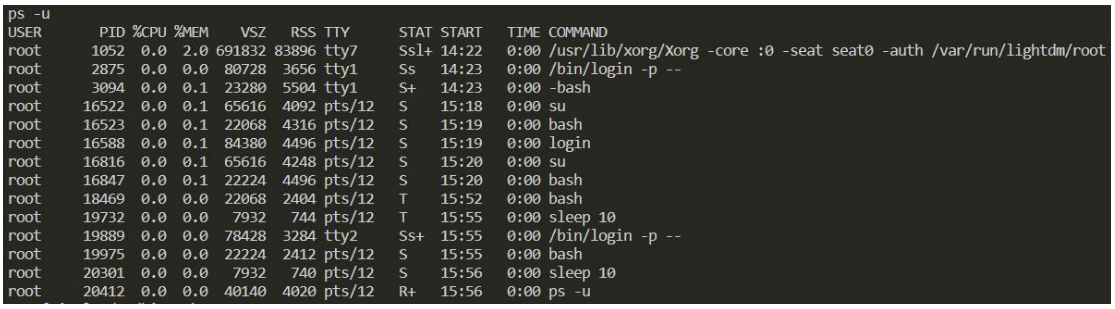
```
root@simpleedu:/bin# Thu Apr  6 15:53:58 UTC 2023
who
root     tty7         2023-04-06 14:17 (:0)
root     tty1         2023-04-06 14:23
tux2     pts/8        2023-04-06 14:52
tux1     pts/10       2023-04-06 14:57
tux1     pts/11       2023-04-06 15:16
tux1     pts/12       2023-04-06 15:19
root@simpleedu:/bin# Thu Apr  6 15:54:08 UTC 2023
Thu Apr  6 15:54:18 UTC 2023
Thu Apr  6 15:54:28 UTC 2023
fg 1
./myclock
Thu Apr  6 15:54:38 UTC 2023
```

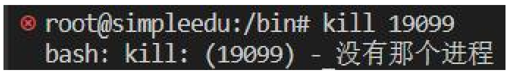
**Terminating a process**

13.Execute the myclock script again, this time in the background. Hint: Take note of the PID.

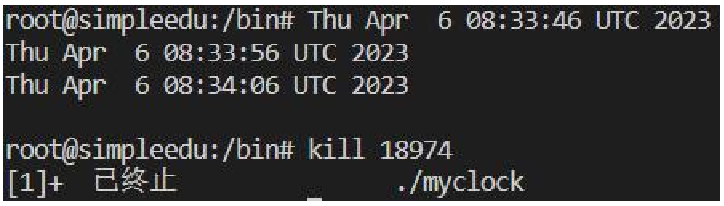

```
root@simpleedu:/bin# ./myclock &
[2] 19975
root@simpleedu:/bin# Thu Apr  6 15:55:29 UTC 2023
Thu Apr  6 15:55:39 UTC 2023
```

14.List all your processes and kill the sleep process. What happened?


```
ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root        1052  0.0  2.0 691832 83896 tty7     Ssl+ 14:22   0:00 /usr/lib/xorg/Xorg -core :0 -seat seat0 -auth /var/run/lightdm/root
root        2875  0.0  0.0  80728  3656 tty1     Ss   14:23   0:00 /bin/login -p --
root        3094  0.0  0.1  23280  5504 tty1     S+   14:23   0:00 -bash
root       16522  0.0  0.1  65616  4092 pts/12   S    15:18   0:00 su
root       16523  0.0  0.1  22068  4316 pts/12   S    15:19   0:00 bash
root       16588  0.0  0.1  84380  4496 pts/12   S    15:19   0:00 login
root       16816  0.0  0.1  65616  4248 pts/12   S    15:20   0:00 su
root       16847  0.0  0.1  22224  4496 pts/12   S    15:20   0:00 bash
root       18469  0.0  0.0  22068  2404 pts/12   T    15:52   0:00 bash
root       19732  0.0  0.0   7932   744 pts/12   T    15:55   0:00 sleep 10
root       19889  0.0  0.0  78428  3284 tty2     Ss+  15:55   0:00 /bin/login -p --
root       19975  0.0  0.0  22224  2412 pts/12   S    15:55   0:00 bash
root       20301  0.0  0.0   7932   740 pts/12   S    15:56   0:00 sleep 10
root       20412  0.0  0.0  40140  4020 pts/12   R+   15:56   0:00 ps -u
```


```
root@simpleedu:/bin# kill 19099
bash: kill: (19099) - 没有那个进程
```

