

Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет
телекоммуникаций и информатики»

Лабораторная работа №1
«Среда пользователя UNIX»

Выполнил: студент 4 курса

ИВТ, гр. ИП-13

Бурдуковский И.А.

Проверил: преподаватель кафедры ПМиК

Белевцова Екатерина Андреевна

Новосибирск, 2021 г.

1. Определить тип файлов /dev/hd0, /dev/console, /dev/ttyp0, /dev/shmem, /dev/mem.

Для выполнения буду пользоваться командой `file -b <путь к файлу>`, которая отображает тип файла. Флаг `-b` (briefly) показывает только тип файла,

```
# file /dev/hd0
/dev/hd0: block special (4/0)
# file /dev/c
cd0      con1      con2      con3      con4      console  crypto
# file /dev/console
/dev/console: character special (0/1)
# file /dev/ttyp
ttyp0    ttyp2    ttyp4    ttyp6    ttyp8    ttyra    ttyrc    ttyre
ttyp1    ttyp3    ttyp5    ttyp7    ttyp9    ttyrb    ttyrd    ttyrf
# file /dev/ttyp0
/dev/ttyp0: character special (7/0)
# file /dev/shmem/
/dev/shmem/: directory
# file /dev/mem
/dev/mem: data
# file _
```

можно передать несколько файлов сразу, что я и сделал, запустив команду:

```
# file -b /dev/hd0 /dev/console /dev/ttyp0 /dev/shmem /dev/mem
```

/dev/hd0 – блочный файл, файл устройства, который предоставляет буферизированный доступ к оборудованию системы

/dev/console – символьный файл – файл устройств, который обеспечивает не буферизованный последовательный доступ к системным аппаратным компонентам.

/dev/ttyp0 – тоже символьный файл

/dev/shmem – директория

/dev/mem – файл с данными – содержит текст, данные или программные инструкции

2. Определить, какой каталог делается рабочим при входе в систему. Почему?

Чтобы определить рабочий каталог, воспользуемся командой `# pwd – print working directory` – вывести рабочую директорию.

```
# pwd
/root
```

/root – рабочий каталог по умолчанию для суперпользователя. Является рабочим по умолчанию, потому что так было решено разработчиками ОС.

3. Создать каталог LAB1 и сделать его рабочим.

Для создания нового каталога воспользуемся командой `# mkdir <название нового каталога>`. Далее сделаем его рабочим с помощью команды `# cd <название каталога для перехода>`. Проверим результат работы с помощью `pwd`.

```
# mkdir LAB1
# cd LAB1
# pwd
/LAB1
#
```

4. Определить (с помощью программы ls), в каком каталоге содержится файл services. Посмотреть его содержимое.

```
# Network services, Internet style
#
#      @(#)services      5.8 (Berkeley) 5/9/91
#
echo      7/tcp
echo      7/udp
discard   9/tcp      sink null
discard   9/udp      sink null
systat    11/tcp     users
daytime   13/tcp
daytime   13/udp
netstat   15/tcp
qotd      17/tcp     quote
chargen   19/tcp     ttytst source
chargen   19/udp     ttytst source
ftp       21/tcp
telnet    23/tcp
smtp      25/tcp     mail
time      37/tcp     timserver
time      37/udp     timserver
rlp       39/udp     resource      # resource location
nameserver 42/tcp     name          # IEN 116
whois     43/tcp     nicname
domain    53/tcp     nameserver    # name-domain server
domain    53/udp     nameserver
mtp       57/tcp     # deprecated
# Bootp experimental (sellgren@vangogh)
bootp     67/udp     bootps        # bootp server
bootpc    68/udp     # bootp client
#
tftp      69/udp
rje       77/tcp     netrjs
finger    79/tcp
link      87/tcp     ttylink
supdup    95/tcp
hostnames 101/tcp     hostname      # usually from sri-nic
tsap      102/tcp     # part of ISODE.
#csnet-cs 105/?
```

Чтобы просмотреть содержимое файла воспользуемся командой `# cat /etc/services`

5. Сколько скрытых файлов в вашем домашнем каталоге?

Для начала необходимо перейти в домашний каталог по пути /home/user. Далее введя команду ls с флагом -la, произойдет вывод всех файлов, в том числе и скрытых, начинающихся с символа точка

```
# ls -la
total 82
drwxr-x--x  9 root    root    4096 Sep 20  2019 .
drwxr-xr-x 15 root    root    4096 Sep 04 10:20 ..
-rw-rw-r--  1 root    root      0 Sep 04 10:04 .lastlogin
drwx-----  3 root    root    4096 Sep 04  2019 .mozilla
drwxrwxr-x  7 root    root    4096 Sep 04 10:06 .ph
-rw-r--r--  1 root    root    191 Apr 20  2001 .profile
drwxr-xr-x  2 root    root    8192 Sep 17  2019 EX
drwxr-xr-x 12 101     user    4096 Feb 09  2015 Lectures
drwxr-xr-x  4 root    root    4096 Sep 18  2019 VG
drwxrwxr-x  2 101     user    4096 Oct 16  2014 demos
drwxr-xr-x  2 101     user    4096 Nov 07  2013 labs
-rw-rw-r--  1 root    root    443 Nov 10  2001 raw.h
```

В домашнем каталоге 4 скрытых файла

6. Определить полное дерево подкаталогов в /boot . Сколько там файлов, размер которых меньше 1К байт? Сколько там исполняемых файлов?

