

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

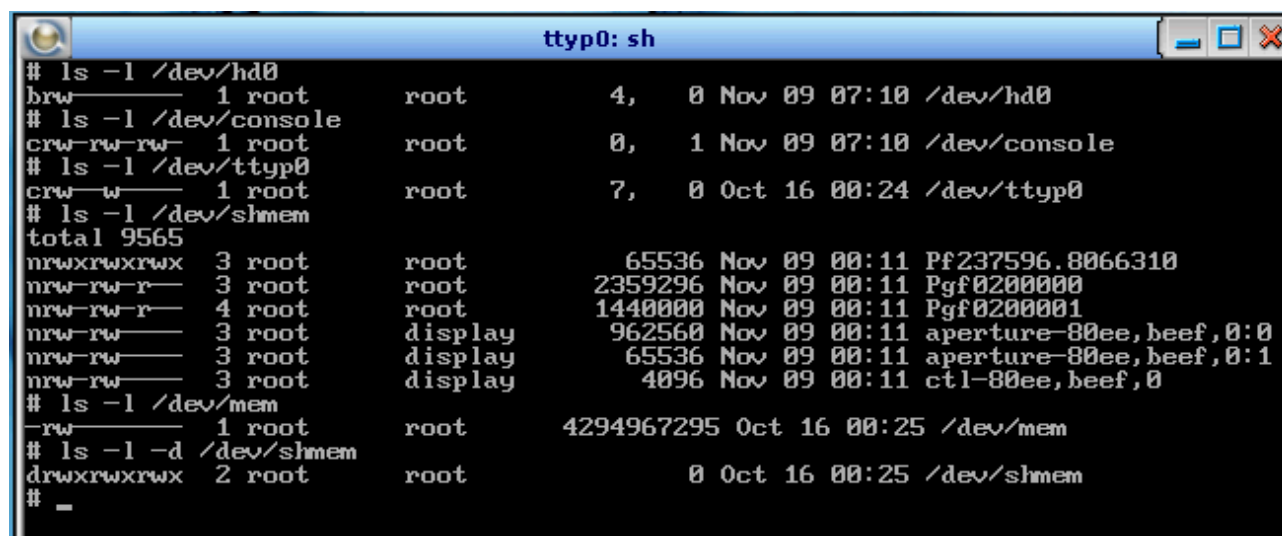
ЛАБОРАТОРНАЯ РАБОТА № 1

Выполнил:
Студент группы: ИП-712
Алексеев С.В.

Проверил: профессор кафедры ПМиК
Фионов А.Н.

1. Работа с командной строкой (составить протокол по выполнению всех пунктов)

Определить тип файлов /dev/hd0, /dev/console, /dev/tty0, /dev/shmem, /dev/mem.



```
tty0: sh
# ls -l /dev/hd0
brw-rw-rw- 1 root root 4, 0 Nov 09 07:10 /dev/hd0
# ls -l /dev/console
crw-rw-rw- 1 root root 0, 1 Nov 09 07:10 /dev/console
# ls -l /dev/tty0
crw-w--w- 1 root root 7, 0 Oct 16 00:24 /dev/tty0
# ls -l /dev/shmem
total 9565
nrwxrwxrwx 3 root root 65536 Nov 09 00:11 Pf237596.8066310
nrw-rw-r-- 3 root root 2359296 Nov 09 00:11 Pg0200000
nrw-rw-r-- 4 root root 1440000 Nov 09 00:11 Pg0200001
nrw-rw-r-- 3 root display 962560 Nov 09 00:11 aperture-80ee,beef,0:0
nrw-rw-r-- 3 root display 65536 Nov 09 00:11 aperture-80ee,beef,0:1
nrw-rw-r-- 3 root display 4096 Nov 09 00:11 ctl-80ee,beef,0
# ls -l /dev/mem
-rw-rw-rw- 1 root root 4294967295 Oct 16 00:25 /dev/mem
# ls -l -d /dev/shmem
drwxrwxrwx 2 root root 0 Oct 16 00:25 /dev/shmem
# -
```

Каталог /dev принадлежит менеджеру процессов и содержит файлы устройств

/dev/hd0 тип файла: блочное устройство (Block special file)

Жесткие диски и разделы дисков, являются представлены специальными файлами, называемыми блочными устройствами. Эти файлы могут быть записаны и прочитаны случайным образом, чтобы считывайте и манипулируйте содержимым диска. Блочные устройства-это обозначается буквой b в первом символе списка ls-l

/dev/console тип файла: символьное устройство (Character special file)

Специальные файлы, называемые символьными устройствами представляют собой еще один вид устройств ввода-вывода. Подобно блочным устройствам, эти устройства устройства можно считывать и манипулировать ими, но в случае символа устройства, они должны быть прочитаны и записаны последовательно, а не случайным образом. Символьные устройства обозначаются буквой c в первом столбце выход ls-l.

/dev/tty0 тип файла: символьное устройство (Character special file)

Виртуальное устройство, принадлежащее менеджеру процессов (procnto), которое преобразуется в управляющее терминальное устройство, связанное с сеансом любого процесса, открывающего файл.

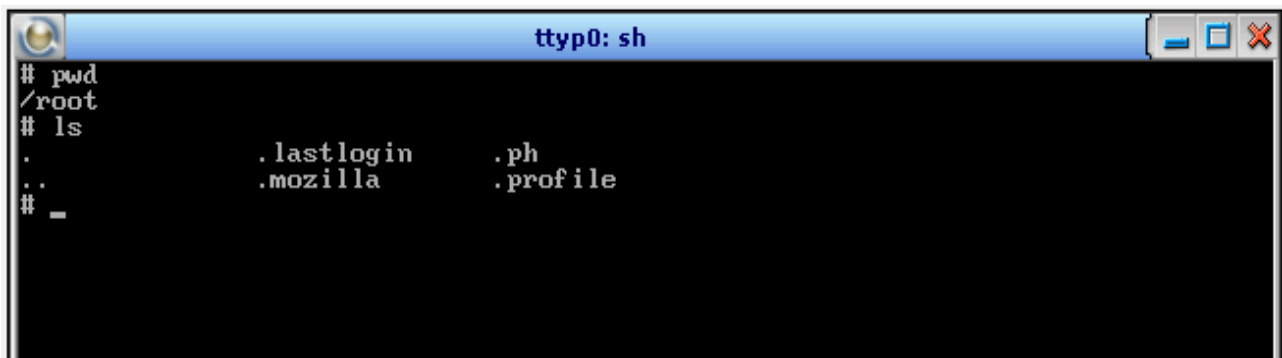
/dev/shmem тип файла: каталог (Directory)

Содержит файлы, представляющие области общей памяти в системе (также иногда используется для общих файлов с отображением памяти)

/dev/mem тип файла: обычный файл (Regular file)

Устройство, представляющее всю физическую память

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

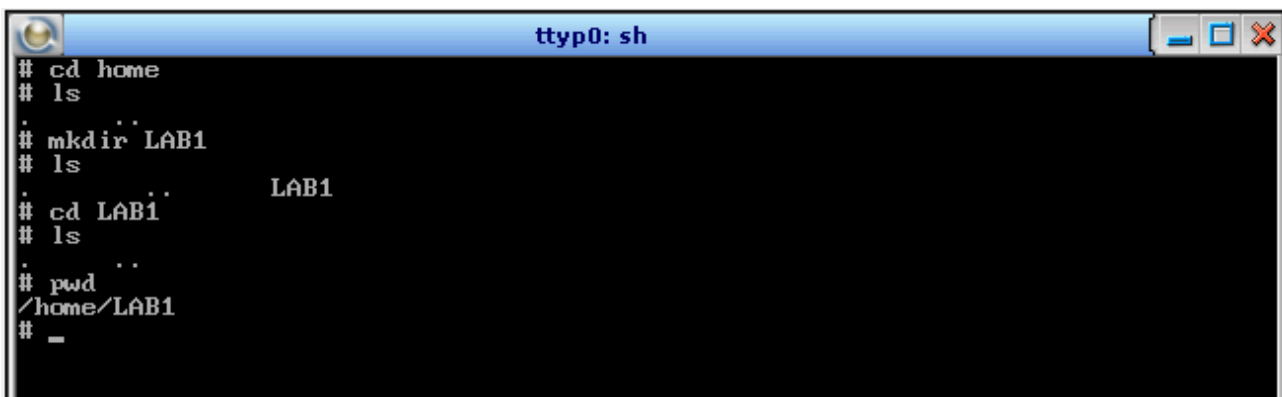


```
ttty0: sh
# pwd
/root
# ls
.      .lastlogin  .ph
..     .mozilla   .profile
# _
```

каталог /root

При первой установке ОСРВ QNX Neutrino автоматически создается единственная учетная запись пользователя с именем root.

3) Создать каталог LAB1 и сделать его рабочим.



```
ttty0: sh
# cd home
# ls
.      ..
# mkdir LAB1
# ls
.      ..      LAB1
# cd LAB1
# ls
.      ..
# pwd
/home/LAB1
# _
```

Определить (с помощью программы ls), в каком каталоге содержится файл

```
ttyp0: sh
# cd
# ls
.          .lastlogin  .ph
..         .mozilla   .profile
# ls -l */services
ls: No such file or directory (*/services)
# cd ..
# ls
.          bin          fs          proc         usr
..         boot         home        root         var
.boot     dev          lib         shin         x86
.diskroot etc         opt         tmp
# ls -l */services
-rw-r--r-- 1 root      root      4125 Oct 13  2007 etc/services
# _
```

Файл Services содержится в каталоге etc/

```
ttyp0: sh
https      443/tcp          # Default HTTPS port
hostnames  101/tcp          hostname         # usually to sri-nic
x400       103/tcp          # ISO Mail
x400-snd   104/tcp
nethios-ns 137/udp          # NETBIOS Name Server
nethios-dgm 138/udp        # NETBIOS Datagram Service
nethios-ssn 139/udp       # NETBIOS Session Service
nethios-ssn 139/tcp
News       144/tcp          news            # Window System
snmp       161/udp          # network management
dirsrv     1525/udp
nfsd       2049/tcp
nfsd       2049/udp
webster    2627/tcp
infleet    5999/tcp
xserver    6000/tcp
irc        6667/tcp
phrelay    4868/tcp
phrelaydbg 4869/tcp
phindemo   4870/tcp
socks      1080/tcp
timesrv    22375/tcp
althttp    25080/tcp          # time server process - RES
dir_svc    33333/tcp          # alternate www port
# _
```