15.Now stop the shell script myclock.


```
root@simpleedu:/bin# Thu Apr  6 08:33:46 UTC 2023
Thu Apr  6 08:33:56 UTC 2023
Thu Apr  6 08:34:06 UTC 2023

root@simpleedu:/bin# kill 18974
[1]+  已终止               ./myclock
```

## 1. 3 Exercise3 Shell scripting

### What this exercise is about

After you have been using Linux for a while, you find certain characteristics of your environment that you would like to customize along with some tasks that you execute regularly that you would like to automate.

This exercise introduces you to some of the more common constructs used to help you write shell scripts to customize and automate your computing environment.

### What you should be able to do

At the end of the exercise, you should be able to:

- List common constructs used in writing shell scripts
- Create and execute simple shell scripts

### Introduction

You need no programming experience to perform this exercise. Refer to the unit in the Student Notebook for help with the syntax of constructs when creating the shell scripts in this exercise.

## Requirements

- This workbook
- A workstation with Fedora, RHEL, or SLES installed

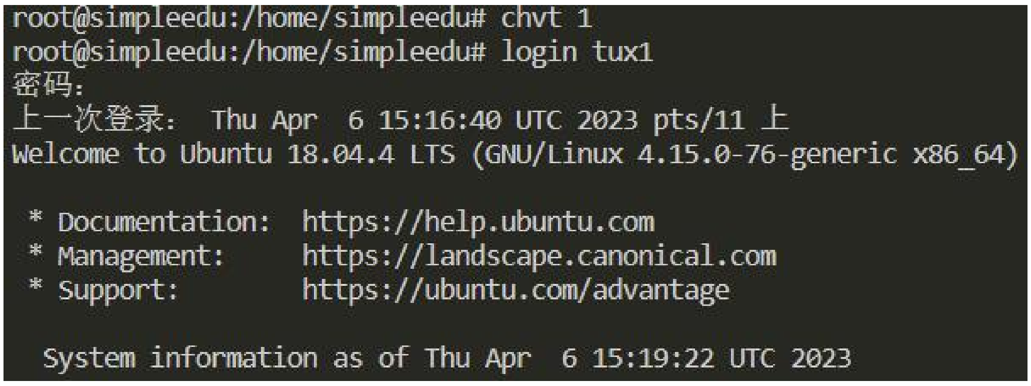
## Exercise instructions

Preface

All exercises in this chapter depend on the availability of specific equipment in your classroom.

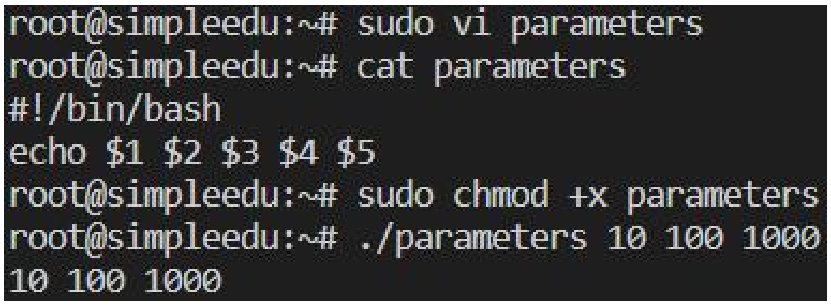
## Working with positional parameters

1.If you are not logged in as tux1 at tty1, log in now.



```
root@simpleedu:/home/simpleedu# chvt 1
root@simpleedu:/home/simpleedu# login tux1
密码:
上一次登录:  Thu Apr  6 15:16:40 UTC 2023 pts/11 上
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-76-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu Apr  6 15:19:22 UTC 2023
```

