**Федеральное государственное автономное**

**Образовательное учреждение высшего образования**

**Российский Университет Дружбы Народов**

Математический университет имени Никольского

Факультет Физико-математических и Естественных наук

Кафедра Прикладной математики и информатики

Отчет по лабораторной работе № 5

“Анализ файловой системы Linux. Команды для работы с файлами и каталогами”

Выполнил:

Студент группы НПМбв-01-10

Адхамова Луиза Шухратовна

Москва

2024 год

# Цель работы:

Ознакомление с файловой системой Linux, её структурой, именами и содержанием каталогов. Приобретение практических навыков по применению команд для работы с файлами и каталогами, по управлению процессами (и работами), по проверке использования диска и обслуживанию файловой системы.

# Выполнение:

1. Выполним все примеры, приведённые в первой части описания лабораторной работы (рис. 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 1.10, 1.11, 1.12, 1.13, 1.14, 1.15).

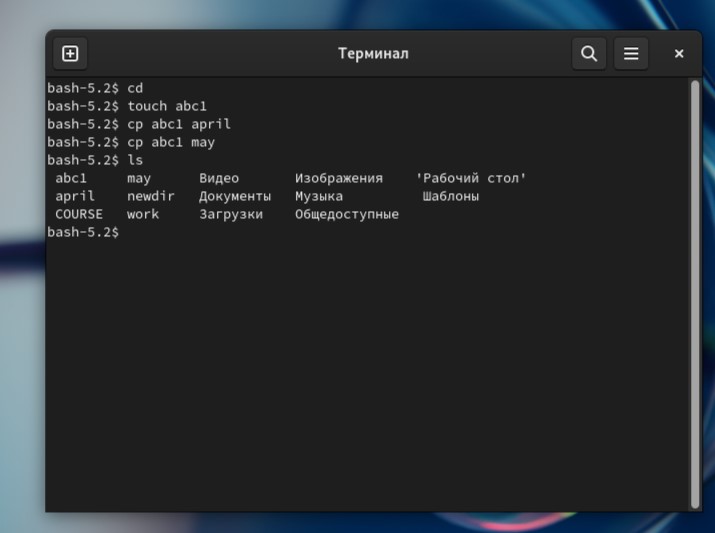


Рисунок 1.1. Копирование файла abc1 в папки may и april.

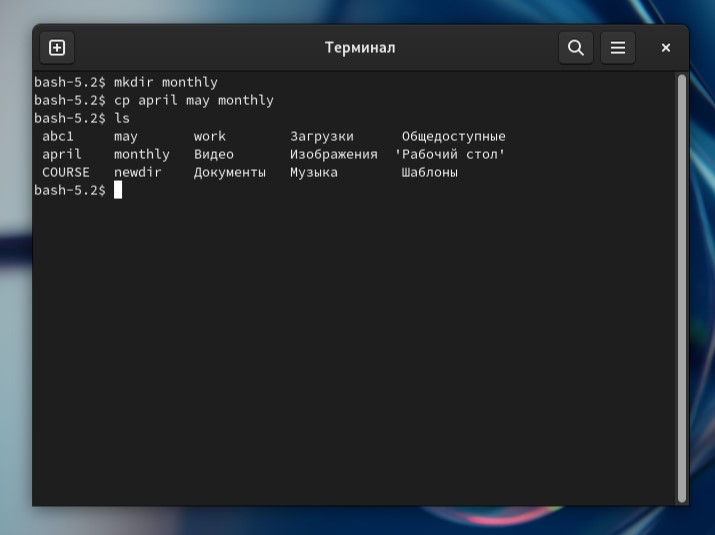


Рисунок 1.2. Создание папки monthly.

