Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа № 1

# по дисциплине «**Операционные системы реального времени**»

Выполнили:

студенты группы ИП-712

Коротеев Максим Сергеевич

Бахарев Максим Евгеньевич

                 ФИО студента

Работу проверил:

заведующий кафедрой Фионов А.Н.

ФИО преподавателя

Новосибирск 2020 г.

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

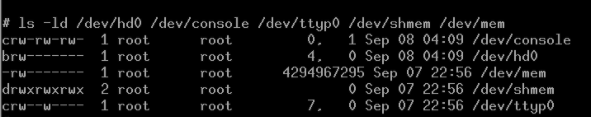
**1. Определить тип файлов /dev/hd0, /dev/console, /dev/ttyp0, /dev/shmem, /dev/mem.**

/dev/hd0 — блоковый специальный файл (устройство)

/dev/console — символьный специальный файл (устройство)

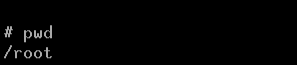
/dev/ttyp0 — символьный специальный файл (устройство)

/dev/shmem — каталог/dev/mem — регулярный файл



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

При входе в систему рабочим делается домашний каталог суперпользователя (root) так как всегда при входе в систему или при открытии терминала мы находимся в домашнем каталоге. Домашний каталог для каждого пользователя задан в файле /etc/passwd.



**3.Создать каталог LAB1 и сделать его рабочим.**

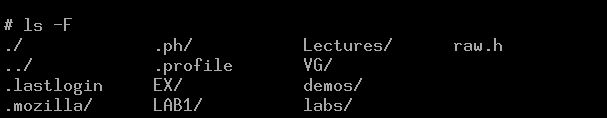


**4. Определить (с помощью программы ls), в каком каталоге содержится файл services. Посмотреть его содержимое.**



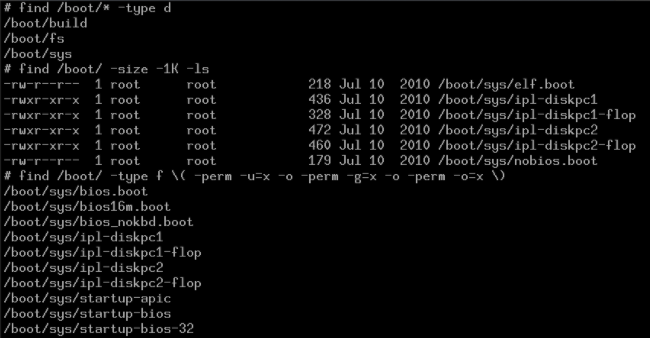
**5. Сколько скрытых файлов в вашем домашнем каталоге?**

2 обычных файла, 2 каталога, 2 специальных файла-ссылки на текущий и родительский каталог. Итого 6.



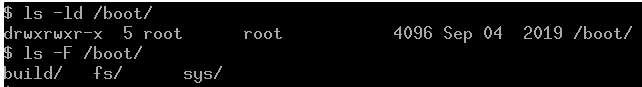
**6. Определить полное дерево подкаталогов в /boot. Сколько там файлов, размер которых меньше 1К байт? Сколько там исполняемых файлов**

Итого: 6 файлов с размером меньше 1К, 10 исполняемых файлов.



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

5 жёстких ссылок, потому что одна ссылка ведёт из /, 3 ведёт из каждого из подкаталогов на родительский каталог (..), и одна в самом каталоге на самого себя (.).



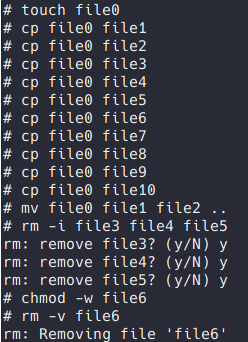
**8. Создать текстовый файл с помощью редактора vi. Какие флаги доступа устанавливаются у вновь создаваемого файла? Почему? Как это исправить?**

Появляются флаги чтения и записи для пользователя и для группы. Также для всех появляется флаг чтения. Потому что umask по-умолчанию имеет значение 002. Исправить права доступа можно при помощи программы chmod для созданного файла или при помощи umask для всех файлов, которые будут созданы.