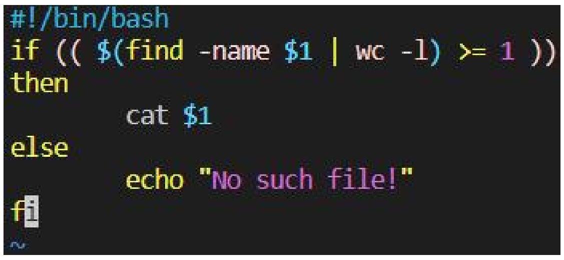
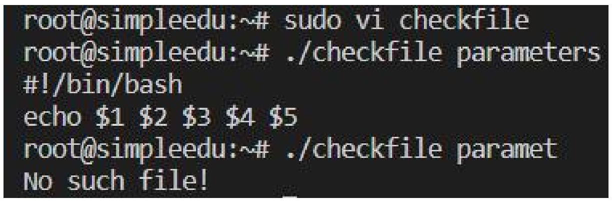
2.Create a shell script named parameters that echoes the five parameters that follow using predefined special variables set by the shell to fill in the blanks. Execute the script using the positional parameters 10 100 1000.

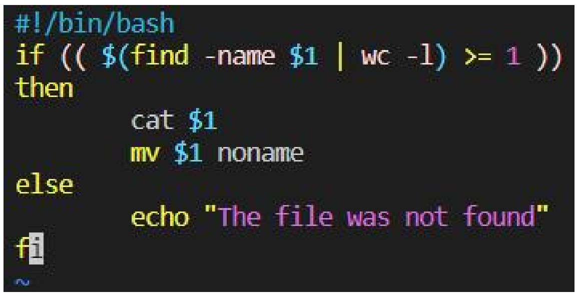
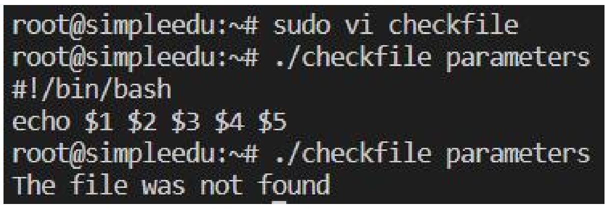


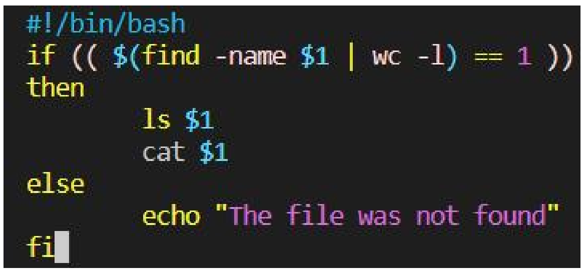
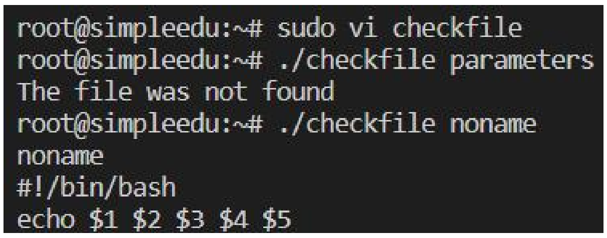
```
root@simpleedu:~# sudo vi parameters
root@simpleedu:~# cat parameters
#!/bin/bash
echo $1 $2 $3 $4 $5
root@simpleedu:~# sudo chmod +x parameters
root@simpleedu:~# ./parameters 10 100 1000
10 100 1000
```