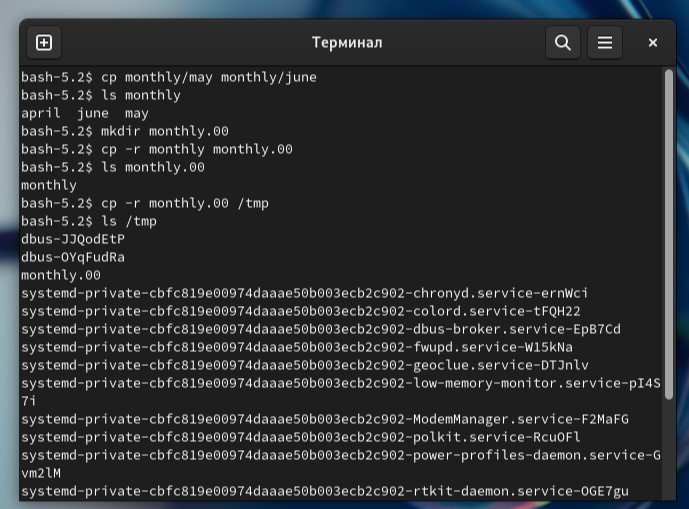


Рисунок 1.3. Копирование файла monthly/may в файл с именем june, копирование каталога monthly в каталог monthly.00, копирование каталога monthly.00 в каталог /tmp

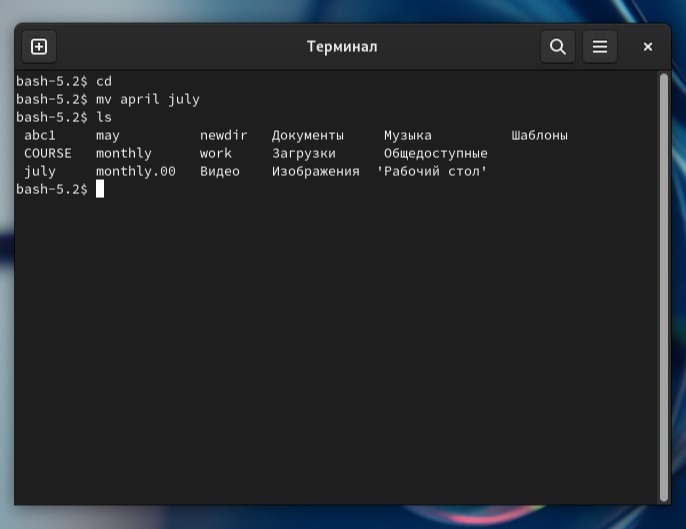


Рисунок 1.4. Изменение названия файла april на july в домашнем каталоге.

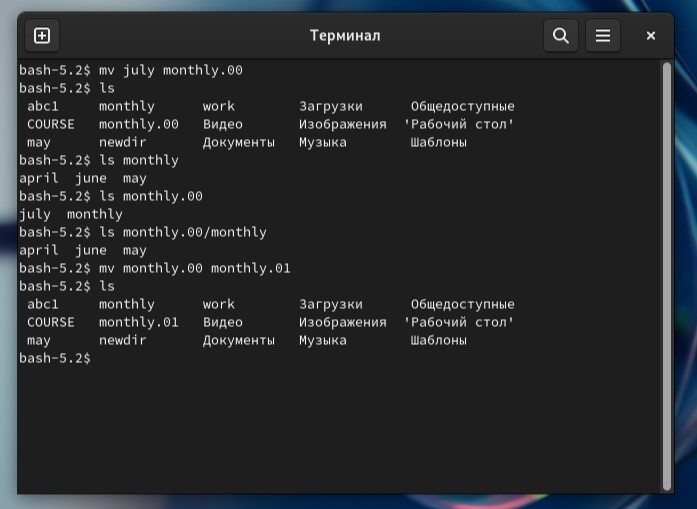


Рисунок 1.5. Перемещение файла july в каталог monthly.00, переименование каталога monthly.00 в monthly.01