Содержимое файла services можно посмотреть командой: cat ect/services

Сколько скрытых файлов в вашем домашнем каталоге?

```
ttyp0: sh
# cd
# ls -a
.          .lastlogin  .ph
..         .mozilla    .profile
# ls -a -l
total 9
drwx----- 4 root      root      1024 Oct 14 18:23 .
drwxr-xr-x 14 root      root      1024 Oct 18 19:51 ..
-rw-rw-r-- 1 root      root         0 Nov 09 00:11 .lastlogin
drwx----- 3 root      root      1024 Oct 14 18:23 .mozilla
drwxrwxr-x 5 root      root      1024 Nov 09 00:16 .ph
-rw-r--r-- 1 root      root       191 Apr 20 2001 .profile
# ls_
```

Посмотреть скрытые файлы в каталоге можно командой: `ls -a`
Как видим в домашнем каталоге 4 скрытых файла

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

```
ttyp0: sh
# ls -l /boot
total 10
drwxrwxr-x 5 root      root      1024 Oct 15 01:13 .
drwxr-xr-x 14 root      root      1024 Oct 18 19:51 ..
drwxrwxr-x 2 root      root      1024 Oct 15 01:13 build
drwxrwxr-x 2 root      root      1024 Oct 15 01:13 fs
drwxrwxr-x 2 root      root      1024 Oct 15 01:13 sys
# _
```

Подкаталоги в каталоге `/boot`: `build`, `fs` и `sys`

```
ttyp0: sh
# ls -l /boot/build
total 36
drwxrwxr-x 2 root      root      1024 Oct 15 01:13 .
drwxrwxr-x 5 root      root      1024 Oct 15 01:13 ..
-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
# _
```

```
ttyp0: sh
# ls -l /boot/fs
total 9048
drwxrwxr-x 2 root root 1024 Oct 15 01:13 .
drwxrwxr-x 5 root root 1024 Oct 15 01:13 ..
-rw-rw-r- 1 root root 1541356 Oct 15 01:13 gnxbase.ifs
-rw-rw-r- 1 root root 1541356 Oct 15 01:13 gnxbasedma.ifs
-rw-rw-r- 1 root root 1547148 Oct 15 01:13 gnxbasesmp.ifs
# _
```

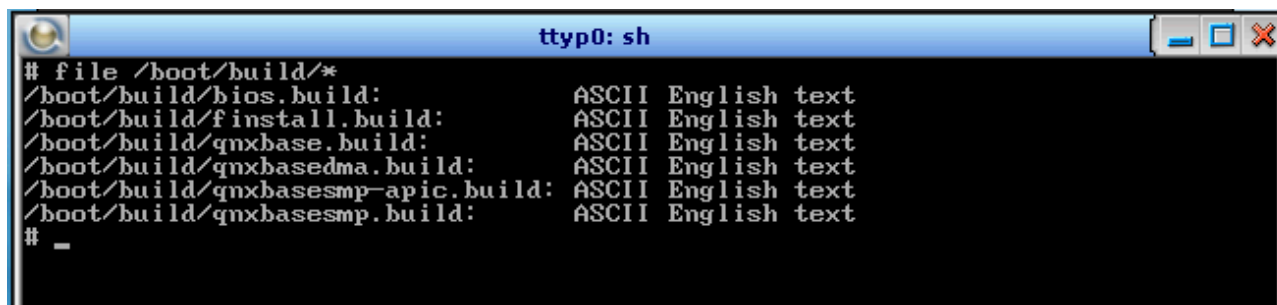
```
ttyp0: sh
# ls -l /boot/sys
total 12404
drwxrwxr-x 2 root root 1024 Oct 15 01:13 .
drwxrwxr-x 5 root root 1024 Oct 15 01:13 ..
-rwxr-xr-x 1 root root 3642 Jul 10 2010 bios.boot
-rwxr-xr-x 1 root root 3667 Jul 10 2010 bios16m.boot
-rwxr-xr-x 1 root root 3618 Jul 10 2010 bios_nokhd.boot
-rw-r--r- 1 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
# _
```

Видим, что в данных подкаталогах всего 6 файлов (elf.boot, ipl-diskpc1, меньше 1 Кбайта или 1024 байт.

Исполняемыми файлами называются файлы, содержащие в себе готовые к запуску компьютерные программы.

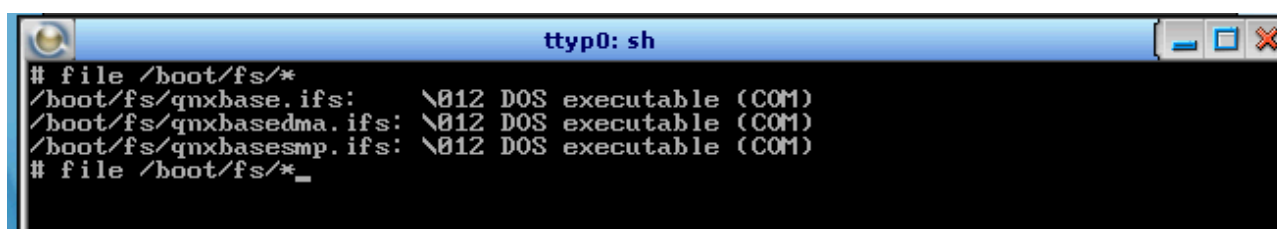
Стандартный исполняемый файл Executable and Linking Format ELF формата содержит множество сегментов, например такие как: *.text* - сегмент кода, *.data* - сегмент инициализированных данных (обычно только для чтения, сюда попадают строки, константные массивы и т.п.), *.bss* - сегмент неинициализированных данных программы, *.dynamic* - сегмент служебных данных о подгружаемых библиотеках во время запуска программы, *.comment* - информация о версиях, *.init* - код инициализации процесса, *.fini* - код окончания выполнения процесса и т.д. Этих сегментов большое множество, но в основном они все стандартные и несут в себе

информацию, необходимую для правильной загрузки исполняемого файла в память и требуемых им библиотек. Также есть сегменты, отвечающие за хранение отладочной информации (*.debug*) и прочих опциональных данных выполнения исполняемого файла и могут быть безопасно удалены.

A terminal window titled 'tty0: sh' showing the output of the command '# file /boot/build/*'. The output lists five files in the /boot/build directory, all identified as ASCII English text files.

```
tty0: sh
# file /boot/build/*
/boot/build/bios.build:      ASCII English text
/boot/build/finstall.build:  ASCII English text
/boot/build/qnxbase.build:   ASCII English text
/boot/build/qnxbasedma.build: ASCII English text
/boot/build/qnxbasesmp-apic.build: ASCII English text
/boot/build/qnxbasesmp.build: ASCII English text
# _
```

В файле /build нет исполняемых файлов

A terminal window titled 'tty0: sh' showing the output of the command '# file /boot/fs/*'. The output lists three files in the /boot/fs directory, all identified as \012 DOS executable (COM) files.

```
tty0: sh
# file /boot/fs/*
/boot/fs/qnxbase.ifs:  \012 DOS executable (COM)
/boot/fs/qnxbasedma.ifs: \012 DOS executable (COM)
/boot/fs/qnxbasesmp.ifs: \012 DOS executable (COM)
# file /boot/fs/*_
```

В файле /fs три исполняемых файла с расширением .ifs

```
ttty0: sh
# file /boot/sys/*
/boot/sys/bios.boot:      ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, 1687698518 bytes lazy stack, 1792077824 bytes prealloc
ated stack, not stripped
/boot/sys/bios16m.boot:   ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, -490077881 bytes lazy stack, 1975136256 bytes prealloc
ated stack, not stripped
/boot/sys/bios_nokbd.boot: ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, 2062819058 bytes lazy stack, -462626816 bytes prealloc
ated stack, not stripped
/boot/sys/elf.boot:       ASCII text
/boot/sys/ipl-diskpc1:    \012 DOS executable (COM)
/boot/sys/ipl-diskpc1-flop: \012 DOS executable (COM)
/boot/sys/ipl-diskpc2:    \012 DOS executable (COM)
/boot/sys/ipl-diskpc2-flop: \012 DOS executable (COM)
/boot/sys/libmod_aps.a:   current ar archive
/boot/sys/nobios.boot:    ASCII text
/boot/sys/procnto:        ELF 32-bit LSB relocatable, Intel 80386, version 1
(SYSV), not stripped
/boot/sys/procnto-instr:  ELF 32-bit LSB relocatable, Intel 80386, version 1
(SYSV), not stripped
/boot/sys/procnto-smp:    ELF 32-bit LSB relocatable, Intel 80386, version 1
(SYSV), not stripped
/boot/sys/procnto-smp-instr: ELF 32-bit LSB relocatable, Intel 80386, version 1
(SYSV), not stripped
/boot/sys/startup-apic:   ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, fullpath /home/builder/hudson/6.5.x_milestone/svn/hard
ware/startup/boards/apic/x86/o/startup-apic, not stripped
/boot/sys/startup-bios:   ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, fullpath /home/builder/hudson/6.5.x_milestone/svn/hard
ware/startup/boards/bios/x86/o/startup-bios, not stripped
/boot/sys/startup-bios-32: ELF 32-bit LSB executable, Intel 80386, version 1 (
SYSV), statically linked, fullpath /home/builder/hudson/6.5.x_milestone/svn/hard
ware/startup/boards/bios/x86/o.32/startup-bios-32, not stripped
# _
```

В файле /sys 10 файлов ELF формата

Сколько жестких связей у каталога /boot и почему?

Жесткими связями у каталога называются структурные составляющие файла — описывающий его элемент каталога

```
ttty0: sh
# ls -l /boot
total 10
drwxrwxr-x  5 root    root    1024 Oct 15 01:13 .
drwxr-xr-x 14 root    root    1024 Oct 18 19:51 ..
drwxrwxr-x  2 root    root    1024 Oct 15 01:13 build
drwxrwxr-x  2 root    root    1024 Oct 15 01:13 fs
drwxrwxr-x  2 root    root    1024 Oct 15 01:13 sys
# _
```

t
o
t
a
l

отражает количество существующих по данному пути указателей нас
Создать текстовый файл с помощью редактора vi. Какие флаги доступа
существующие файлы.
устанавливаются у вновь создаваемого файла? Почему? Как это исправить?