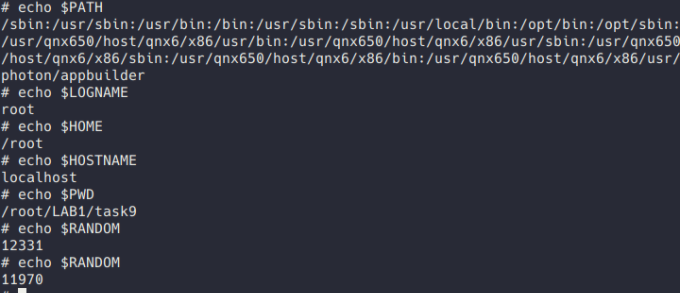
**9. Сделать каталог и создать в нем 10 копий некоторого файла. Перенести три из них в вышестоящий каталог. Удалить (с подтверждением) некоторые из оставшихся файлов. Проверить влияние флага w на команду удаления файла.**

Работать из виртуальной машины очень неудобно, поэтому я запустил на ней sshd, прокинул порты и подключился с хост-системы. На удаление файла флаг записи никак не влияет.



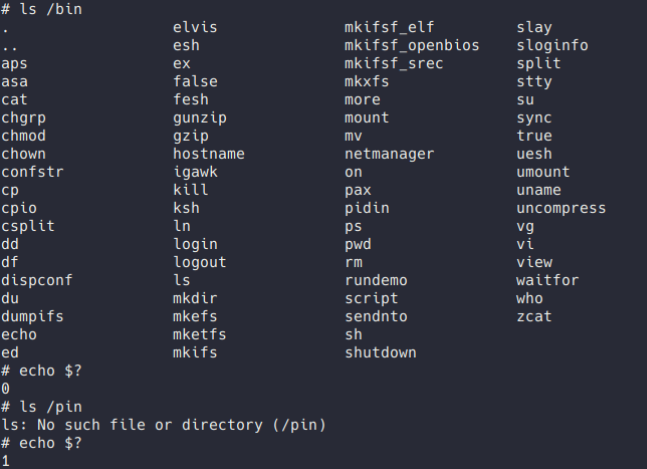
**10. Определить значения переменных среды PATH, LOGNAME, HOME, HOSTNAME, PWD, RANDOM. Меняются ли они со временем?**

Меняется только RANDOM, каждый раз выдавая новое случайное число.

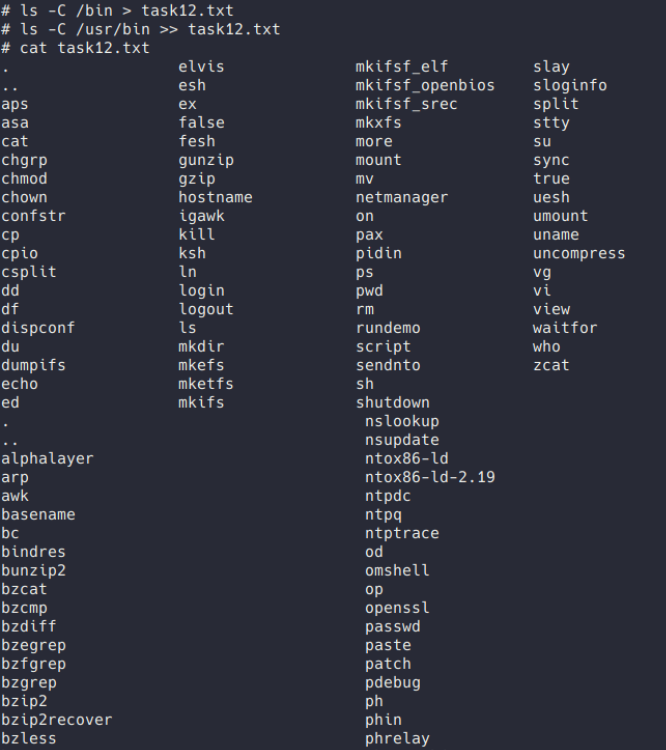


**11. Определить коды завершения команд ls /bin и ls /pin.**

В случае удачного завершения код возврата 0. Иначе другой. Чаще всего 1, но не обязательно.



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



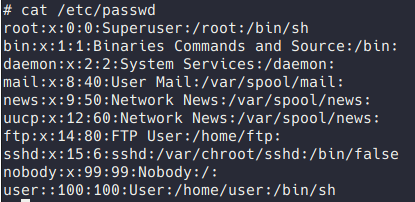
**13. Сколько файлов удалили бы команды rm /usr/bin/g\* и rm /usr/bin/t?? ? (просьба файлы не удалять)**

Первая 7, вторая 5.



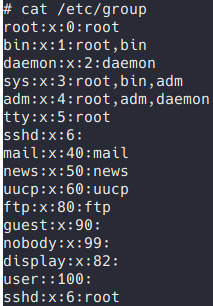
**14. Сколько всего пользователей зарегистрировано в системе?**

10 пользователей



**15. Сколько различных групп пользователей в системе?**

16 групп



**16. Определить имена пользователей, у которых нет пароля.**

Пароля нет ни у одного пользователя. Все из пункта 14



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

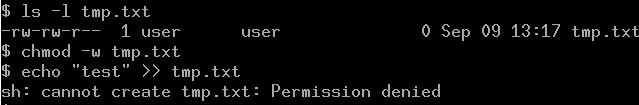
Суперпользователь может читать даже несмотря на то, что у файла нет флага чтения. Обычный пользователь не может читать свой файл без флага чтения.