## Conditional execution

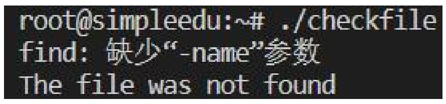
3.Using conditional execution, create a shell script named checkfile that checks to see if the file named parameters exists in your directory. If it exists, use a command to show the contents of the file. Execute the script.


```
#!/bin/bash
if (( $(find -name $1 | wc -l) >= 1 ))
then
        cat $1
else
        echo "No such file!"
fi
~
```

```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile parameters
#!/bin/bash
echo $1 $2 $3 $4 $5
root@simpleedu:~# ./checkfile paramet
No such file!
```
4.Modify the checkfile script and change the name of the file from parameters to noname (check to ensure that you do not have a file by this name in your current directory). Also, using conditional execution, if the cat command was not successful, display the error message, The file was not found. Execute the script.

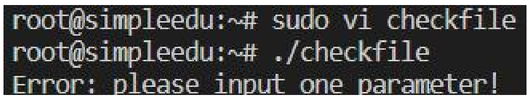

```
#!/bin/bash
if (( $(find -name $1 | wc -l) >= 1 ))
then
        cat $1
        mv $1 noname
else
        echo "The file was not found"
fi
~
```

```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile parameters
#!/bin/bash
echo $1 $2 $3 $4 $5
root@simpleedu:~# ./checkfile parameters
The file was not found
```
5.Modify the checkfile script to accept a single parameter from the command line as input to the ls and cat commands. Execute the script twice, once using the file named parameters and again using the file named noname.


```
#!/bin/bash
if (( $(find -name $1 | wc -l) == 1 ))
then
        ls $1
        cat $1
else
        echo "The file was not found"
fi
```

```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile parameters
The file was not found
root@simpleedu:~# ./checkfile noname
noname
#!/bin/bash
echo $1 $2 $3 $4 $5
```

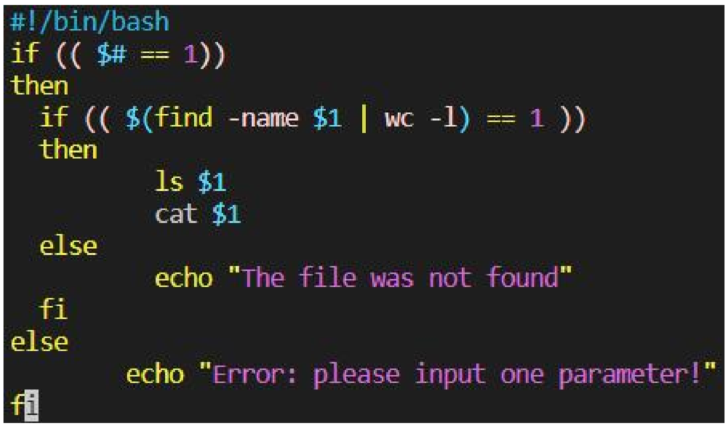
6.Execute the checkfile script again but this time with no parameters. What happens? Modify the script so that this does not happen again.



```
root@simpleedu:~# ./checkfile
find: 缺少“-name”参数
The file was not found
```



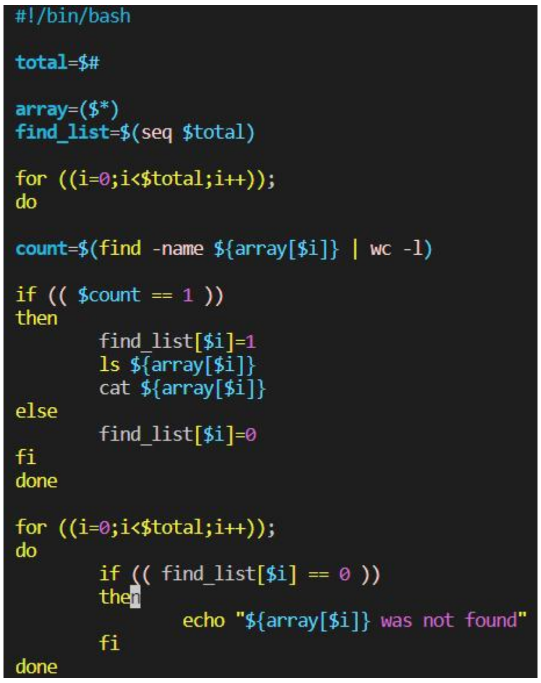
```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile
Error: please input one parameter!
```