Для отображения дерева подкаталогов в директории /boot использовать команду ls чтобы увидеть структуру, и, впоследствии, использовать команду ls на самих подкаталогах. Имеем 6 файлов весом меньше 1КБ.

```
# ls -l /boot/fs
total 9060
drwxrwxr-x  2 root    root      4096 Sep 04  2019 .
drwxrwxr-x  5 root    root      4096 Sep 04  2019 ..
-rw-rw-r--  1 root    root    1541356 Sep 04  2019 qnxbase.ifs
-rw-rw-r--  1 root    root    1541356 Sep 04  2019 qnxbasedma.ifs
-rw-rw-r--  1 root    root    1547148 Sep 04  2019 qnxbasesmp.ifs
# ls -l /boot/build/
total 48
drwxrwxr-x  2 root    root      4096 Sep 04  2019 .
drwxrwxr-x  5 root    root      4096 Sep 04  2019 ..
-rw-r--r--  1 root    root     3253 Oct 24  2008 bios.build
-rw-r--r--  1 root    root     2543 May 11  2010 finstall.build
-rw-r--r--  1 root    root     2266 May 11  2010 qnxbase.build
-rw-r--r--  1 root    root     2266 May 11  2010 qnxbasedma.build
-rw-r--r--  1 root    root     2282 Jun 14  2010 qnxbasesmp-apic.build
-rw-r--r--  1 root    root     2270 May 11  2010 qnxbasesmp.build
# ls -l /boot/sys
total 12416
drwxrwxr-x  2 root    root      4096 Sep 04  2019 .
drwxrwxr-x  5 root    root      4096 Sep 04  2019 ..
-rwxr-xr-x  1 root    root     3642 Jul 10  2010 bios.boot
-rwxr-xr-x  1 root    root     3667 Jul 10  2010 bios16m.boot
-rwxr-xr-x  1 root    root     3618 Jul 10  2010 bios_nokbd.boot
-rw-r--r--  1 root    root       218 Jul 10  2010 elf.boot
-rwxr-xr-x  1 root    root       436 Jul 10  2010 ipl-diskpc1
-rwxr-xr-x  1 root    root       328 Jul 10  2010 ipl-diskpc1-flop
-rwxr-xr-x  1 root    root       472 Jul 10  2010 ipl-diskpc2
-rwxr-xr-x  1 root    root       460 Jul 10  2010 ipl-diskpc2-flop
-rw-r--r--  1 root    root    63532 Jul 10  2010 libmod_aps.a
-rw-r--r--  1 root    root       179 Jul 10  2010 nobios.boot
-rw-r--r--  1 root    root    872512 Jul 10  2010 procnto
-rw-r--r--  1 root    root   971452 Jul 10  2010 procnto-instr
-rw-r--r--  1 root    root   913104 Jul 10  2010 procnto-smp
-rw-r--r--  1 root    root  1014952 Jul 10  2010 procnto-smp-instr
-rwxr-xr-x  1 root    root   911546 Jul 10  2010 startup-apic
-rwxr-xr-x  1 root    root   793685 Jul 10  2010 startup-bios
-rwxr-xr-x  1 root    root   789699 Jul 10  2010 startup-bios-32
```

7. Сколько жестких связей у каталога /boot и почему?

Для этого воспользуемся командой `# ls -l /`. -l полный вывод информации, второй параметр как раз показывает количество жестких связей, у каталога boot их 5.

```
# ls -l boot/
total 40
drwxrwxr-x  5 root      root      4096 Sep 04 2019 .
drwxr-xr-x 15 root      root      4096 Sep 04 10:20 ..
drwxrwxr-x  2 root      root      4096 Sep 04 2019 build
drwxrwxr-x  2 root      root      4096 Sep 04 2019 fs
drwxrwxr-x  2 root      root      4096 Sep 04 2019 sys
#
```

8. Создать текстовый файл с помощью редактора vi. Какие флаги доступа устанавливаются у вновь создаваемого файла? Почему? Как это исправить?

Для создание воспользуемся командой `# vi text`. Сохранимся и выйдем. Покоманде `# ls -l` можно посмотреть флаги доступа у файла text.

```
# ls -l
total 17
drwxrwxr-x  2 root      root      4096 Sep 04 10:58 .
drwxr-xr-x 15 root      root      4096 Sep 04 10:20 ..
-rw-rw-r--  1 root      root       5 Sep 04 10:59 text
#
```

По умолчанию установились флаги доступа:

- 1) Для пользователя (user) – чтение (r), запись(w).
- 2) Для группы (group) - чтение (r), запись(w).
- 3) Для других (other) – чтение(r).

Такие флаги доступа установились по умолчанию так, чтобы был доступ к записи файла только для процессов, созданных пользователем или группой.

Для изменения флагов доступа существует утилита `chmod`

9. Сделать каталог и создать в нем 10 копий некоторого файла. Перенести три из них в вышестоящий каталог. Удалить (с подтверждением) некоторые из оставшихся файлов. Проверить влияние флага w на команду удаления файла.