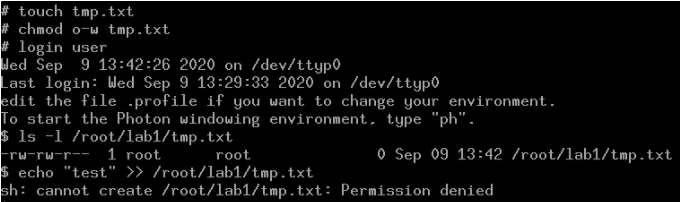
**18. Защитить файл для чтения со стороны других пользователей, проверить.**



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

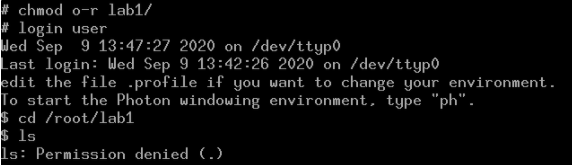


**20. Защитить файл для записи со стороны других пользователей, проверить.**

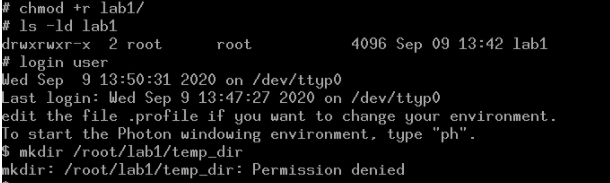


**21. Открыть / закрыть свой основной каталог для доступа со стороны других пользователей, проверить.**

Закрыл доступ для каталога lab1 для остальных.

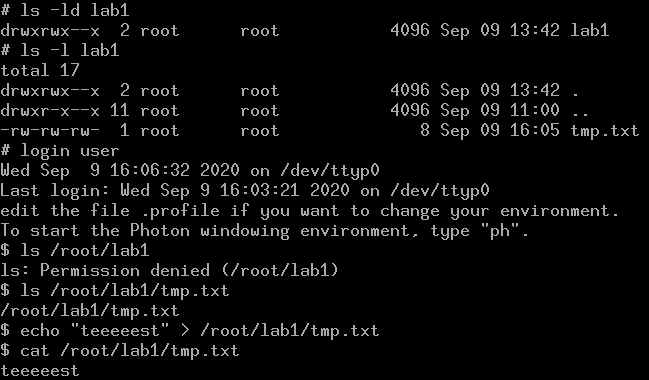


**22. Разрешить доступ к своему основному каталогу, но запретить его изменение, проверить.**



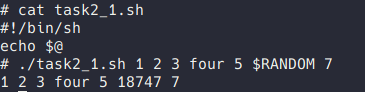
**23. Разрешить доступ к файлам только с известными именами, проверить.**

Убрал у каталога флаг чтения для всех. Теперь взаимодействовать с файлами можно только зная их название.

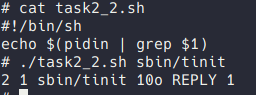


**2. Создание простых скриптов**

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

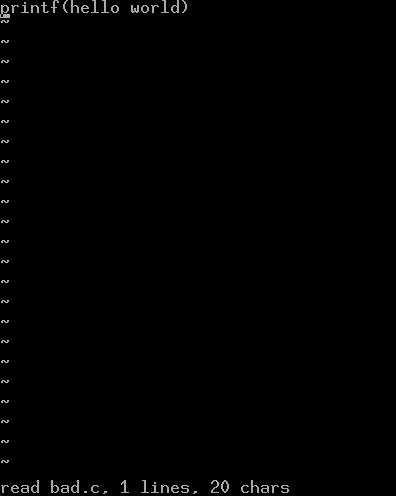


**2. Написать скрипт, который с помощью утилит pidin и grep выводит на экран информацию об указанном по имени процессе.**