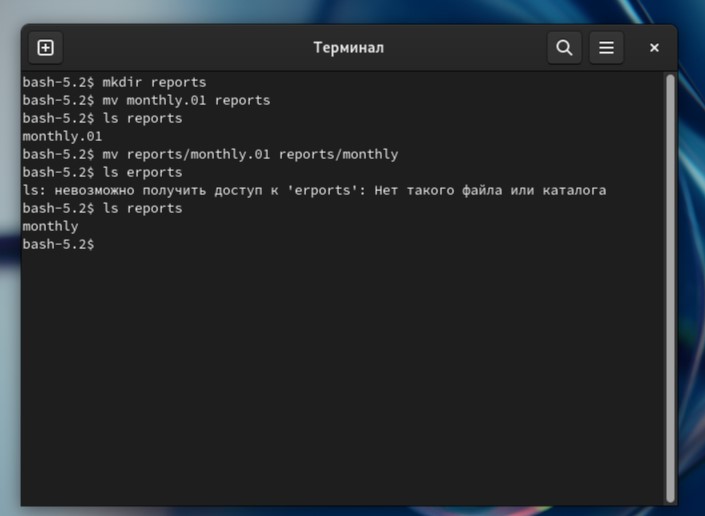


Рисунок 1.6. Перемещение каталога monthly.01 в каталог reports, переименование каталога reports/monthly.01 в reports/monthly

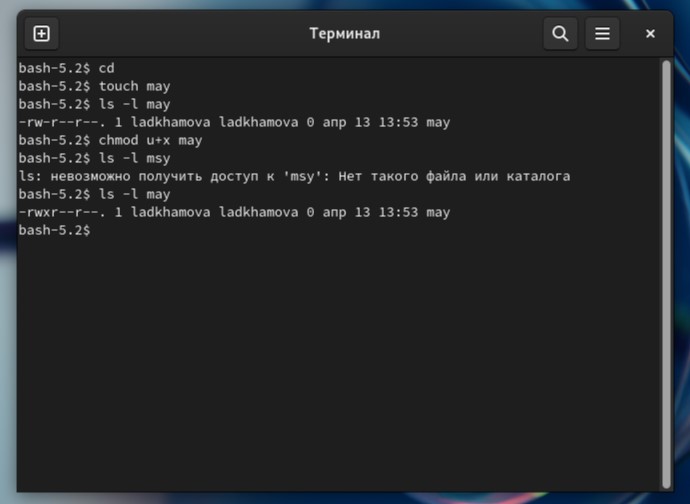


Рисунок 1.7. Требуется создать файл ~/may с правом выполнения для владельца.

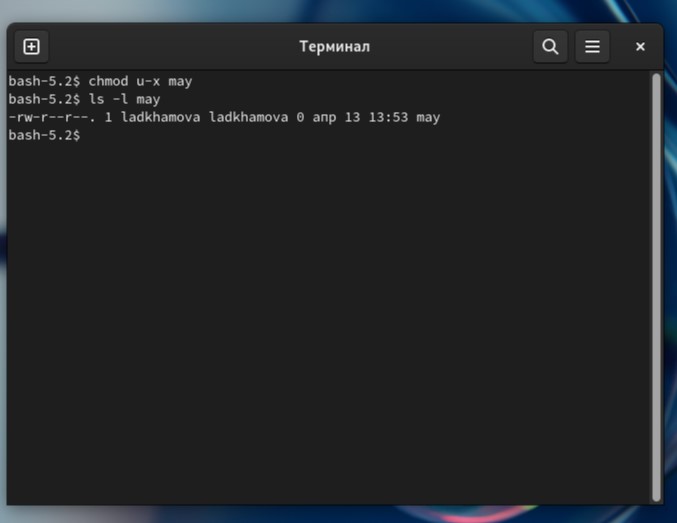


Рисунок 1.8. Требуется лишить владельца файла ~/may права на выполнение

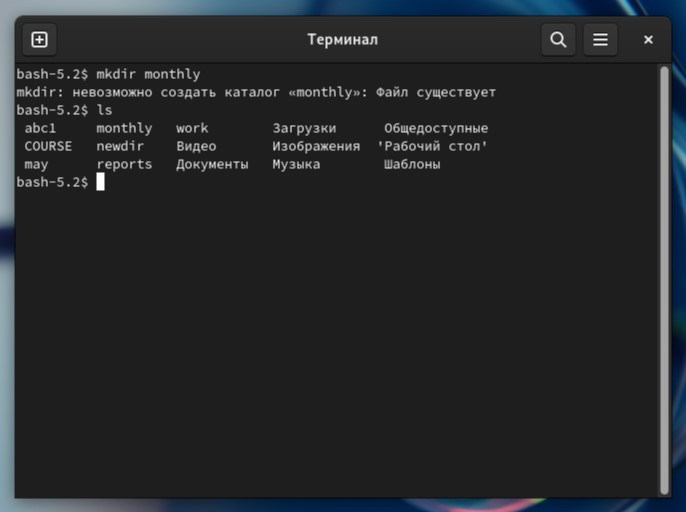


Рисунок 1.9. Требуется создать каталог monthly с запретом на чтение для членов группы и всех остальных пользователей