Для создания каталога – `mkdir`, для создания файла – `touch`. Для того чтобы скопировать файл 10 раз напишем `сору`. Для переноса файлов – `mv`, для удаления `rm`.

```

# cp copy copy6
# cp copy copy7
# cp copy copy8
# cp copy copy9
# ls
.          copy      copy2      copy4      copy6      copy8
..         copy1     copy3     copy5     copy7     copy9
# mv copy ..
# mv copy2 ..
# mv copy1 ..
# ls
.          copy3     copy5     copy7     copy9
..         copy4     copy6     copy8
#
# rm copy3
# ls
.          copy4     copy6     copy8
..         copy5     copy7     copy9
# rm -w copy
copy4 copy5 copy6 copy7 copy8 copy9
# rm -w copy4
rm: illegal option -- w
# rm -i copy4
rm: remove copy4? (y/N) y
#

```

Файлы были удалены с подтверждением.

Состояние флага w не влияет на удаление файла.

10. Определить значения переменных среды PATH, LOGNAME, HOME, HOSTNAME, PWD, RANDOM. Меняются ли они со временем?

```

# echo $PATH
/sbin:/usr/sbin:/bin:/usr/bin:/usr/photon/bin:/usr/photon/appbuilder:/opt/X11R6/
bin:/usr/X11R6/bin:/usr/local/bin:/opt/bin:/opt/sbin:/usr/qnx650/host/qnx6/x86/u
sr/bin:/usr/qnx650/host/qnx6/x86/usr/sbin:/usr/qnx650/host/qnx6/x86/sbin:/usr/qn
x650/host/qnx6/x86/bin:/usr/qnx650/host/qnx6/x86/usr/photon/appbuilder
# echo $LOGNAME
root
# echo $HOME
/root
# echo $HOSTNAME
localhost
# echo $PWD
/LAB1/10
# echo $RANDOM
10549
# echo $RANDOM
31599
#

```

`$PATH` — это переменная среды, используемая для указания оболочке, где искать исполняемые файлы. Не меняется со временем, но поддается редактированию.

`$LOGNAME` — содержит имя пользователя, так же не меняется со временем, но поддается редактированию.

`$HOME` — домашний каталог, можно изменить, со временем не меняется.

`$HOSTNAME` — имя компьютера, не изменяется.

`$PWD` — рабочий каталог, изменяется при изменении рабочего каталога.

`$RANDOM` — случайное число, изменяется в зависимости от текущего времени.

11. Определить коды завершения команд `ls /bin` и `ls /pin`

Коды завершения команд можно посмотреть, введя `# echo $?` - после выполнения команды.

Код завершения - 0 — т.к. команда успешно завершилась, выведя результат.

Код завершения — 1. Т.к. работа команды завершилась ошибкой — нет каталога.

```
# echo $?
0
# ls /pin
ls: No such file or directory (/pin)
# echo $?
1
#
```

12. Вывести содержимое каталога /bin в файл в несколько колонок. Затем добавить к нему распечатку каталога /usr/bin.

Чтобы результат работы команды добавить в файл, необходимо после команды добавить символ «>» и ввести название файла.

```
dumpefs          nslookup         termdef
egl-gears        nsupdate         textto
egl-intermix     ntox86-ld        tftp
egl-mipmap       ntox86-ld-2.19   tic
egl-pdemo        ntpdc            time
egl-planetary    ntpq             top
egl-spheres      ntptrace         touch
egl-testegl      od              tr
egrep            omshell          traceprinter
env              op               traceroute
epijs-qnx        openssl          traceroute6
errno            passwd           tsort
etfscctl         paste            tty
expand           patch            umask
expr             pdebug           unexpand
fcatt            ph               unifdef
fgrep            phin             uniq
file             phrelay          unlink
find             phs-to-bjc       unzip
flashctl         phs-to-bmp       unzipsfx
fmt              phs-to-escp2     uptime
fold             phs-to-ijs       use
font-cache       phs-to-pcl       uud
font-test        phs-to-phs       uudecode
freeze           phs-to-ps        uue
fsysinfo         ping             uuencode
ftp              ping6            vsync
fullpath         portmap          wave
funzip           pr               waverec
gawk             printf           wc
get_hw_info      qconfig          which
getconf          qcp             xargs
gf-calib         qdbc            zap
gf_cursor        qed             zip
gfi-demo         qinst           zipcloak
grep             qnxactivate     zipinfo
hd               qtalk           zipnote
head             rcp             zipsplit
hogs             renice
```

Cat – проверка что файл успешно записался. Чтобы добавить таким же образом данные в уже существующий файл необходимо использовать управляющий символ «>>>»

13. Сколько файлов удалили бы команды `rm /usr/bin/g*` и `rm /usr/bin/t??` ? (просьба файлы не удалять)

Для того чтобы узнать это, можно воспользоваться командой `ls -l <путь к каталогу>` совместно с утилитой `wc` с флагом `-l` для подсчета количества строк в результате работы команды `ls`.

```
# ls usr/bin/g* | wc -l
7
# ls usr/bin/t?? | wc -l
5
```

Сначала удалится 7 файлом, а затем 5.

14. Сколько всего пользователей зарегистрировано в системе?

Количество пользователей можно посмотреть в файле `/etc/passwd`, имеющем синтаксис:

имя_пользователя:пароль:ид:ид_группы:группа:домашний_каталог:оболочка

Пользователи с `ID < 100` – системные, они были созданы во время установки.
Пользователь с `ID = 0` – суперпользователь.

```
# cat etc/passwd
root::0:0:Superuser:/root:/bin/sh
bin:x:1:1:Binaries Commands and Source:/bin:
daemon:x:2:2:System Services:/daemon:
mail:x:8:40:User Mail:/var/spool/mail:
news:x:9:50:Network News:/var/spool/news:
uucp:x:12:60:Network News:/var/spool/news:
ftp:x:14:80:FTP User:/home/ftp:
sshd:x:15:6:sshd:/var/chroot/sshd:/bin/false
nobody:x:99:99:Nobody:/:
user::100:100:User:/home/user:/bin/sh
```

В данном случае пользователей 2 – root – суперпользователь и вручную зарегистрированный – user. Всего их 10.