Создали файл командой: `vi lab1`

```
ttyp0: sh
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
# ls
.      ..      lab1
# pwd
/home/LAB1
# ls -l
total 5
drwxrwxr-x 2 root    root        1024 Oct 16 02:29 .
drwxrwxr-x 3 root    root        1024 Oct 16 00:59 ..
-rw-rw-r-- 1 root    root          6 Oct 16 02:29 lab1
# _
```

Флаги доступа у файла:

$$\begin{array}{c} \mathbf{r} \\ \mathbf{w} \\ \mathbf{r} \\ \mathbf{w} \end{array}$$

Чтобы исправить это, нужно использовать команду UMASK “значение новой маски”. Она означает, что владелец файла и члены группы, к которой принадлежит владелец имеют право на чтение и запись, а остальные пользователи имеют право только на чтение файла.

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.

Создаем каталог «PAR9», создаем файл «file» в данном каталоге и делаем десять копий этого файла

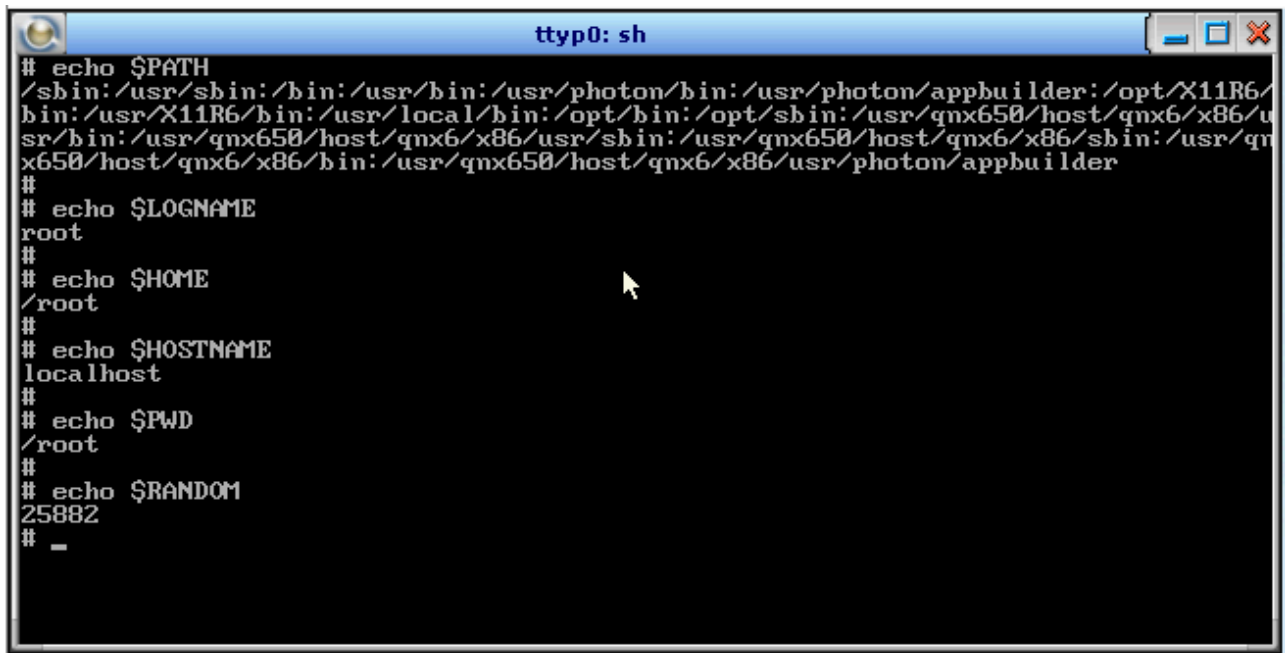
```
ttyp0: sh
# mkdir PAR9
# ls
.          PAR9      lab1
..
# cd PAR9
# ls
.          ..
# touch file
# ls
.          ..      file
# cp file file1
# ls
.          ..      file      file1
# cp file file2
# cp file file3
# cp file file4
# ls
.          ..      file      file1      file2      file3      file4
# cp file file5
# cp file file6
# cp file file7
# cp file file8
# cp file file9
# cp file file10
# ls
.          ..      file      file1      file10      file3      file5      file7      file9
..         file1      file2      file4      file6      file8
# _
```

Перемещаем три файла (file10, file9, file8) в вышестоящий каталог

И удаляем некоторые файлы из каталога PAR9 командой «rm» с опцией -i, для запрашивания подтверждения перед удалением каждого существующего файла

```
ttyp0: sh
# mv file10 file9 file8 ..
# ls
.          file      file2      file4      file6
..         file1      file3      file5      file7
# rm -i file1 file3 file5 file7
rm: remove file3? (y/N) n
rm: remove file5? (y/N) n
rm: remove file7? (y/N) n
# ls
.          file      file2      file4      file6
..         file1      file3      file5      file7
# rm -i file1 file3 file5 file7
rm: remove file1? (y/N) y
rm: remove file3? (y/N) y
rm: remove file5? (y/N) y
rm: remove file7? (y/N) y
# ls
.          ..      file      file2      file4      file6
# _
```

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/usr/bin:/usr/qnx650/host/qnx6/x86/usr/sbin:/usr/qnx650/host/qnx6/x86/sbin:/usr/qnx650/host/qnx6/x86/bin:/usr/qnx650/host/qnx6/x86/usr/photon/appbuilder
#
# echo $LOGNAME
root
#
# echo $HOME
/root
#
# echo $HOSTNAME
localhost
#
# echo $PWD
/root
#
# echo $RANDOM
25882
# -
```

Переменная PATH пользователя root включает в себя каталоги, которые содержат системные исполняемые файлы. Меняется вручную

L

О
Н
С
C
Z
M
H
P
D
R
A
M

содержит имя домашнего каталога под которым зашел пользователь. Меняется в зависимости какой пользователь пошел в систему

показывает имя пользователя под которым он вошел в систему. Меняется от пользователя

показывает текущий рабочий каталог. Меняется в зависимости от того, в каком каталоге работаем

Определить коды завершения команд ls /bin и ls /pin

Показывает текущую хост-систему. Не меняется

M

генерирует случайное десятичное число. Меняется с каждым запуском команды

```
ttyp0: sh
# ls /bin
.                elvis            mkifsfs_elf      sloginfo
..               esh             mkifsfs_openbios split
aps              ex             mkifsfs_srec     stty
asa              false          mkxfs            su
cat              fesh           more            sync
chgrp            gunzip         mount           true
chmod            gzip           mv              uesh
chown            hostname       netmanager      umount
confstr          igawk          ln              uname
cp               kill           pax             uncompress
cpio             ksh           pidin           vi
csplit           ln            ps              view
dd               login          pwd             waitfor
df               logout         rm              who
disconf          ls             script          zcat
du               mkdir          sendnto
dumpifs          mkefs         sh
echo             mkefs         shutdown
ed               mkifs         slay
# _
```

```
ttyp0: sh
# ls /pin
ls: No such file or directory (/pin)
# _
```

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

Создали файл “bin” в каталоге “PAR9” и вывели в него содержимое каталога /bin в несколько колонок с помощью команд:

```
#touch bin
```

```
#ls -l -h /bin >bin
```

```

t-u-g-o In Owner      Group      Size Date      Filename
total 4368
drwxrwxr-x 2 root      root      3072 Oct 15 01:13 .
drwxr-xr-x 14 root      root      1024 Oct 18 19:51 ..
-rwxr-xr-x 1 root      root      67656 Jul 10 2010 aps
-rwxr-xr-x 1 root      root      6721 Jul 10 2010 asa
-rwxr-xr-x 1 root      root      8125 Jul 10 2010 cat
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 chgrp -> chown
-rwxr-xr-x 1 root      root      13578 Jul 10 2010 chmod
-rwxr-xr-x 1 root      root      11197 Jul 10 2010 chown
-rwxr-xr-x 1 root      root      7381 Jul 10 2010 confstr
-rwxr-xr-x 1 root      root      41136 Jul 10 2010 cp
lrwxrwxrwx 1 root      root      3 Oct 15 01:13 cpio -> pax
-rwxr-xr-x 1 root      root      42534 Jul 10 2010 csplit
-rwxr-xr-x 1 root      root      17261 Jul 10 2010 dd
-rwxr-xr-x 1 root      root      16041 Jul 10 2010 df
-rwxr-xr-x 1 root      root      24194 Jul 10 2010 dispconf
-rwxr-xr-x 1 root      root      11658 Jul 10 2010 du
-rwxr-xr-x 1 root      root      32745 Jul 10 2010 dumpifs
-rwxr-xr-x 1 root      root      7250 Jul 10 2010 echo
-rwxr-xr-x 1 root      root      83449 Jul 10 2010 ed
-rwxr-xr-x 1 root      root      481278 Jul 10 2010 elvis
-rwxr-xr-x 1 root      root      19270 Jul 10 2010 esh
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 ex -> elvis
-rwxr-xr-x 1 root      root      5432 Jul 10 2010 false
-rwxr-xr-x 1 root      root      22968 Jul 10 2010 fesh
-rwxr-xr-x 1 root      root      112 Jul 10 2010 gunzip
-rwxr-xr-x 1 root      root      72831 Jul 10 2010 gzip
-rwxr-xr-x 1 root      root      6746 Jul 10 2010 hostname
-rwxr-xr-x 1 root      root      3089 Jul 10 2010 igawk
-rwxr-xr-x 1 root      root      9481 Jul 10 2010 kill
-rwxr-xr-x 1 root      root      203834 Jul 10 2010 ksh
-rwxr-xr-x 1 root      root      13055 Jul 10 2010 ln
-rwsrwxr-x 1 root      root      53654 Jul 10 2010 login
-rwxr-xr-x 1 root      root      5581 Jul 10 2010 logout
-rwxr-xr-x 1 root      root      20745 Jul 10 2010 ls
-rwxr-xr-x 1 root      root      10724 Jul 10 2010 mkdir
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mkefs -> mkxfs
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mketfs -> mkxfs
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mkifs -> mkxfs
-rwxr-xr-x 1 root      root      20651 Jul 10 2010 mkifsf_elf
-rwxr-xr-x 1 root      root      14428 Jul 10 2010 mkifsf_openbios
-rwxr-xr-x 1 root      root      18735 Jul 10 2010 mkifsf_srec

```