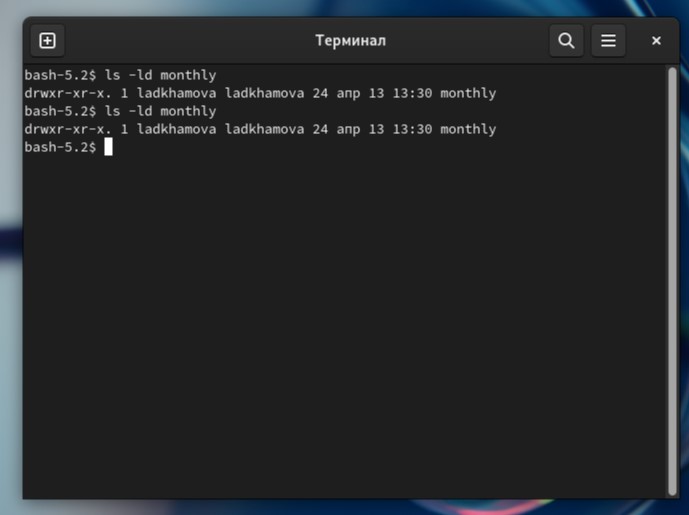


Рисунок 1.10. Требуется создать каталог monthly с запретом на чтение для членов группы и всех остальных пользователей

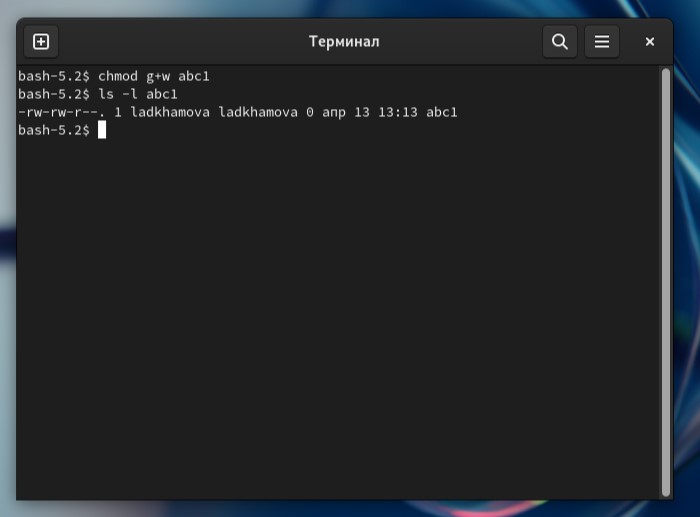


Рисунок 1.11. Требуется создать файл ~/abc1 с правом записи для членов группы

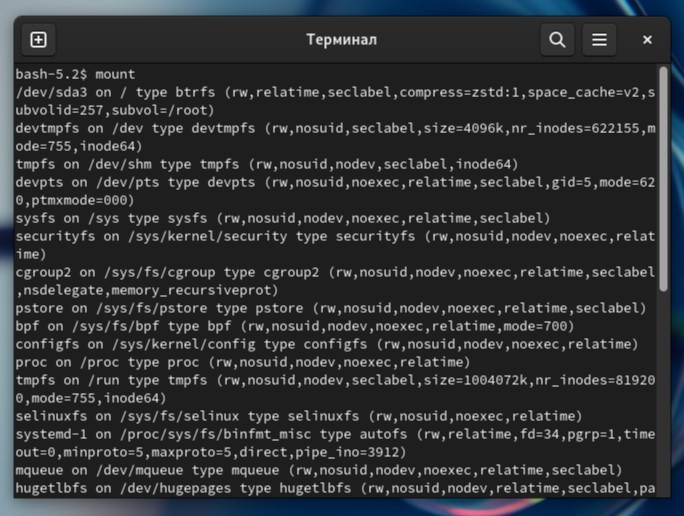


Рисунок 1.12. Для просмотра используемых в операционной системе файловых систем можно воспользоваться командой mount без параметров