15. Сколько различных групп пользователей в системе?

Группы пользователей хранятся в файле /etc/group и их 15.

```
# cat etc/passwd
root::0:0:Superuser:/root:/bin/sh
bin:x:1:1:Binaries Commands and Source:/bin:
daemon:x:2:2:System Services:/daemon:
mail:x:8:40:User Mail:/var/spool/mail:
news:x:9:50:Network News:/var/spool/news:
uucp:x:12:60:Network News:/var/spool/news:
ftp:x:14:80:FTP User:/home/ftp:
sshd:x:15:6:sshd:/var/chroot/sshd:/bin/false
nobody:x:99:99:Nobody:/:
user::100:100:User:/home/user:/bin/sh
# cat etc/group
root:x:0:root
bin:x:1:root,bin
daemon:x:2:daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:root
sshd:x:6:
mail:x:40:mail
news:x:50:news
uucp:x:60:uucp
ftp:x:80:ftp
guest:x:90:
nobody:x:99:
display:x:82:
user::100:
```

16. Определить имена пользователей, у которых нет пароля.

Из задания 14 можно узнать, что это «пользователи» с символом X на втором месте, а именно: bin, daemon, main, news, uucp, ftp, sshd, nobody – 8 «пользователей»

17. Защитить файл для чтения со стороны владельца, проверить.

По умолчанию установились флаги доступа: для пользователя (user) – чтение (r), запись(w); для группы (group) - чтение (r), запись(w); для других (other) – чтение(r).

```
# chmod 264 copy
# ls -l
total 28
drwxrwxr-x  3 root    root    4096 Sep 04 11:06 .
drwxr-xr-x 15 root    root    4096 Sep 04 10:20 ..
drwxrwxr-x  2 root    root    4096 Sep 04 11:07 10
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy1
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy2
-rw-rw-r--  1 root    root      5 Sep 04 10:59 text
#
```

Как видно, только у владельца нет прав чтения для файла test.

18. Защитить файл для чтения со стороны других пользователей, проверить.

Изменим права доступа на запись для файла

```
# chmod 222 copy
# ls -l
total 28
drwxrwxr-x  3 root    root    4096 Sep 04 11:06 .
drwxr-xr-x 15 root    root    4096 Sep 04 10:20 ..
drwxrwxr-x  2 root    root    4096 Sep 04 11:07 10
--w--w--w-  1 root    root      3 Sep 04 11:04 copy
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy1
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy2
-rw-rw-r--  1 root    root      5 Sep 04 10:59 text
#
```

Как видно все не могут читать файл test.

19. Защитить файл для записи со стороны владельца, проверить.

Изменяем права, проверяем

```
# chmod 422 copy
# ls -l
total 28
drwxrwxr-x  3 root    root    4096 Sep 04 11:06 .
drwxr-xr-x 15 root    root    4096 Sep 04 10:20 ..
drwxrwxr-x  2 root    root    4096 Sep 04 11:07 10
-r--w--w-  1 root    root      3 Sep 04 11:04 copy
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy1
-rw-rw-r--  1 root    root      3 Sep 04 11:04 copy2
-rw-rw-r--  1 root    root      5 Sep 04 10:59 text
#
```

20. Защитить файл для записи со стороны других пользователей, проверить.

Изменяем права, проверяем

```
# chmod 244 copy
# ls -l
total 28
drwxrwxr-x 3 root root 4096 Sep 04 11:06 .
drwxr-xr-x 15 root root 4096 Sep 04 10:20 ..
drwxrwxr-x 2 root root 4096 Sep 04 11:07 10
--w-r--r-- 1 root root 3 Sep 04 11:04 copy
-rw-rw-r-- 1 root root 3 Sep 04 11:04 copy1
-rw-rw-r-- 1 root root 3 Sep 04 11:04 copy2
-rw-rw-r-- 1 root root 5 Sep 04 10:59 text
#
```

21. Открыть / закрыть свой основной каталог для доступа со стороны других пользователей, проверить.

Закроем его для доступа всем, кроме владельца.

```
# chmod 700 root/
# ls -l
total 3435849
drwxr-xr-x 15 root root 4096 Sep 04 11:37 .
drwxr-xr-x 15 root root 4096 Sep 04 11:37 ..
-rw----- 1 root root 1541356 Sep 04 2019 .altboot
-r--r--r-- 1 root root 1048234 Sep 04 2019 .bitmap
-rw----- 1 root root 1547148 Sep 04 2019 .boot
-rw-rw-r-- 1 root root 8 Jul 10 2010 .diskroot
-r--r--r-- 1 root root 355328 Sep 04 2019 .inodes
-r--r--r-- 1 root root 49152 Sep 04 2019 .longfilenames
drwxrwxr-x 3 root root 4096 Sep 04 11:06 LAB1
drwxrwxr-x 2 root root 8192 Sep 18 2019 bin
drwxrwxr-x 5 root root 4096 Sep 04 2019 boot
dr-xr-xr-x 2 root root 0 Sep 04 11:39 dev
drwxr-xr-x 15 root root 4096 Sep 04 10:06 etc
-rw-rw-r-- 1 root root 15960 Sep 04 11:37 file.txt
dr-xr-xr-x 2 root root 0 Sep 04 11:39 fs
drwxrwxr-x 3 root root 4096 Sep 04 2019 home
drwxrwxr-x 4 root root 8192 Sep 04 2019 lib
drwxrwxr-x 3 root root 4096 Sep 04 2019 opt
dr-xr-xr-x 2 root root 1754525696 Sep 04 11:39 proc
drwx----- 9 root root 4096 Sep 20 2019 root
drwxrwxr-x 2 root root 8192 Sep 04 2019 sbin
drwxrwxrwt 2 root root 4096 Sep 04 11:31 tmp
drwxrwxr-x 11 root root 4096 Sep 04 2019 usr
drwxrwxr-x 12 root root 4096 Sep 04 2019 var
lrwxrwxrwx 1 root root 1 Sep 04 2019 x86 -> .
#
```