```
ttyp0: sh
-rwxr-xr-x 1 root root 13055 Jul 10 2010 ln
-rwsrwxr-x 1 root root 53654 Jul 10 2010 login
-rwxr-xr-x 1 root root 5581 Jul 10 2010 logout
-rwxr-xr-x 1 root root 20745 Jul 10 2010 ls
-rwxr-xr-x 1 root root 10724 Jul 10 2010 mkdir
lrwxrwxrwx 1 root root 5 Oct 15 01:13 mkefs -> mkxfs
lrwxrwxrwx 1 root root 5 Oct 15 01:13 mketfs -> mkxfs
lrwxrwxrwx 1 root root 5 Oct 15 01:13 mkifs -> mkxfs
-rwxr-xr-x 1 root root 20651 Jul 10 2010 mkisfs_elf
-rwxr-xr-x 1 root root 14428 Jul 10 2010 mkisfs_openbios
-rwxr-xr-x 1 root root 18735 Jul 10 2010 mkisfs_srec
-rwxr-xr-x 1 root root 262620 Jul 10 2010 mkxfs
lrwxrwxrwx 1 root root 15 Oct 15 01:13 more -> ../usr/bin/less
-rwxr-xr-x 1 root root 18322 Jul 10 2010 mount
-rwxr-xr-x 1 root root 12408 Jul 10 2010 mv
-rwxr-xr-x 1 root root 56198 Jul 10 2010 netmanager
-rwxr-xr-x 1 root root 27035 Jul 10 2010 on
-rwxr-xr-x 1 root root 54715 Jul 10 2010 pax
-rwxr-xr-x 1 root root 72148 Jul 10 2010 pidin
-rwxr-xr-x 1 root root 25413 Jul 10 2010 ps
-rwxr-xr-x 1 root root 6030 Jul 10 2010 pwd
-rwxr-xr-x 1 root root 12161 Jul 10 2010 rm
-rwxr-xr-x 1 root root 11712 Jul 10 2010 script
-rwxr-xr-x 1 root root 20556 Jul 10 2010 sendnto
lrwxrwxrwx 1 root root 3 Oct 15 01:13 sh -> ksh
-rwxr-xr-x 1 root root 15050 Jul 10 2010 shutdown
-rwxr-xr-x 1 root root 19945 Jul 10 2010 slay
-rwxr-xr-x 1 root root 11091 Jul 10 2010 sloginfo
-rwxr-xr-x 1 root root 90388 Jul 10 2010 split
-rwxr-xr-x 1 root root 22821 Jul 10 2010 stty
-rwsrwxr-x 1 root root 53945 Jul 10 2010 su
-rwxr-xr-x 1 root root 5789 Jul 10 2010 sync
-rwxr-xr-x 1 root root 5420 Jul 10 2010 true
-rwxr-xr-x 1 root root 13116 Jul 10 2010 uesh
-rwxr-xr-x 1 root root 6557 Jul 10 2010 umount
-rwxr-xr-x 1 root root 7743 Jul 10 2010 uname
lrwxrwxrwx 1 root root 6 Oct 15 01:13 uncompress -> gunzip
lrwxrwxrwx 1 root root 5 Oct 15 01:13 vi -> elvis
lrwxrwxrwx 1 root root 5 Oct 15 01:13 view -> elvis
lrwxrwxrwx 1 root root 2 Oct 15 01:13 waitfor -> on
-rwxr-xr-x 1 root root 7921 Jul 10 2010 who
-rwxr-xr-x 1 root root 113 Jul 10 2010 zcat
#
#
```

Затем добавили к файлу bin распечатку каталога /usr/bin командой

```
ls -l -h /usr/bin >>bin
```

```

ttyp0: more
t-u-g-o  ln Owner      Group      Size Date      Filename
total 4368
drwxrwxr-x 2 root      root      3072 Oct 15 01:13 .
drwxr-xr-x 14 root      root      1024 Oct 18 19:51 ..
-rwxr-xr-x 1 root      root      67656 Jul 10 2010 aps
-rwxr-xr-x 1 root      root      6721 Jul 10 2010 asa
-rwxr-xr-x 1 root      root      8125 Jul 10 2010 cat
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 chgrp -> chown
-rwxr-xr-x 1 root      root      13578 Jul 10 2010 chmod
-rwxr-xr-x 1 root      root      11197 Jul 10 2010 chown
-rwxr-xr-x 1 root      root      7381 Jul 10 2010 confstr
-rwxr-xr-x 1 root      root      41136 Jul 10 2010 cp
lrwxrwxrwx 1 root      root      3 Oct 15 01:13 cpio -> pax
-rwxr-xr-x 1 root      root      42534 Jul 10 2010 csplit
-rwxr-xr-x 1 root      root      17261 Jul 10 2010 dd
-rwxr-xr-x 1 root      root      16041 Jul 10 2010 df
-rwxr-xr-x 1 root      root      24194 Jul 10 2010 dispconf
-rwxr-xr-x 1 root      root      11658 Jul 10 2010 du
-rwxr-xr-x 1 root      root      32745 Jul 10 2010 dumpifs
-rwxr-xr-x 1 root      root      7250 Jul 10 2010 echo
-rwxr-xr-x 1 root      root      83449 Jul 10 2010 ed
-rwxr-xr-x 1 root      root      481278 Jul 10 2010 elvis
-rwxr-xr-x 1 root      root      19270 Jul 10 2010 esh
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 ex -> elvis
-rwxr-xr-x 1 root      root      5432 Jul 10 2010 false
-rwxr-xr-x 1 root      root      22968 Jul 10 2010 fesh
-rwxr-xr-x 1 root      root      112 Jul 10 2010 gunzip
-rwxr-xr-x 1 root      root      72831 Jul 10 2010 gzip
-rwxr-xr-x 1 root      root      6746 Jul 10 2010 hostname
-rwxr-xr-x 1 root      root      3089 Jul 10 2010 igawk
-rwxr-xr-x 1 root      root      9481 Jul 10 2010 kill
-rwxr-xr-x 1 root      root      203834 Jul 10 2010 ksh
-rwxr-xr-x 1 root      root      13055 Jul 10 2010 ln
-rwsrwxr-x 1 root      root      53654 Jul 10 2010 login
-rwxr-xr-x 1 root      root      5581 Jul 10 2010 logout
-rwxr-xr-x 1 root      root      20745 Jul 10 2010 ls
-rwxr-xr-x 1 root      root      10724 Jul 10 2010 mkdir
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mkefs -> mkxfs
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mketfs -> mkxfs
lrwxrwxrwx 1 root      root      5 Oct 15 01:13 mkifs -> mkxfs
-rwxr-xr-x 1 root      root      20651 Jul 10 2010 mkifsf_elf
-rwxr-xr-x 1 root      root      14428 Jul 10 2010 mkifsf_openbios
-rwxr-xr-x 1 root      root      18735 Jul 10 2010 mkifsf_src
More (13%)

```

```
ttyp0: sh
-rwxr-xr-x 1 root root 14434 Jul 10 2010 tail
-rwxr-xr-x 1 root root 388796 Jul 10 2010 tar
-rwxr-xr-x 1 root root 7878 Jul 10 2010 tee
-rwxr-xr-x 1 root root 81103 Jul 10 2010 telnet
-rwxr-xr-x 1 root root 19196 Jul 10 2010 termdef
-rwxr-xr-x 1 root root 10189 Jul 10 2010 textto
-rwxr-xr-x 1 root root 31977 Jul 10 2010 tftp
-rwxr-xr-x 1 root root 107701 Jul 10 2010 tic
-rwxr-xr-x 1 root root 8175 Jul 10 2010 time
-rwxr-xr-x 1 root root 23593 Jul 10 2010 top
-rwxr-xr-x 1 root root 9789 Jul 10 2010 touch
-rwxr-xr-x 1 root root 19628 Jul 10 2010 tr
-rwxr-xr-x 1 root root 126945 Jul 10 2010 traceprinter
-rwsrwxr-x 1 root root 106668 Jul 10 2010 traceroute
-rwsrwxr-x 1 root root 89080 Jul 10 2010 traceroute6
-rwxr-xr-x 1 root root 11206 Jul 10 2010 tsort
-rwxr-xr-x 1 root root 6542 Jul 10 2010 tty
-rwxr-xr-x 1 root root 8825 Jul 10 2010 umask
-rwxr-xr-x 1 root root 8031 Jul 10 2010 unexpand
-rwxr-xr-x 1 root root 11976 Jul 10 2010 unifdef
-rwxr-xr-x 1 root root 9123 Jul 10 2010 uniq
-rwxr-xr-x 1 root root 6146 Jul 10 2010 unlink
-rwxr-xr-x 1 root root 133099 Jul 10 2010 unzip
-rwxr-xr-x 1 root root 57350 Jul 10 2010 unzipsfx
-rwxr-xr-x 1 root root 7020 Jul 10 2010 uptime
-rwxr-xr-x 1 root root 21441 Jul 10 2010 use
-rwxr-xr-x 1 root root 90865 Jul 10 2010 uud
lrwxrwxrwx 1 root root 3 Oct 15 01:13 uudecode -> uud
-rwxr-xr-x 1 root root 83195 Jul 10 2010 uue
lrwxrwxrwx 1 root root 3 Oct 15 01:13 uuencode -> uue
-rwxr-xr-x 1 root root 11262 Jul 10 2010 vsync
-rwxr-xr-x 1 root root 18345 Jul 10 2010 wave
-rwxr-xr-x 1 root root 16157 Jul 10 2010 waverec
-rwxr-xr-x 1 root root 9112 Jul 10 2010 wc
-rwxr-xr-x 1 root root 10950 Jul 10 2010 which
-rwxr-xr-x 1 root root 12104 Jul 10 2010 xargs
-rwxr-xr-x 1 root root 19135 Jul 10 2010 zap
-rwxr-xr-x 1 root root 170753 Jul 10 2010 zip
-rwxr-xr-x 1 root root 69213 Jul 10 2010 zipcloak
lrwxrwxrwx 1 root root 5 Oct 15 01:13 zipinfo -> unzip
-rwxr-xr-x 1 root root 64771 Jul 10 2010 zipnote
-rwxr-xr-x 1 root root 68358 Jul 10 2010 zipsplit
#
#
```

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

```
ttyp0: sh
# ls -l /usr/bin/g*
-rwxr-xr-x 1 root root 373187 Jul 10 2010 /usr/bin/gawk
-rwxr-xr-x 1 root root 1041 Jul 10 2010 /usr/bin/get_hw_info
-rwxr-xr-x 1 root root 21432 Jul 10 2010 /usr/bin/getconf
-rwxr-xr-x 1 root root 51135 Jul 10 2010 /usr/bin/gf-calib
-rwxr-xr-x 1 root root 8424 Jul 10 2010 /usr/bin/gf_cursor
-rwxr-xr-x 1 root root 9270 Jul 10 2010 /usr/bin/gfi-demo
-rwxr-xr-x 1 root root 45563 Jul 10 2010 /usr/bin/grep
#
#
# ls -l /usr/bin/t??
-rwxr-xr-x 1 root root 388796 Jul 10 2010 /usr/bin/tar
-rwxr-xr-x 1 root root 7878 Jul 10 2010 /usr/bin/tee
-rwxr-xr-x 1 root root 107701 Jul 10 2010 /usr/bin/tic
-rwxr-xr-x 1 root root 23593 Jul 10 2010 /usr/bin/top
-rwxr-xr-x 1 root root 6542 Jul 10 2010 /usr/bin/tty
#
```


Команда «rm /usr/bin/g*» удалила 6 7 файлов

Команда «rm /usr/bin/t??» удалила 6 5 файлов

Сколько всего пользователей зарегистрировано в системе?