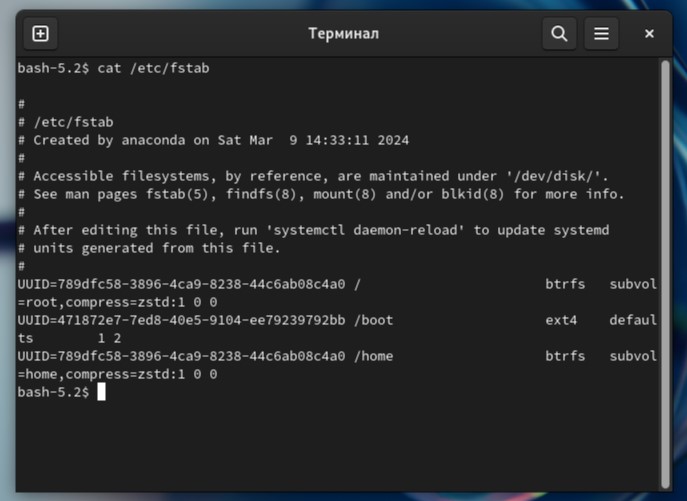


Рисунок 1.13. Другой способ определения смонтированных в операционной системе файловых систем — просмотр файла/etc/fstab

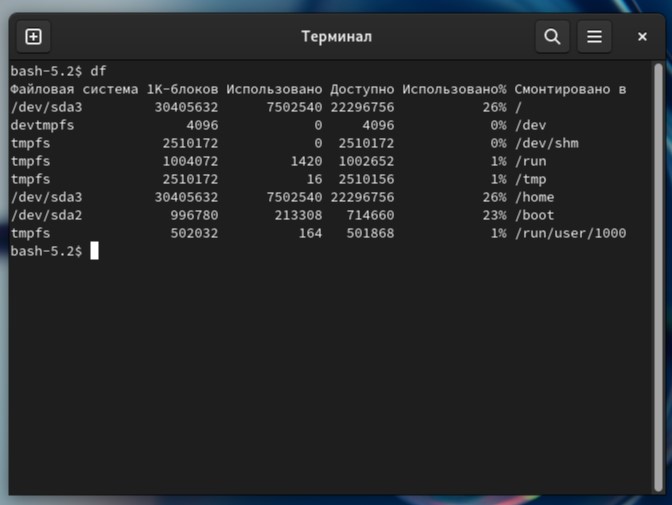


Рисунок 1.14. Для определения объёма свободного пространства на файловой системе можно воспользоваться командой df, которая выведет на экран список всех файловых систем в соответствии с именами устройств, с указанием размера и точки монтирования

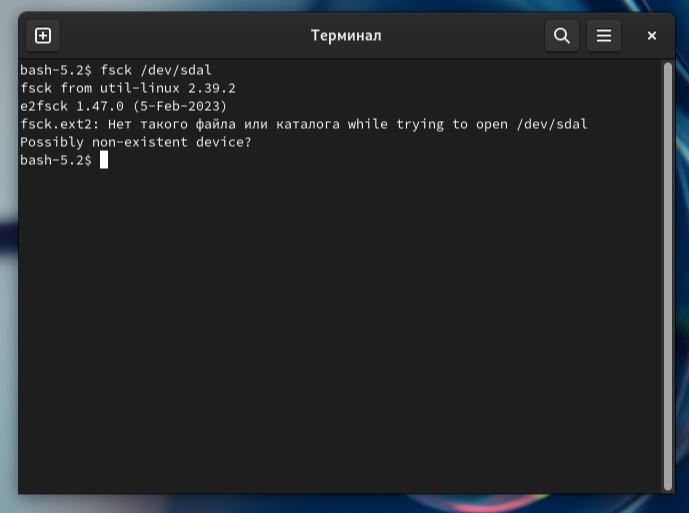


Рисунок 1.15. С помощью команды fsck можно проверить (а в ряде случаев восстановить) целостность файловой системы

2. Выполним следующие действия, зафиксировав в отчёте по лабораторной работе используемые при этом команды и результаты их выполнения:

2.1. Скопируйем файл /usr/include/sys/io.h в домашний каталог и назовем его equipment. Если файла io.h нет, то используем любой другой файл в каталоге /usr/include/sys/ вместо него (рис. 2.1):