Как видно возможность «исполнить», открыть каталог есть только у владельца.

22. Разрешить доступ к своему основному каталогу, но запретить его изменение, проверить.

Изменение каталога:

```
# chmod 755 root/
# ls -l
total 3435849
drwxr-xr-x 15 root      root      4096 Sep 04 11:37 .
drwxr-xr-x 15 root      root      4096 Sep 04 11:37 ..
-rw----- 1 root      root    1541356 Sep 04 2019 .altboot
-r--r--r-- 1 root      root    1048234 Sep 04 2019 .bitmap
-rw----- 1 root      root    1547148 Sep 04 2019 .boot
-rw-rw-r-- 1 root      root         8 Jul 10 2010 .diskroot
-r--r--r-- 1 root      root    355328 Sep 04 2019 .inodes
-r--r--r-- 1 root      root     49152 Sep 04 2019 .longfilenames
drwxrwxr-x 3 root      root      4096 Sep 04 11:06 LAB1
drwxrwxr-x 2 root      root      8192 Sep 18 2019 bin
drwxrwxr-x 5 root      root      4096 Sep 04 2019 boot
dr-xr-xr-x 2 root      root         0 Sep 04 11:40 dev
drwxr-xr-x 15 root      root      4096 Sep 04 10:06 etc
-rw-rw-r-- 1 root      root     15960 Sep 04 11:37 file.txt
dr-xr-xr-x 2 root      root         0 Sep 04 11:40 fs
drwxrwxr-x 3 root      root      4096 Sep 04 2019 home
drwxrwxr-x 4 root      root      8192 Sep 04 2019 lib
drwxrwxr-x 3 root      root      4096 Sep 04 2019 opt
dr-xr-xr-x 2 root      root    1754525696 Sep 04 11:40 proc
drwxr-xr-x 9 root      root      4096 Sep 20 2019 root
drwxrwxr-x 2 root      root      8192 Sep 04 2019 sbin
drwxrwxrwt 2 root      root      4096 Sep 04 11:31 tmp
drwxrwxr-x 11 root      root      4096 Sep 04 2019 usr
drwxrwxr-x 12 root      root      4096 Sep 04 2019 var
lrwxrwxrwx 1 root      root         1 Sep 04 2019 x86 -> .
```

Все могут зайти в него, посмотреть, но не изменить.

23. Разрешить доступ к файлам только с известными именами, проверить.

Сначала под суперпользователем, для каталога /LAB1 у остальных пользователей были отобраны права для чтения. # chmod o-r LAB1.

```
drwxrwx--x 3 root      root      4096 Sep 18 14:48 LAB1
```

Теперь перелогинившись под обычным юзером, доступа к каталогу /LAB1 нет, но обращаясь к заранее созданному файлу по известному имени, можно считать его содержимое:

```
$ ls /root
ls: Permission denied (/root)
$ ls /root/LAB1
ls: Permission denied (/root/LAB1)
$ ls /root/LAB1/test_23.txt
/root/LAB1/test_23.txt
$ cat /root/LAB1/test_23.txt
Test file for ex.23 :)$
```

2. Создание простых скриптов

1. Написать скрипт, который просто выводит значения переданных ему параметров.

```
# ./s1.sh 2
2
# ./s1.sh 2 sas
2 sas
#
# cat s1.sh

echo $@
```

Результат работы скрипта с 1 аргументом и с 2 различными аргументами, как видно скрипт работает корректно.

2. Написать скрипт, который с помощью утилит *pidin* и *grep* выводит на экран информацию об указанном по имени процессе.