```
ttyp0: sh
# 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:/:
forward:x:100:100:Egor:/home/forward:/bin/sh
#
#
#
#
#
```

Сколько различных групп пользователей в системе?

```
# 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:
#
#
```

Определить имена пользователей, у которых нет пароля.

```
ttyp0: sh
# cat /etc/passwd | grep -E "(.:::.)"
root::0:0:Superuser:/root:/bin/sh
#
#
#
#
```

Защитить файл для чтения со стороны владельца, проверить.

```
ttty0: sh
# ls -l
total 44
drwxrwxr-x 2 root root 1024 Nov 22 14:26 .
drwxrwxr-x 3 root root 1024 Nov 21 16:06 ..
-rw-rw-r- 1 root root 20308 Nov 22 14:34 bin
-rw-rw-r- 1 root root 0 Nov 21 15:46 file
-rw-rw-r- 1 root root 0 Nov 21 15:46 file2
-rw-rw-r- 1 root root 0 Nov 21 15:46 file4
#
# chmod u-r file4
#
# ls -l
total 44
drwxrwxr-x 2 root root 1024 Nov 22 14:26 .
drwxrwxr-x 3 root root 1024 Nov 21 16:06 ..
-rw-rw-r- 1 root root 20308 Nov 22 14:34 bin
-rw-rw-r- 1 root root 0 Nov 21 15:46 file
-rw-rw-r- 1 root root 0 Nov 21 15:46 file2
-rw-rw-r- 1 root root 0 Nov 21 15:46 file4
#
```

Защитили файл «file4» для чтения со стороны владельца командой

с
h
m
o

Защитить файл для чтения со стороны других пользователей, проверить.

u

```
ttty0: sh
# ls -l
total 44
drwxrwxr-x 2 root root 1024 Nov 22 14:26 .
drwxrwxr-x 3 root root 1024 Nov 21 16:06 ..
-rw-rw-r- 1 root root 20308 Nov 22 14:34 bin
-rw-rw-r- 1 root root 0 Nov 21 15:46 file
-rw-rw-r- 1 root root 0 Nov 21 15:46 file2
-rw-rw-r- 1 root root 0 Nov 21 15:46 file4
# chmod o-r file2
# ls -l
total 44
drwxrwxr-x 2 root root 1024 Nov 22 14:26 .
drwxrwxr-x 3 root root 1024 Nov 21 16:06 ..
-rw-rw-r- 1 root root 20308 Nov 22 14:34 bin
-rw-rw-r- 1 root root 0 Nov 21 15:46 file
-rw-rw-r- 1 root root 0 Nov 21 15:46 file2
-rw-rw-r- 1 root root 0 Nov 21 15:46 file4
#
```

Защитить файл для записи со стороны владельца, проверить.

```
ttyp0: sh
# ls -l
total 4
drwxrwxr-x  2 root    root    1024 Nov 22 16:45 .
drwxrwxr-x  4 root    root    1024 Nov 22 16:44 ..
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file1
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file2
# chmod u-w file1
# ls -l
total 4
drwxrwxr-x  2 root    root    1024 Nov 22 16:45 .
drwxrwxr-x  4 root    root    1024 Nov 22 16:44 ..
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file
-r--rw-r--  1 root    root      0 Nov 22 16:45 file1
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file2
# _
```

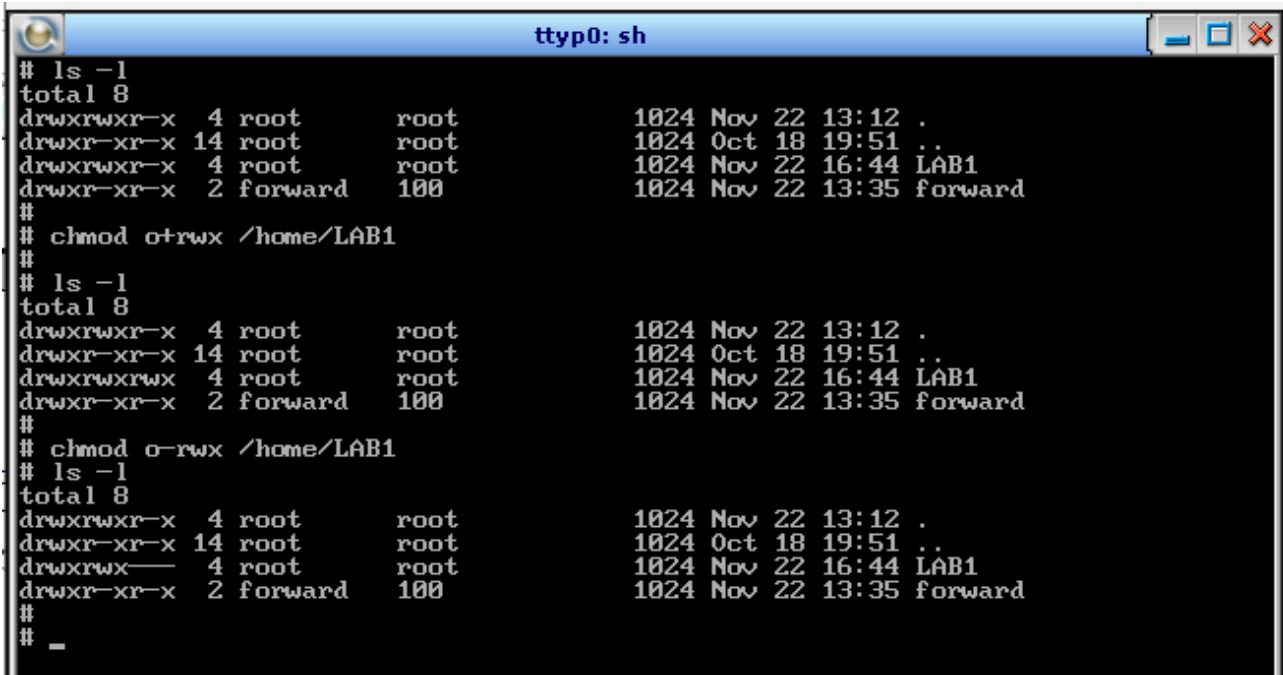
Защитили файл «file1» для записи со стороны владельца

Защитить файл для записи со стороны других пользователей, проверить.

```
ttyp0: sh
# ls -l
total 4
drwxrwxr-x  2 root    root    1024 Nov 22 16:45 .
drwxrwxr-x  4 root    root    1024 Nov 22 16:44 ..
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file
-r--rw-rw-  1 root    root      0 Nov 22 16:45 file1
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file2
# chmod o-w file1
# ls -l
total 4
drwxrwxr-x  2 root    root    1024 Nov 22 16:45 .
drwxrwxr-x  4 root    root    1024 Nov 22 16:44 ..
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file
-r--rw-r--  1 root    root      0 Nov 22 16:45 file1
-rw-rw-r--  1 root    root      0 Nov 22 16:45 file2
#
#
# _
```

Защитили файл “file1” для записи со стороны других пользователей

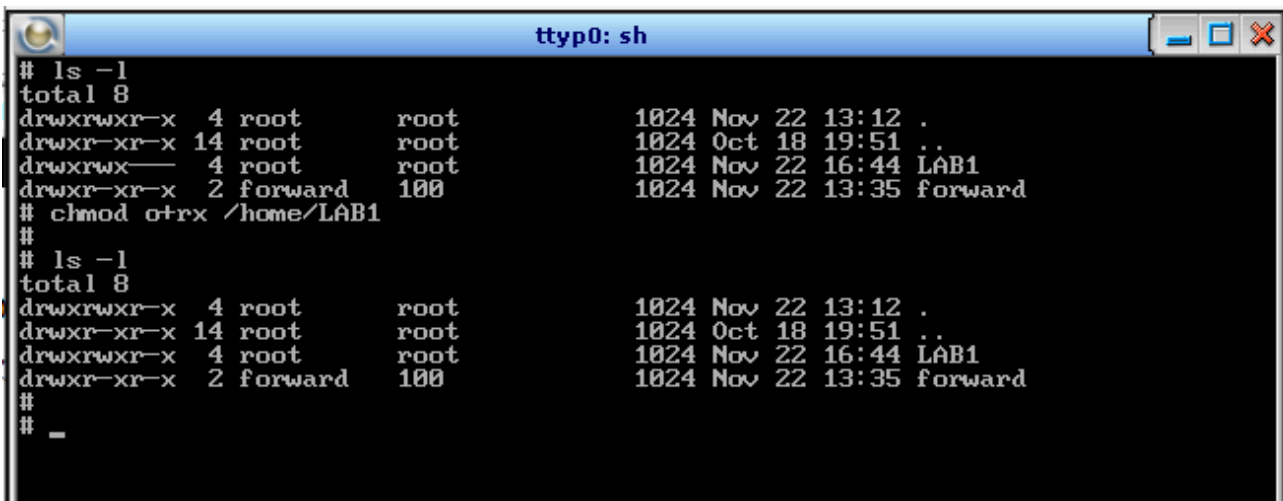
21) Открыть / закрыть свой основной каталог для доступа со стороны других пользователей, проверить.