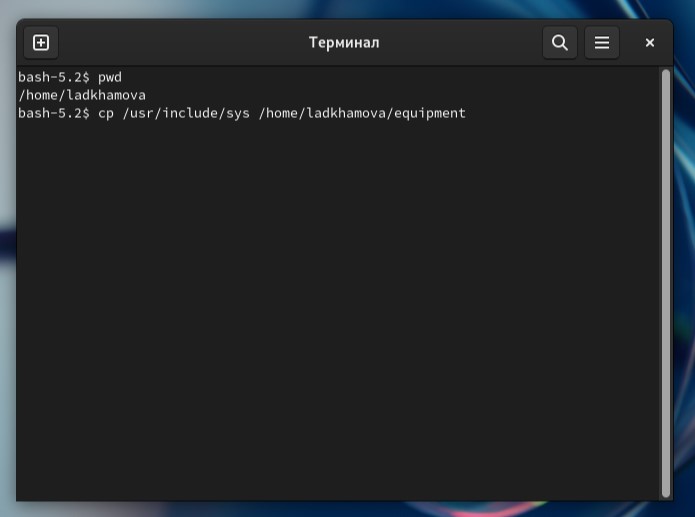


Рисунок 2.1. Копирование файла.

2.2. В домашнем каталоге создадим директорию ~/ski.plases (рис. 2.2):

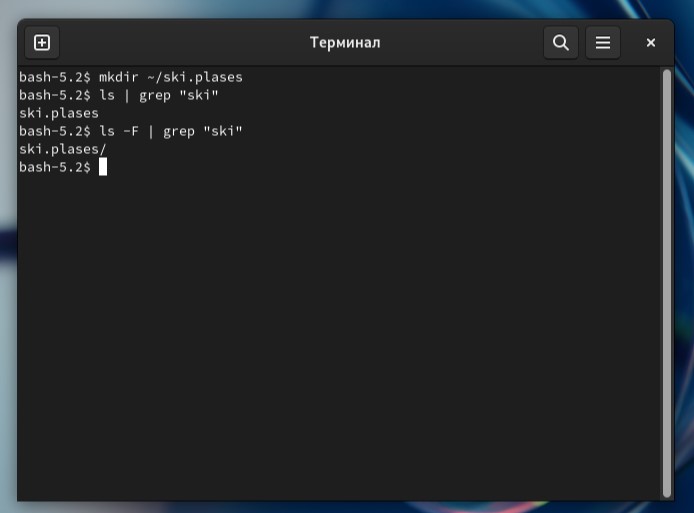


Рисунок 2.2. Создание каталога.

2.3. Переместим файл equipment в каталог ~/ski.plases (рис. 2.3):

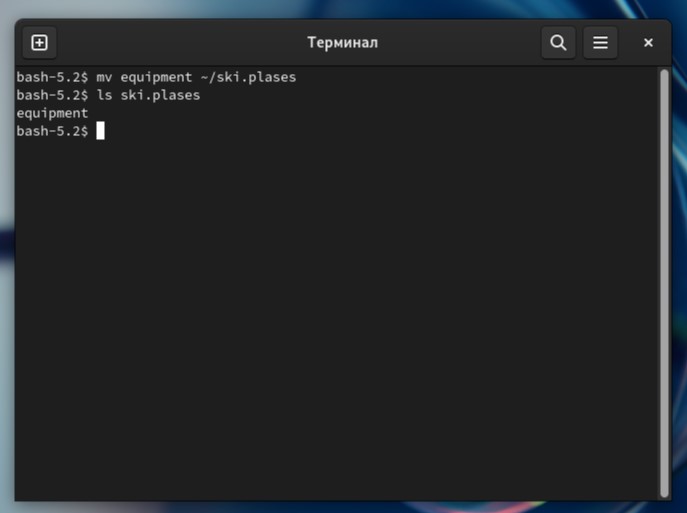


Рисунок 2.3. Перемещение файла.