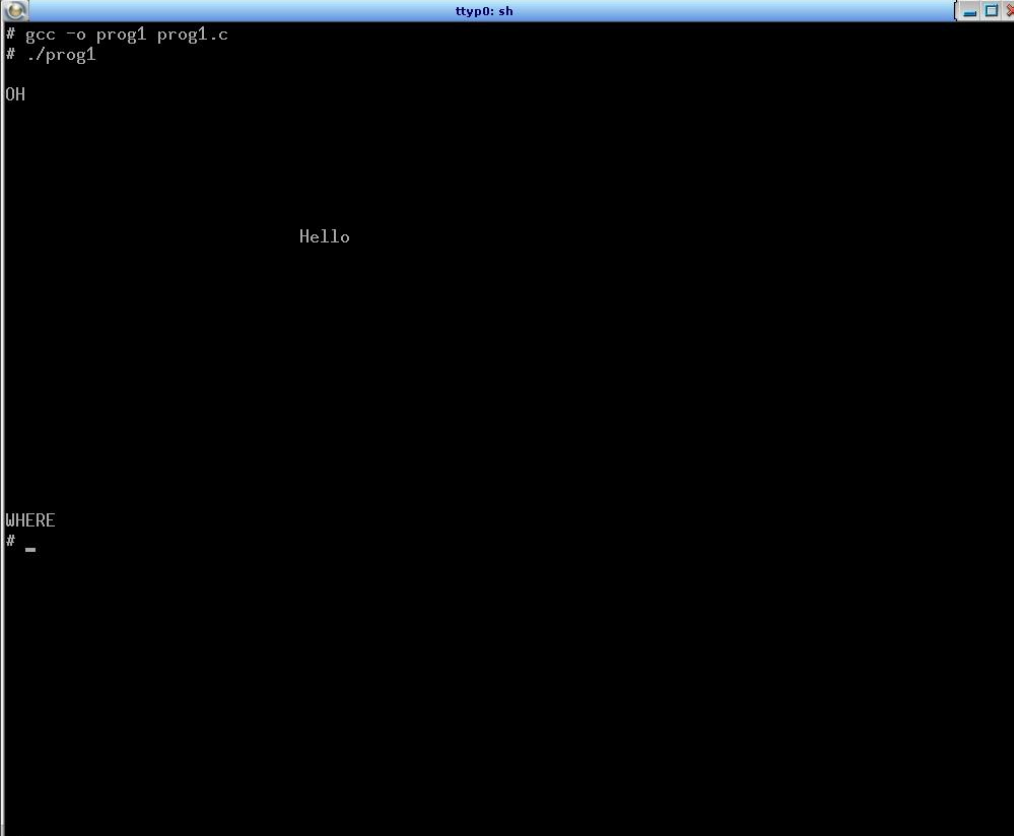
Данный скрипт требует только один аргумент – имя процесса. Получив имя процесса запускается сам скрипт, внутри которого pipeline из команд *pidin* и *grep* результата по переданному имени процесса через аргумент.

```
# cat s2.sh
pidin | grep $1
```

```
# ./s2.sh bin/sh
368670  1 r/photon/bin/shelf  10r RECEIVE      1
368670  2 r/photon/bin/shelf  10r CONDVAR      (0x8076f88)
426013  1 bin/sh               10r SIGSUSPEND
471075  1 bin/sh               10r SIGSUSPEND
#
```


3. Создание простых скриптов

1. Написать программу, выводящую сообщение “HELLO” в центре экрана



```
ttyp0: sh
# gcc -o prog1 prog1.c
# ./prog1
OH
Hello
WHERE
# -
```

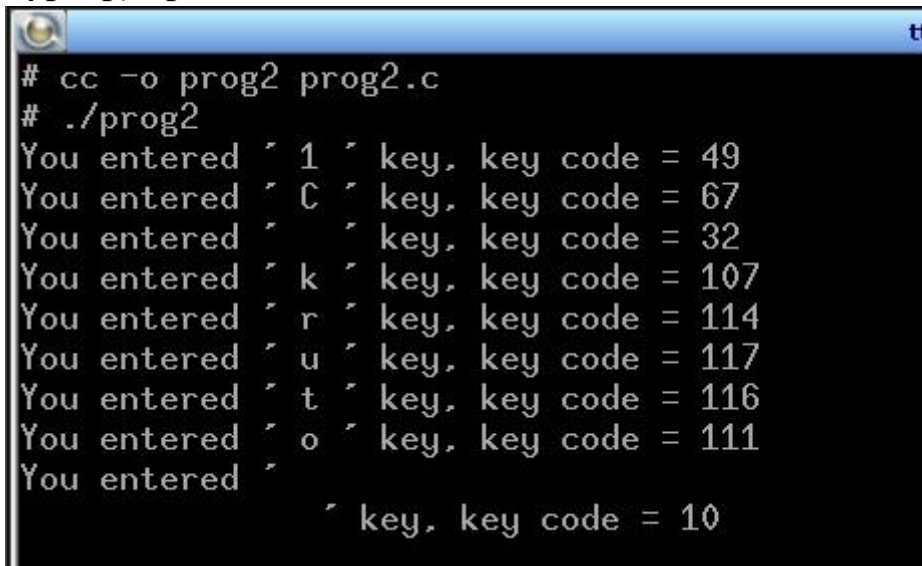


```
ttyp0: sh
#include <stdio.h>

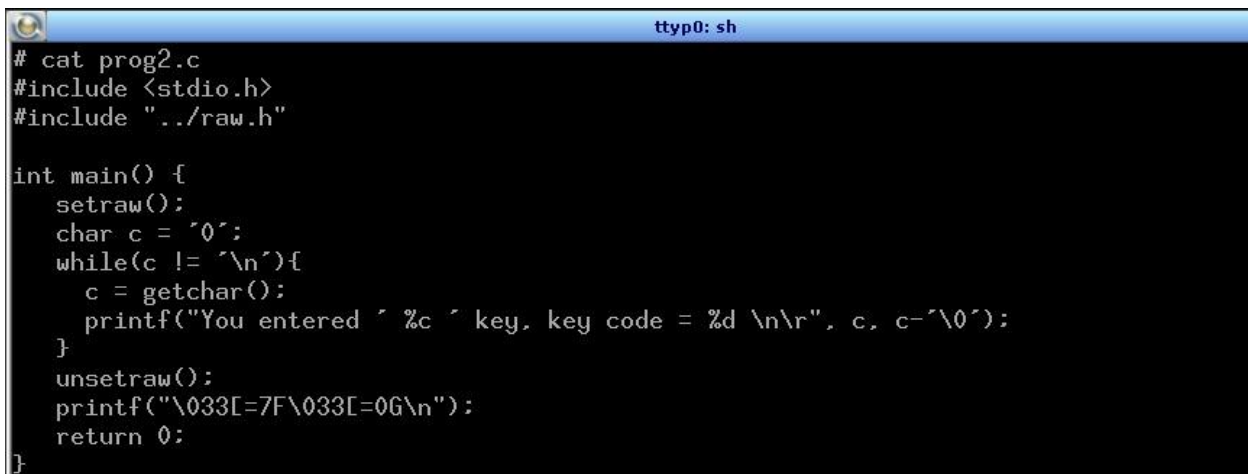
int main()
{
    printf("\nOH\033[11;30HHello\033[25;1HWHERE\n");
    return 0;
}
```