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





**3. Разработка программ**

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

#include <stdio.h>

#include <sys/ioctl.h>

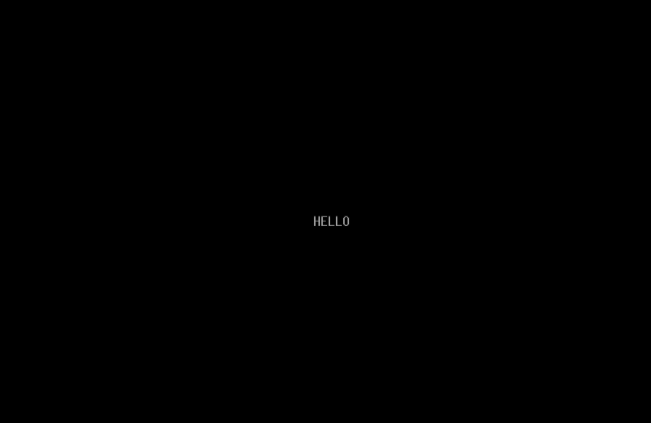
int main() {

struct winsize ws;

ioctl(1, TIOCGWINSZ, &ws);

printf("\033[2J\033[%d;%dHHELLO\n\033[99B", ws.ws\_row / 2, ws.ws\_col / 2);

}



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

#include <stdio.h>

#include "../raw.h" int main() {

setraw();

int c;

while ((c = getchar()) != 3) {

printf("%d", c);

}

unsetraw();

putchar('\n');

}

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

#include <unistd.h>

#include <stdio.h>

int main() {

setbuf(stdout, NULL);

printf("\033[?25l"); // Hide cursor

for (int i = 1; i <= 30; ++i) {

printf("\033[%dG\*", i);

usleep(100000);

printf("\033[D ");

}

printf("\033[?25h"); // Show cursor

putchar('\r');

}

**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).**

Перенаправление stdout на другие терминалы производится как ./a.out -s 4 -d right -c blue > /dev/ttyp0 & ./a.out -s 4 -d down -c red > /dev/ttyp1, где /dev/ttyp0 и /dev/ttyp1 — и есть другие терминалы.

#include <unistd.h>

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

#include <signal.h>

enum DIRECTION {

UP, DOWN, LEFT, RIGHT

};

enum COLOR {

BLACK, RED, GREEN, YELLOW, BLUE, MAGENTA, CYAN, WHITE

};

void shandler(int sig) {

printf("\033[?25h\033[0m\n");

exit(0);

}

int main(int argc, char \*argv[]) {

signal(SIGINT, shandler);

long speed = -1;

enum DIRECTION direction = -1;

enum COLOR color = -1;

int opt;

while ((opt = getopt(argc, argv, "s:d:c:")) != -1) {

char \*endptr;

switch (opt) {

case 's':

speed = strtol(optarg, &endptr, 0);

if (\*endptr != '\0' || speed < 1 || speed > 10) {

fprintf(stderr, "Param -s has value %s. It can be only integer in range [1; 10]\n", optarg);

exit(1);

}

break;

case 'd':

if (strcasecmp(optarg, "up") == 0) {

direction = UP;

} else if (strcasecmp(optarg, "down") == 0) {

direction = DOWN;

} else if (strcasecmp(optarg, "left") == 0) {

direction = LEFT;

} else if (strcasecmp(optarg, "right") == 0) {

direction = RIGHT;

} else {

fprintf(stderr, "Param -d has value '%s'. It can handle only the next values: "

"'up', 'down', 'left', 'right'\n", optarg);

exit(1);

}

break;

case 'c':

if (strcasecmp(optarg, "black") == 0) {

color = BLACK;

} else if (strcasecmp(optarg, "red") == 0) {

color = RED;

} else if (strcasecmp(optarg, "green") == 0) {

color = GREEN;

} else if (strcasecmp(optarg, "yellow") == 0) {

color = YELLOW;

} else if (strcasecmp(optarg, "blue") == 0) {

color = BLUE;

} else if (strcasecmp(optarg, "magenta") == 0) {

color = MAGENTA;

} else if (strcasecmp(optarg, "cyan") == 0) {

color = CYAN;

} else if (strcasecmp(optarg, "white") == 0) {

color = WHITE;

} else {

fprintf(stderr, "Param -c has value '%s'. It can handle only the next values: "

"'black', 'red', 'green', 'yellow', 'blue', 'magenta', 'cyan', 'white'\n", optarg);

exit(1);

}

break;

}

}

if (speed == -1 || direction == -1 || color == -1) {

fprintf(stderr, "There are one or more missing parameters. Usage: ./a.out -s 4 -d down -c red\n");

exit(1);

}

int dx = 0, dy = 0;

switch (direction) {

case DOWN:

dy = 1;

break;

case UP:

dy = -1;

break;

case LEFT:

dx = -1;

break;

case RIGHT:

dx = 1;

break;

}

setbuf(stdout, NULL);

printf("\033[2J");

useconds\_t sleep = 5000000 / speed;

printf("\033[?25l\033[3%dm", color);

for (int y = 12, x = 40;; y += dy, x += dx) {

printf("\033[%d;%dH\*", y, x);

usleep(sleep);

printf("\033[D ");

}

}