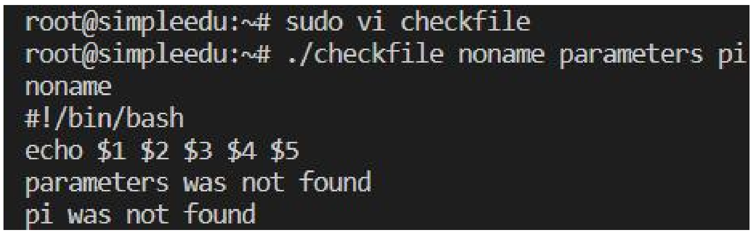
```
#!/bin/bash
if (( $# == 1))
then
  if (( $(find -name $1 | wc -l) == 1 ))
  then
          ls $1
          cat $1
  else
          echo "The file was not found"
  fi
else
        echo "Error: please input one parameter!"
fi
```

**Loops**

7.Using the for loop, modify the checkfile script to accept multiple files as input from the command line instead of just one. If the files are found, display all of them. If the files are not found, display an error message showing all file names that were not found. Look in your directory and note a few valid file names that you can use as input. Execute the script using valid and invalid file names.

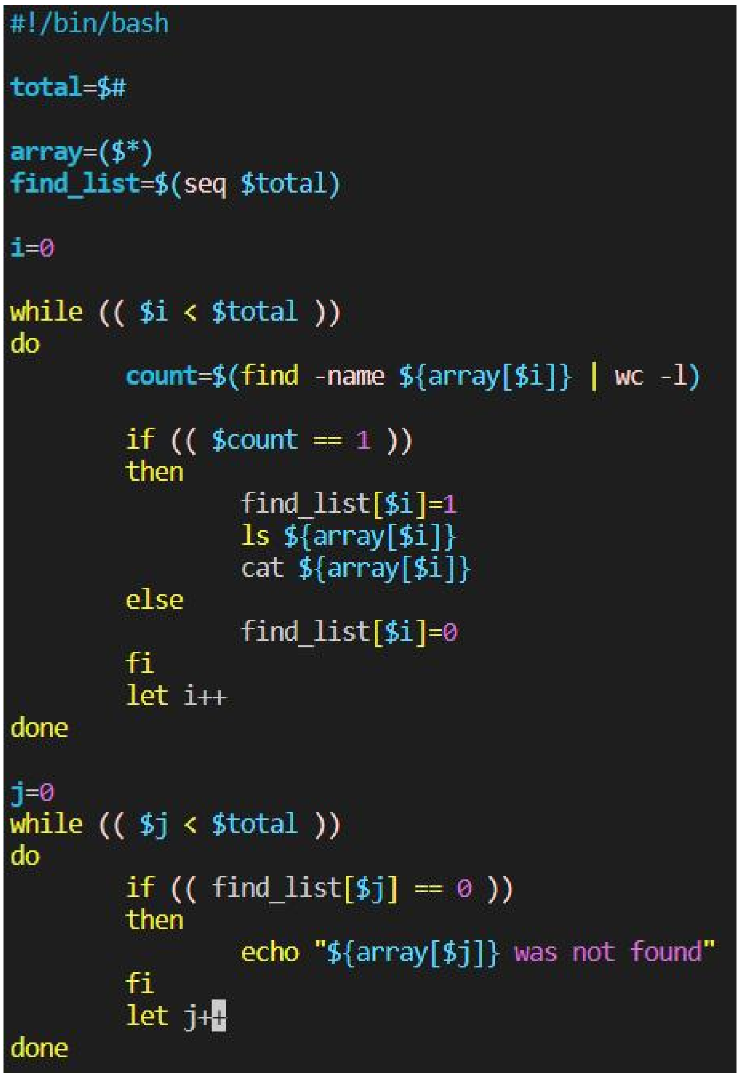


```
#!/bin/bash

total=$#

array=($*)
find_list=$(seq $total)

for ((i=0;i<$total;i++));
do

count=$(find -name ${array[$i]} | wc -l)

if (( $count == 1 ))
then
        find_list[$i]=1
        ls ${array[$i]}
        cat ${array[$i]}
else
        find_list[$i]=0
fi
done

for ((i=0;i<$total;i++));
do
        if (( find_list[$i] == 0 ))
        then
                echo "${array[$i]} was not found"
        fi
done
```