Для выполнения задания была написана программа с использованием esc-последовательности: «\033[y;xH» - установка курсора в позицию (x,y).

2. Написать программу, позволяющую определять коды нажимаемых клавиш и восстанавливающую исходный вид терминала (цвет, курсор) при выходе.



```
# cc -o prog2 prog2.c
# ./prog2
You entered ^ 1 ^ key, key code = 49
You entered ^ C ^ key, key code = 67
You entered ^      ^ key, key code = 32
You entered ^ k ^ key, key code = 107
You entered ^ r ^ key, key code = 114
You entered ^ u ^ key, key code = 117
You entered ^ t ^ key, key code = 116
You entered ^ o ^ key, key code = 111
You entered ^      ^ key, key code = 10
```

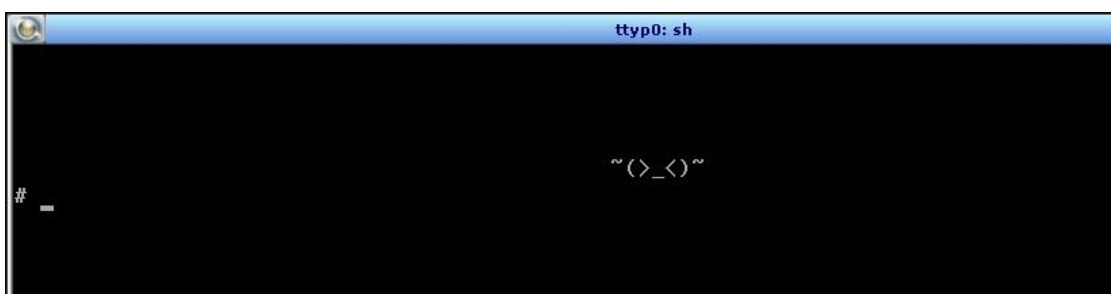


```
# cat prog2.c
#include <stdio.h>
#include "../raw.h"

int main() {
    setraw();
    char c = '\0';
    while(c != '\n'){
        c = getchar();
        printf("You entered ^ %c ^ key, key code = %d \n\r", c, c-'\0');
    }
    unsetraw();
    printf("\033[7F\033[0G\n");
    return 0;
}
```

3. Написать программу, рисующую движущийся символ (при выключенном курсоре, без использования функции стирания экрана).

Для выполнения данного задания, была реализована программа, где вначале идет отключение курсора специальной последовательностью, затем начинается цикл от 1 до 25 для движения по строкам, там идет отрисовка символа, ожидание 50000usec, сдвиг обратно, затирание символа и далее заново по циклу.



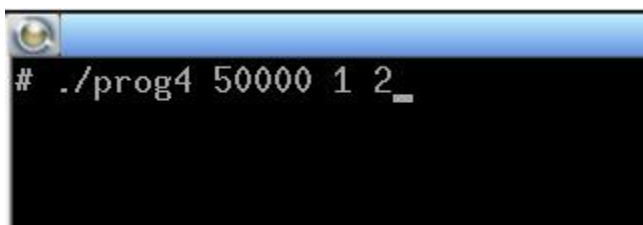
```
# _ ~(>_<)~
```

```
# cat prog3.c
#include <stdio.h>
#include <unistd.h>

int main(){
    int i;
    printf("\033[?25l");
    for(i = 1; i <=45; i++){
        printf("\033[2J");
        printf("\033[5;%dH",i);
        printf("~(>_<)~");
        fflush(stdout);
        usleep(50000);
    }
    printf("\n\033[?25h");
    return 0;
}
# _
```

4. Написать программу, рисующую бесконечно движущийся символ. Характер движения (скорость, направление, цвет и т.д.) задавать с помощью параметров командной строки. Предусмотреть восстановление параметров дисплея (цвет, курсор) при принудительном завершении программы. Осуществить запуск нескольких экземпляров программы с разными параметрами движения (запуск с одного терминала, вывод на другой).