```
ttyp0: sh
# ls -l
total 8
drwxrwxr-x 4 root root 1024 Nov 22 13:12 .
drwxr-xr-x 14 root root 1024 Oct 18 19:51 ..
drwxrwxr-x 4 root root 1024 Nov 22 16:44 LAB1
drwxr-xr-x 2 forward 100 1024 Nov 22 13:35 forward
#
# chmod o+rwx /home/LAB1
#
# ls -l
total 8
drwxrwxr-x 4 root root 1024 Nov 22 13:12 .
drwxr-xr-x 14 root root 1024 Oct 18 19:51 ..
drwxrwxrwx 4 root root 1024 Nov 22 16:44 LAB1
drwxr-xr-x 2 forward 100 1024 Nov 22 13:35 forward
#
# chmod o-rwx /home/LAB1
#
# ls -l
total 8
drwxrwxr-x 4 root root 1024 Nov 22 13:12 .
drwxr-xr-x 14 root root 1024 Oct 18 19:51 ..
drwxrwx--- 4 root root 1024 Nov 22 16:44 LAB1
drwxr-xr-x 2 forward 100 1024 Nov 22 13:35 forward
#
# _
```

Открыли и закрыли каталог LAB1 для доступа со стороны других пользователей

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



```
ttyp0: sh
# ls -l
total 8
drwxrwxr-x 4 root root 1024 Nov 22 13:12 .
drwxr-xr-x 14 root root 1024 Oct 18 19:51 ..
drwxrwx--- 4 root root 1024 Nov 22 16:44 LAB1
drwxr-xr-x 2 forward 100 1024 Nov 22 13:35 forward
# chmod o+rx /home/LAB1
#
# ls -l
total 8
drwxrwxr-x 4 root root 1024 Nov 22 13:12 .
drwxr-xr-x 14 root root 1024 Oct 18 19:51 ..
drwxrwxr-x 4 root root 1024 Nov 22 16:44 LAB1
drwxr-xr-x 2 forward 100 1024 Nov 22 13:35 forward
#
# _
```

Разрешить доступ к файлам только с известными именами, проверить.

```
ttty0: sh
# ls -l
total 8
drwxrwxr-x  4 root    root    1024 Nov 22 13:12 .
drwxr-xr-x 14 root    root    1024 Oct 18 19:51 ..
drwxrwxr-x  4 root    root    1024 Nov 22 16:44 LAB1
drwxr-xr-x  2 forward 100    1024 Nov 22 13:35 forward
#
# chmod ugo+rwx /home/LAB1
# ls -l
total 8
drwxrwxr-x  4 root    root    1024 Nov 22 13:12 .
drwxr-xr-x 14 root    root    1024 Oct 18 19:51 ..
drwxrwxrwx  4 root    root    1024 Nov 22 16:44 LAB1
drwxr-xr-x  2 forward 100    1024 Nov 22 13:35 forward
# -
```

P

S

Создаем файл file5.txt:

Для проверки доступа к файлу

#vi file5.txt

Защищаем файл от других пользователей:

c

h

Заходим на другого пользователя, который существует в системе, в нашем случае

m

пользователь «forward»:

o

su - forward

d

И проверяем доступ к данному файлу:

rwX

c

a

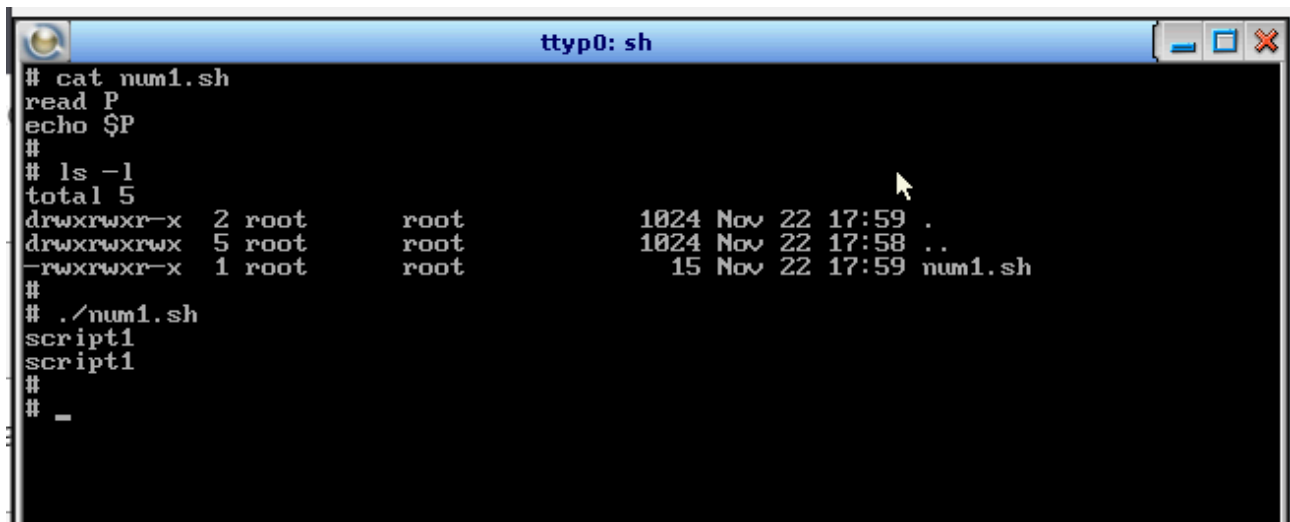
В результате чего увидим:

t

```
$ ls -l
total 41
-rw-rw-r--  1 root    root    20308 Nov 22 14:34 bin
-rw-rw-r--  1 root    root         0 Nov 21 15:46 file
-rw-rw-r--  1 root    root         0 Nov 21 15:46 file2
-rw-rw-r--  1 root    root         0 Nov 21 15:46 file4
-rw-rw-r--  1 root    root         1 Nov 22 17:23 file5.txt
$ cat file5.txt
file5.txt: Permission denied
$
$ su root
#
```

2. Создание простых скриптов

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



```
# cat num1.sh
read P
echo $P
#
# ls -l
total 5
drwxrwxr-x 2 root root 1024 Nov 22 17:59 .
drwxrwxrwx 5 root root 1024 Nov 22 17:58 ..
-rwxrwxr-x 1 root root 15 Nov 22 17:59 num1.sh
#
# ./num1.sh
script1
script1
#
# _
```

Создали файл num1.sh:

```
#vi num1.sh
```

И прописали в нем команды:

```
r
```

```
e
```

```
s
```

```
a
```

```
s
```

```
p
```

```
r
```

```
s
```

```
h
```

```
m
```

```
o
```

```
d
```

```
a
```

```
x
```

```
n
```

```
u
```

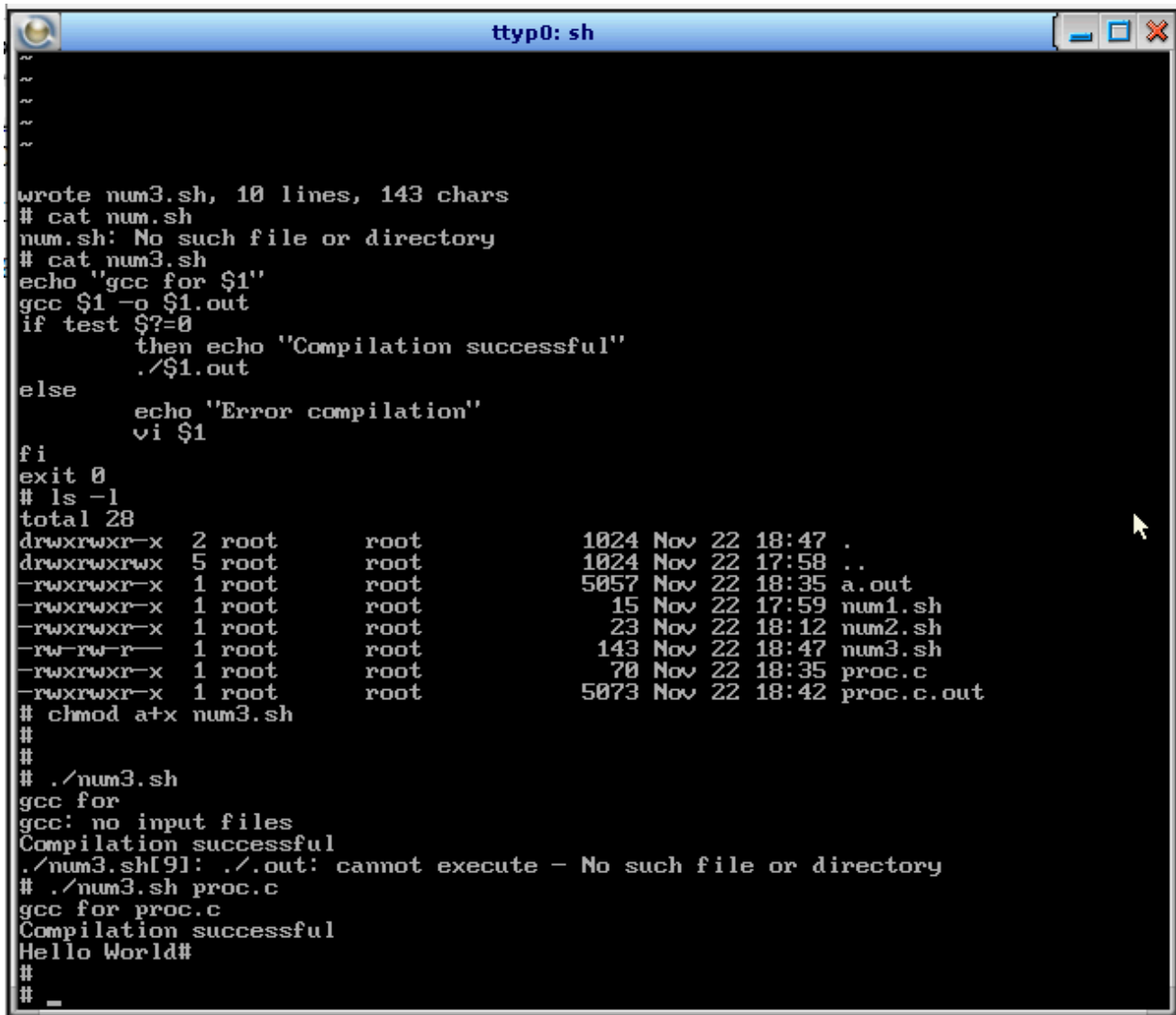
```
m
```

```
sh
```

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


```
ttty0: sh
118801 3 shbin/io-display 12o RECEIVE 1
118801 4 shbin/io-display 10o RECEIVE 3
118801 5 shbin/io-display 10o RECEIVE 1
135186 1 shbin/io-pkt-v4-hc 21o SIGWAITINFO
135186 2 shbin/io-pkt-v4-hc 21o RECEIVE 1
135186 3 shbin/io-pkt-v4-hc 21r RECEIVE 16
159759 1 shbin/devc-pty 10o RECEIVE 1
172052 1 usr/shbin/dumper 10o RECEIVE 1
176151 1 r/shbin/dhcp.client 10o NANOSLEEP
188437 1 bin/login 10o REPLY 4103
188438 1 bin/login 10o REPLY 4103
188440 1 bin/login 10o REPLY 4103
217115 1 /photon/bin/Photon 10r RECEIVE 1
237596 1 on/bin/io-graphics 12r CONDUAR (0x805dc60)
237596 2 on/bin/io-graphics 10r RECEIVE 1
237596 3 on/bin/io-graphics 12r REPLY 217115
253983 1 hoton/bin/devi-hid 10o RECEIVE 1
253983 2 hoton/bin/devi-hid 10o REPLY 4102
253983 3 hoton/bin/devi-hid 12o SIGWAITINFO
253983 5 hoton/bin/devi-hid 10o RECEIVE 1
266265 1 bin/login 10o REPLY 4103
1085469 1 usr/photon/bin/pwm 10r RECEIVE 1
1114142 1 r/photon/bin/shelf 10r CONDUAR (0x8076f88)
1114142 2 r/photon/bin/shelf 10r RECEIVE
1134624 1 photon/bin/bkgdmgr 10r RECEIVE 1
1134625 1 hoton/bin/wmswitch 10r RECEIVE 2
1134626 1 r/photon/bin/saver 10r RECEIVE 1
1171475 1 r/photon/bin/pterm 10r RECEIVE 1
1171482 1 bin/sh 10r SIGSUSPEND
2121763 1 usr/bin/grep 10r REPLY 159759
2150436 1 usr/bin/less 10r STOPPED
2711589 1 bin/sh 10r SIGSUSPEND
2801702 1 bin/sh 10r SIGSUSPEND
3104807 1 bin/sh 10r SIGSUSPEND
3252264 1 bin/sh 10r SIGSUSPEND
3694633 1 bin/pidin 10r REPLY 1
#
#
# ls
# ./num2.sh .. num1.sh num2.sh
#
266265
266265 1 bin/login 10o REPLY 4103
#
#
```


3. Написать скрипт, который компилирует указанную программу и при отсутствии ошибок запускает её. Если же есть ошибки, то автоматически вызывает редактор для их исправления.



```
~
~
~
~
wrote num3.sh, 10 lines, 143 chars
# cat num.sh
num.sh: No such file or directory
# cat num3.sh
echo "gcc for $1"
gcc $1 -o $1.out
if test $?=0
    then echo "Compilation successful"
    ./ $1.out
else
    echo "Error compilation"
    vi $1
fi
exit 0
# ls -l
total 28
drwxrwxr-x  2 root    root    1024 Nov 22 18:47 .
drwxrwxr-x  5 root    root    1024 Nov 22 17:58 ..
-rwxrwxr-x  1 root    root     5057 Nov 22 18:35 a.out
-rwxrwxr-x  1 root    root       15 Nov 22 17:59 num1.sh
-rwxrwxr-x  1 root    root       23 Nov 22 18:12 num2.sh
-rw-rw-r-  1 root    root      143 Nov 22 18:47 num3.sh
-rwxrwxr-x  1 root    root       70 Nov 22 18:35 proc.c
-rwxrwxr-x  1 root    root    5073 Nov 22 18:42 proc.c.out
# chmod a+x num3.sh
#
#
# ./num3.sh
gcc for
gcc: no input files
Compilation successful
./num3.sh[9]: ./out: cannot execute - No such file or directory
# ./num3.sh proc.c
gcc for proc.c
Compilation successful
Hello World#
#
#
# _
```

3. Разработка программ

Написать программу, выводящую сообщение "HELLO" в центре чистого экрана.



Файл «prog1.c»:

```
#
# cat prog1.c
#include <stdio.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <stdlib.h>

void getScreenSize(int* width, int *height)
{
    struct winsize wwinsize;
    ioctl(STDOUT_FILENO, TIOCGWINSZ, &wwinsize);
    *width=wwinsize.ws_col -1;
    *height=wwinsize.ws_row -1;
}

int main(int argc, char* argv[])
{
    int width;
    int height;
    getScreenSize(&width, &height);
    printf("%d:%d\n", width, height);
    int i=0;
    for(i=0; i<height; i++)
    {
        if(i==height/2)
            printf("%s\n", width/2, "Hello");
        else
            printf("\n");
    }
    return 0;
}
# _
```

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



```
# ./prog2.out
key: 49
key: 50
key: 51
key: 52
key: 113
key: 119
key: 101
key: 114
key: 32
key: 10
# _
```

Файл Prog2.c

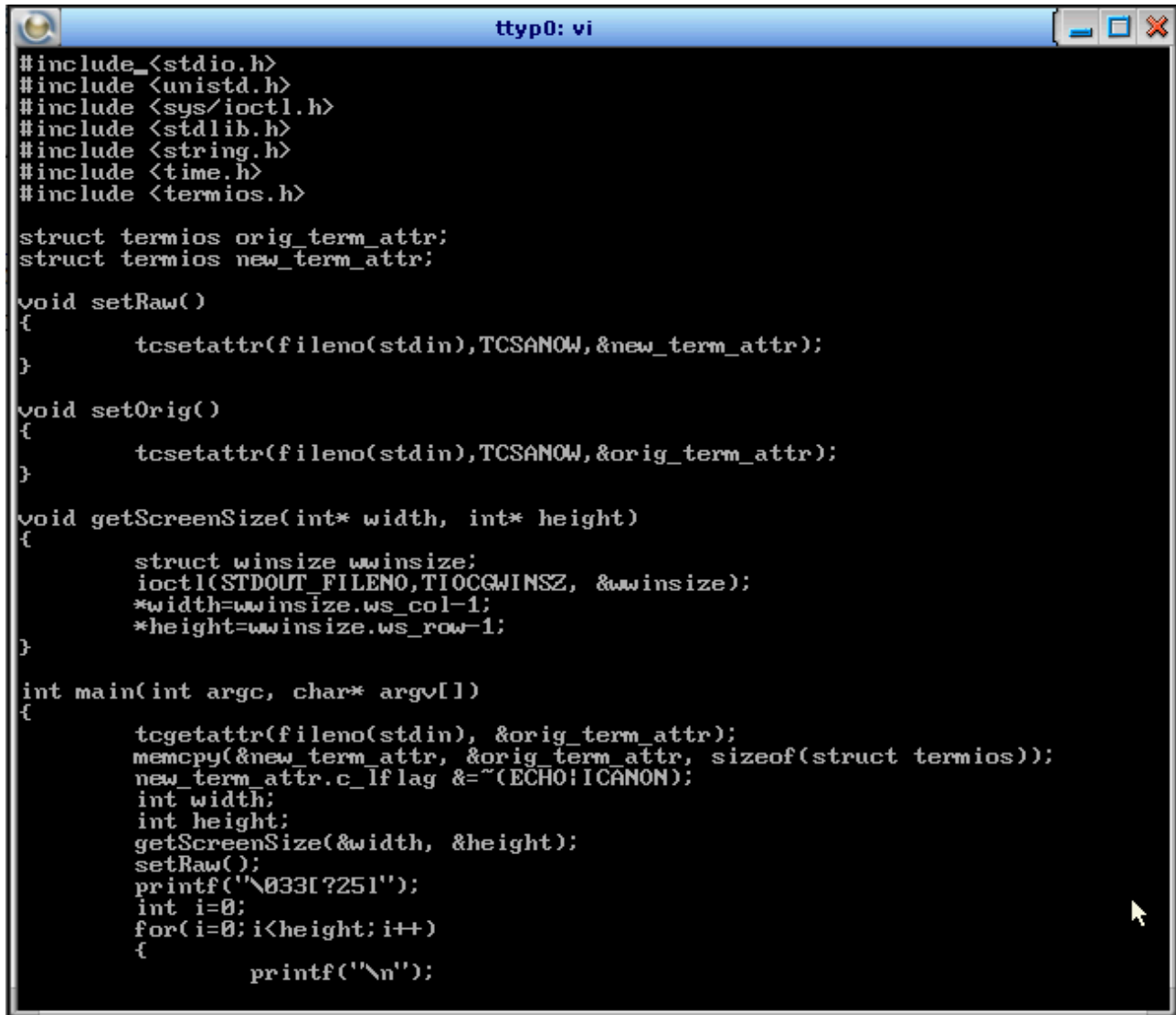
```
ttyp0: vi
#include <stdio.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <termios.h>
#include <string.h>
#include <time.h>

int getKey()
{
    int character;
    struct termios orig_term_attr;
    struct termios new_term_attr;
    tcgetattr(fileno(stdin), &orig_term_attr);
    memcpy(&new_term_attr, &orig_term_attr, sizeof(struct termios));
    new_term_attr.c_lflag &= ~(ECHO | ICANON);
    tcsetattr(fileno(stdin), TCSANOW, &new_term_attr);
    character = fgetc(stdin);
    tcsetattr(fileno(stdin), TCSANOW, &orig_term_attr);
    return character;
}

int main(int argc, char* argv[])
{
    int key;
    while(1)
    {
        key = getKey();
        if (key == 0x1B || key == 0x04)
            break;
        else
            printf("key: %d\n", key);
    }
    return 0;
}
~
~
~
~
~
~
~
read prog2.c, 35 lines, 695 chars
```

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

Файл prog2.c



```
#include <stdio.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <termios.h>