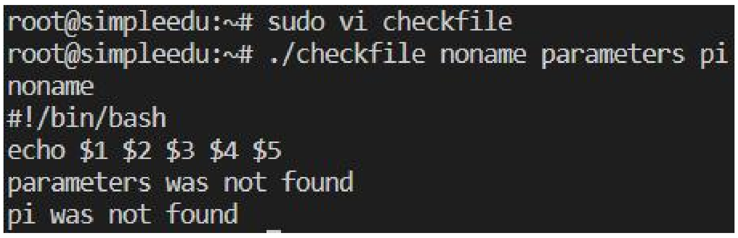


```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile noname parameters pi
noname
#!/bin/bash
echo $1 $2 $3 $4 $5
parameters was not found
pi was not found
```

8.Now do the same thing, but use a while loop in combination with the shift command.



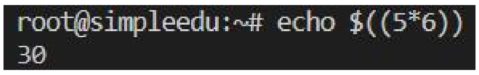
```
#!/bin/bash

total=$#

array=($*)
find_list=$(seq $total)

i=0

while (( $i < $total ))
do
        count=$(find -name ${array[$i]} | wc -l)

        if (( $count == 1 ))
        then
                find_list[$i]=1
                ls ${array[$i]}
                cat ${array[$i]}
        else
                find_list[$i]=0
        fi
        let i++
done

j=0
while (( $j < $total ))
do
        if (( find_list[$j] == 0 ))
        then
                echo "${array[$j]} was not found"
        fi
        let j++
done
```



```
root@simpleedu:~# sudo vi checkfile
root@simpleedu:~# ./checkfile noname parameters pi
noname
#!/bin/bash
echo $1 $2 $3 $4 $5
parameters was not found
pi was not found
```

**Arithmetic**

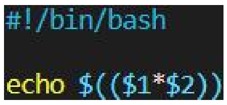
9.From the command line, display the results of multiplying 5 times 6.



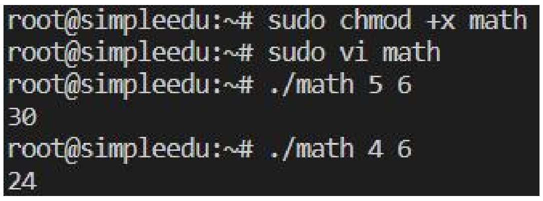
```
root@simpleedu:~# echo $((5*6))
30
```

10.Now create a shell script named math to multiply any two numbers when entered as

input from the command line. Execute the script multiplying 5 times 6. Experiment with any other two numbers.