Для выполнения данного задания была написана программа, в которой сначала идет считывание аргументов запуска – скорости, направления и цвета символа. Если аргументов не 3 или они неправильные – программа завершается с ошибкой.





```
ttyp0: sh
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <stdlib.h>

void suicide(int sig)
{
    printf("\033[?25h\033[=0G\033[=7F\n");
    exit(0);
}

void error_suicide()
{
    printf("\033[?25h\033[=0G\033[=7F\n");
    exit(-1);
}

int main(int argc, char *argv[])
{
    signal(SIGINT, suicide);
    if(argc != 4){
        printf("Too few arguments");
        error_suicide();
    }
    int speed = atoi(argv[1]);
    int direction = atoi(argv[2]);
    int color = atoi(argv[3]);

    if(direction < 1 || direction > 4) {
        printf("Invalid direction");
        error_suicide();
    }
    if(color < 0 || color > 7) {
        printf("Invalid color");
        error_suicide();
    }
    if(speed < 1) {
        printf("Invalid speed");
        error_suicide();
    }
}
```

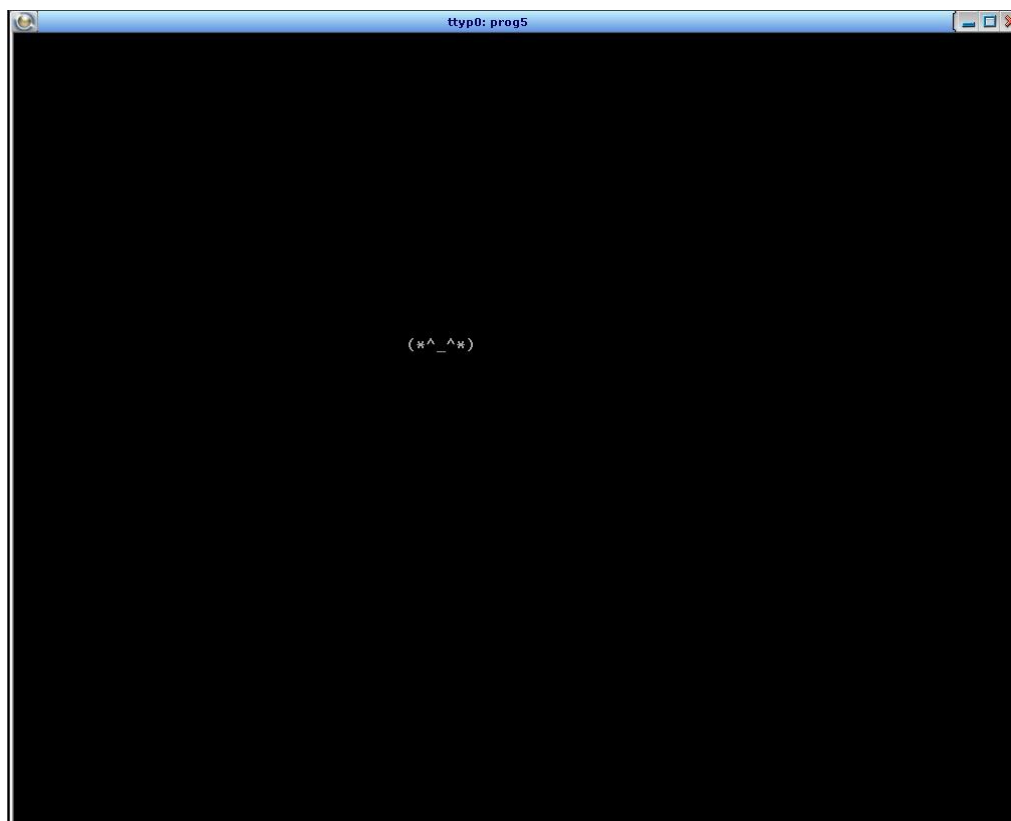
```

int dx = 0, dy = 0;
if(direction == 1)
    dy = -1;
if(direction == 2)
    dx = -1;
if(direction == 3)
    dy = 1;
if(direction == 4)
    dx = 1;
printf("\033[=dF\033[?25l",color);
int x, y;
for(x = 40, y = 15; ;x += dx, y += dy)
{
    printf("\033[2J");
    printf("\033[%d;%dH", y, x);
    printf("(^_^)");
    fflush(stdout);
    usleep(speed);
    if(y+dy < 0 || y+dy > 42)
        y = y - 42 * dy;
    if(x+dx < 0 || x+dx > 80)
        x = x - 80 * dx;
}
return 0;
}

```

5. Программно реализовать команду по заданию преподавателя.

Была модифицирована программа из 4го и 3го заданий – добавлена возможность управлять символом при помощи клавиш на клавиатуре.



```
ttty0: less
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <stdlib.h>
#include "../raw.h"

void suicide(int sig)
{
    printf("\033[?25h\033[0G\033[7F\n");
    exit(0);
}

int main()
{
    signal(SIGINT, suicide);
    int speed = 0;
    int direction = 0;
    int color = 0;

    int dx = 0, dy = 0;
    int x = 40, y = 15;
    printf("\033[?25l");
    setraw();
    char c = '^0';
    while (c != '\n'){
        c = getchar();
        dx = 0;
        dy = 0;
        if(c == 49)
            dx = -1;
        if(c == 71)
            dy = -1;
        if(c == 52)
            dx = 1;
        if(c == 67)
            dy = 1;
        x = x + dx;
        y = y + dy;
        if(dx != 0 || dy != 0){
```

```
            printf("\033[2J");
            printf("\033[%d;%dH", y, x);
            printf("( * ^ * )");
            if(y+dy < 0 || y+dy > 42)
                y = y - 42 * dy;
            if(x+dx < 0 || x+dx > 80)
                x = x - 80 * dx;
        }
        fflush(stdout);
        usleep(50000);
    }
    unsetraw();
    printf("\033[7F\033[0G\033[?25h\n");
    return 0;
}
(END)
```