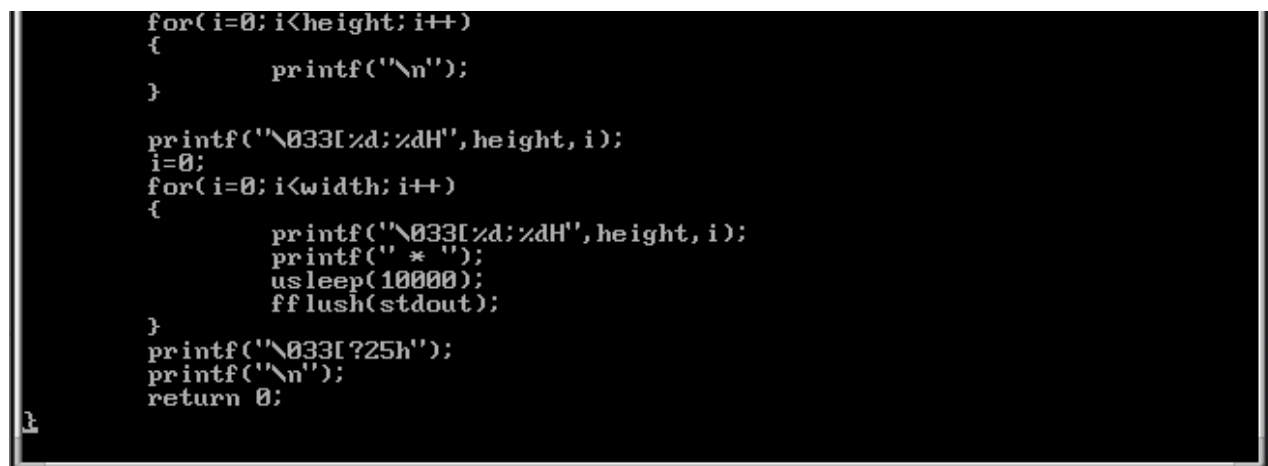
struct termios orig_term_attr;
struct termios new_term_attr;

void setRaw()
{
    tcsetattr(fileno(stdin), TCSANOW, &new_term_attr);
}

void setOrig()
{
    tcsetattr(fileno(stdin), TCSANOW, &orig_term_attr);
}

void getScreenSize(int* width, int* height)
{
    struct winsize wwinsize;
    ioctl(STDOUT_FILENO, TIOCGWINSZ, &wwinsize);
    *width = wwinsize.ws_col - 1;
    *height = wwinsize.ws_row - 1;
}

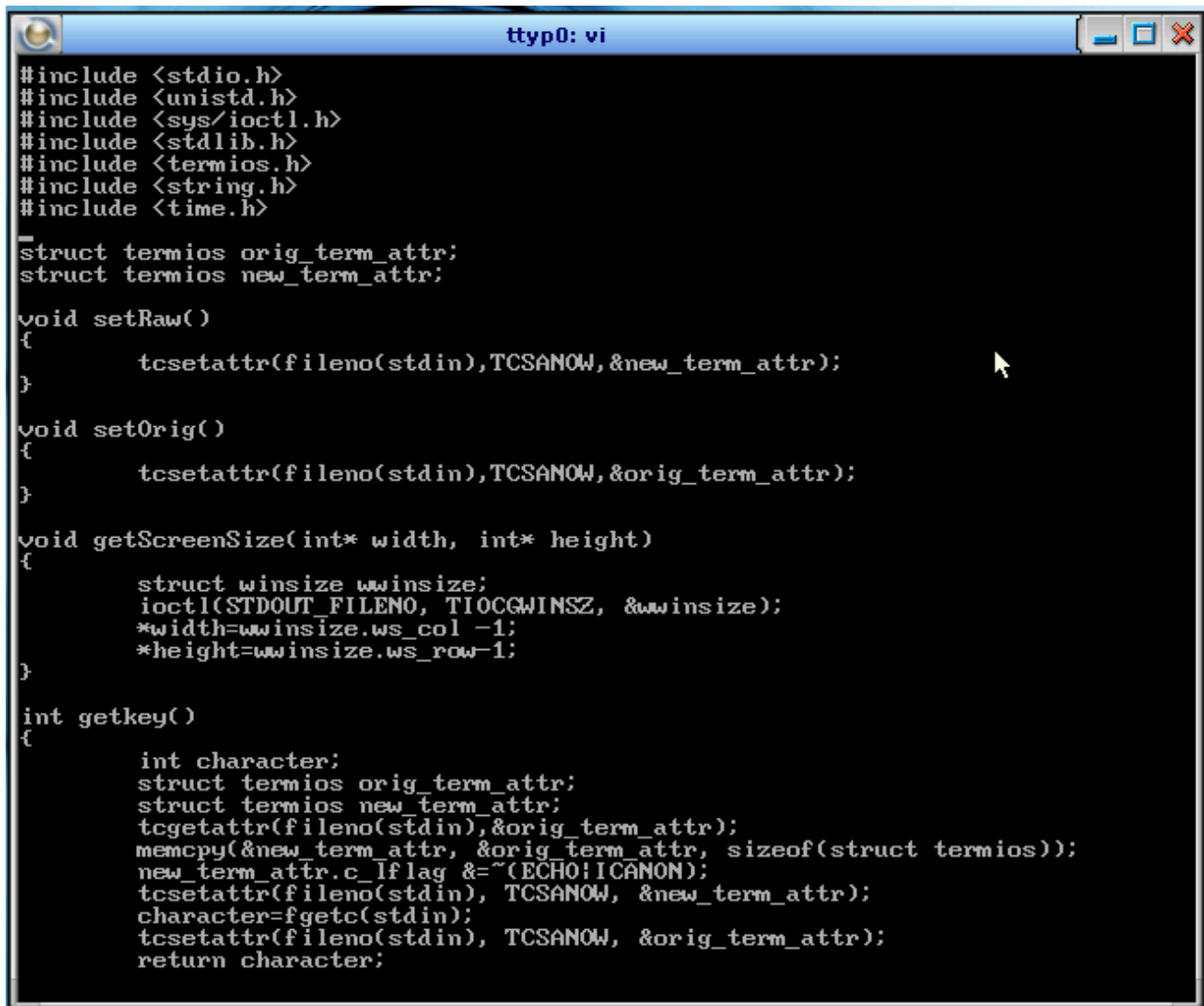
int main(int argc, char* argv[])
{
    tcgetattr(fileno(stdin), &orig_term_attr);
    memcpy(&new_term_attr, &orig_term_attr, sizeof(struct termios));
    new_term_attr.c_lflag &= ~(ECHO | ICANON);
    int width;
    int height;
    getScreenSize(&width, &height);
    setRaw();
    printf("\033[?25l");
    int i = 0;
    for(i = 0; i < height; i++)
    {
        printf("\n");
    }
```



```
    for(i = 0; i < height; i++)
    {
        printf("\n");
    }

    printf("\033[?d:zdH", height, i);
    i = 0;
    for(i = 0; i < width; i++)
    {
        printf("\033[?d:zdH", height, i);
        printf(" * ");
        usleep(10000);
        fflush(stdout);
    }
    printf("\033[?25h");
    printf("\n");
    return 0;
}
```

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).



```
#include <stdio.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <stdlib.h>
#include <termios.h>
#include <string.h>
#include <time.h>

struct termios orig_term_attr;
struct termios new_term_attr;

void setRaw()
{
    tcsetattr(fileno(stdin), TCSANOW, &new_term_attr);
}

void setOrig()
{
    tcsetattr(fileno(stdin), TCSANOW, &orig_term_attr);
}

void getScreenSize(int* width, int* height)
{
    struct winsize wwinsize;
    ioctl(STDOUT_FILENO, TIOCGWINSZ, &wwinsize);
    *width=wwinsize.ws_col-1;
    *height=wwinsize.ws_row-1;
}

int getKey()
{
    int character;
    struct termios orig_term_attr;
    struct termios new_term_attr;
    tcgetattr(fileno(stdin), &orig_term_attr);
    memcpy(&new_term_attr, &orig_term_attr, sizeof(struct termios));
    new_term_attr.c_lflag &= ~(ECHO|ICANON);
    tcsetattr(fileno(stdin), TCSANOW, &new_term_attr);
    character=fgetc(stdin);
    tcsetattr(fileno(stdin), TCSANOW, &orig_term_attr);
    return character;
}
```

```
ttyp0: vi

    return character;
}

int main (int argc, char* argv[])
{
    char color=0;
    int speed=100000;
    int dirX=0;
    int dirY=0;
    int i=0;

    for(i=0; i<argc; i++)
    {
        if(strcmp(argv[i], "-speed")==0)
        {
            speed=strtol(argv[i+1], 0, 10);
        }

        if(strcmp(argv[i], "-color")==0)
        {
            color=argv[i+1][0];
        }

        if(strcmp(argv[i], "-dirX")==0)
        {
            dirX=strtol(argv[i+1], 0, 10);
        }

        if(strcmp(argv[i], "-dirY")==0)
        {
            dirY=strtol(argv[i+1], 0, 10);
        }
    }

    printf("color=%c\n", color);
    printf("speed=%d\n", speed);
    printf("dirX=%d\n", dirX);
    printf("dirY=%d\n", dirY);

    usleep(1000000);
    tcgetattr(fileno(stdin), &orig_term_attr);
}
```

```
ttyp0: vi
usleep(1000000);
tcgetattr(fileno(stdin), &orig_term_attr);
memcpy(&new_term_attr, &orig_term_attr, sizeof(struct termios));
new_term_attr.c_lflag &= ~(ECHO|ICANON);
int width;
int height;
getScreenSize(&width, &height);
setRaw();

printf("\033[?25l");
i=0;
for(i=0; i<height; i++)
{
    printf("\n");
}
printf("\033[%cF", color);
printf("\033[%d;%dH", height, i);

int posX=0;
int posY=0;

if(dirX==0)
    posX=1;
if(dirX==1)
    posX=width-1;
if(dirY==0)
    posY=1;
if(dirY==1)
    posY=height-1;

while(1)
{
    printf("\033[%d;%dH", posY-1, posX+1);
    printf(" ");
    printf("\033[%d;%dH", posY-1, posX);
    printf(" ");
    printf("\033[%d;%dH", posY-1, posX-1);
    printf(" ");
    printf("\033[%d;%dH", posY, posX);
    printf(" ");
    _printf("\033[%d;%dH", posY+1, posX-1);
}
```

```
ttyp0: vi

printf("\033[%d;%dH", posY+1, posX-1);
printf(" ");
printf("\033[%d;%dH", posY+1, posX);
printf(" ");
printf("\033[%d;%dH", posY+1, posX+1);
printf(" ");

usleep(speed);
fflush(stdout);

if(dirX==0)
    posX++;
if(dirX==1)
    posX--;
if(dirY==0)
    posY++;
if(dirY==1)
    posY--;

if(posX<=1)
    dirX=0;
if(posX>=width)
    dirX=1;
if(posY<=1)
    dirY=0;
if(posY>=height)
    dirY=1;

setOrig();
int key=getkey();
if(key==0x1B || key==0x04)
    break;
setRaw();
printf("\033[1;1H");
printf("POSX:%d POSY:%d, KEY:%d", posX, posY, key);
}

printf("\033[7F");
printf("\033[?25h");
printf("\n");
setOrig();
```

```
setOrig();
return 0;
}
```