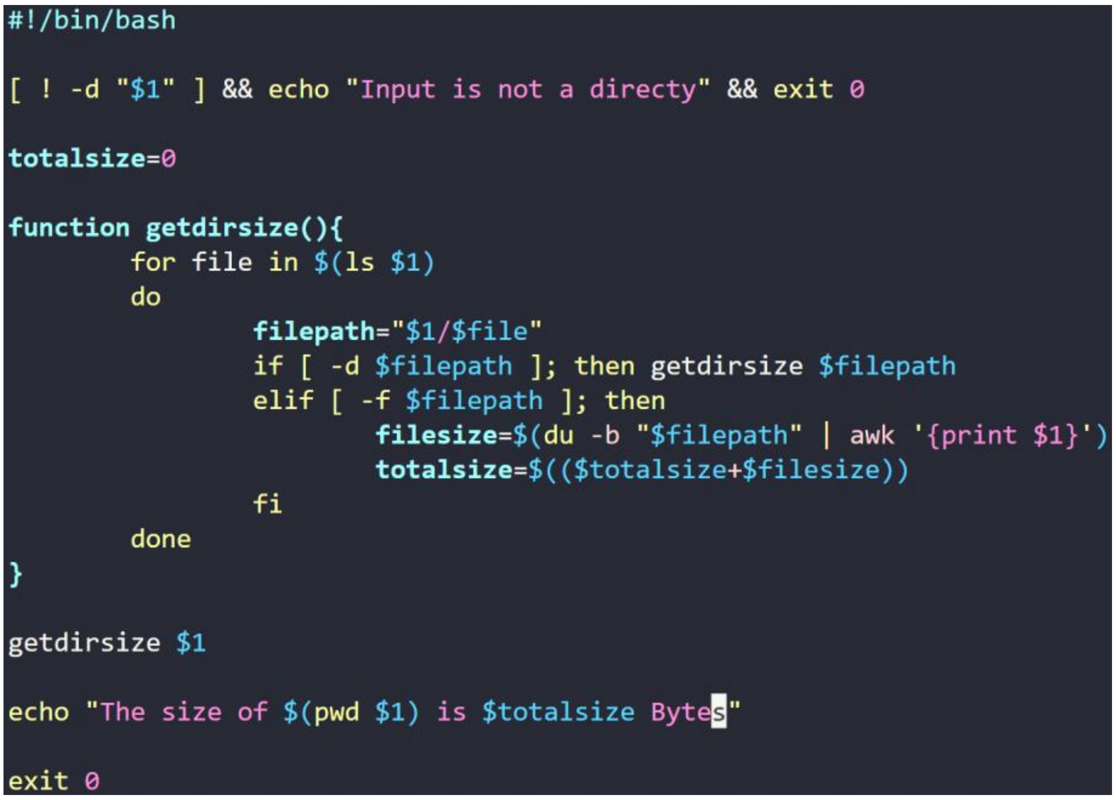
```
#!/bin/bash

echo $(($1*$2))
```



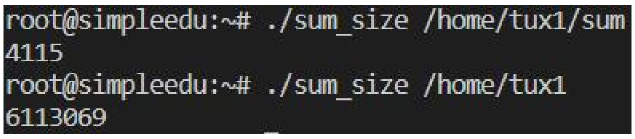
```
root@simpleedu:~# sudo chmod +x math
root@simpleedu:~# sudo vi math
root@simpleedu:~# ./math 5 6
30
root@simpleedu:~# ./math 4 6
24
```

**Integration exercise**

11.Use the knowledge you gained in this course to write a script that accepts a directory name as a parameter and calculate the total size of the files in this directory.



```
#!/bin/bash

[ ! -d "$1" ] && echo "Input is not a directy" && exit 0

totalsize=0

function getdirsize(){
        for file in $(ls $1)
        do
                filepath="$1/$file"
                if [ -d $filepath ]; then getdirsize $filepath
                elif [ -f $filepath ]; then
                        filesize=$(du -b "$filepath" | awk '{print $1}')
                        totalsize=$(($totalsize+$filesize))
                fi
        done
}

getdirsize $1

echo "The size of $(pwd $1) is $totalsize Bytes"

exit 0
```



```
root@simpleedu:~# ./sum_size /home/tux1/sum
4115
root@simpleedu:~# ./sum_size /home/tux1
6113069
```

Note: The column numbers might need to be adjusted a little.

End of exercise