2.4. Переименуем файл ~/ski.plases/equipment в ~/ski.plases/equiplist (рис. 2.4):

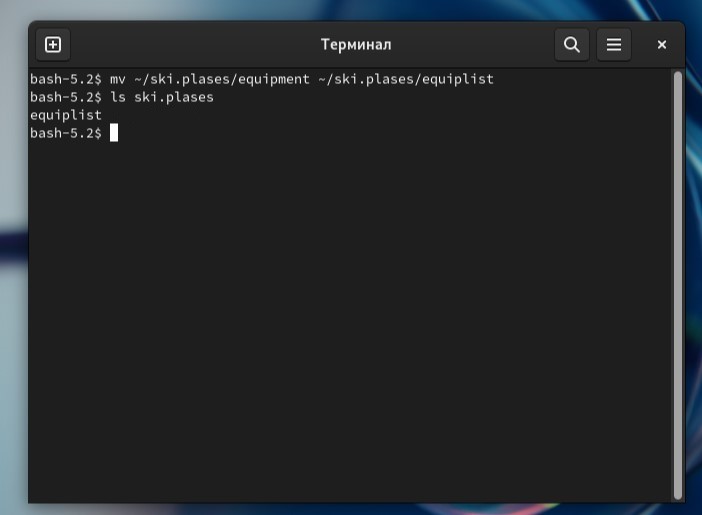


Рисунок 2.4. Переименование файла.

2.5. Создадим в домашнем каталоге файл abc1 и скопируем его в каталог ~/ski.plases, назовем его equiplist2 (рис 2.5):

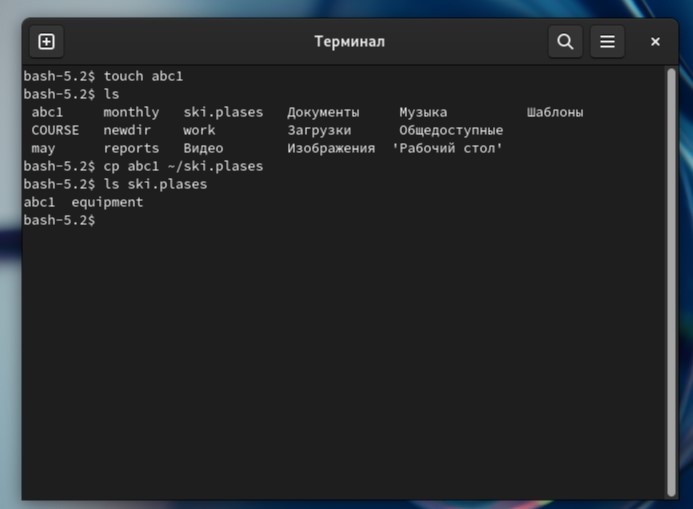


Рисунок 2.5. Создание каталога abc1 и копирование его в ~/ski.plases.

2.6. Создадим каталог с именем equipment в каталоге ~/ski.plases (рис. 2.6):

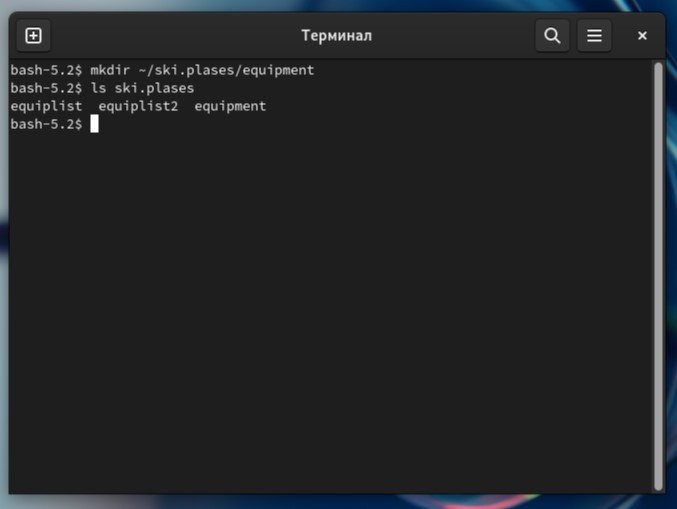


Рисунок 2.6. Создание каталога equipment.

2.7. Переместим файлы ~/ski.plases/equiplist и equiplist2 в каталог ~/ski.plases/equipment (рис. 2.7.):

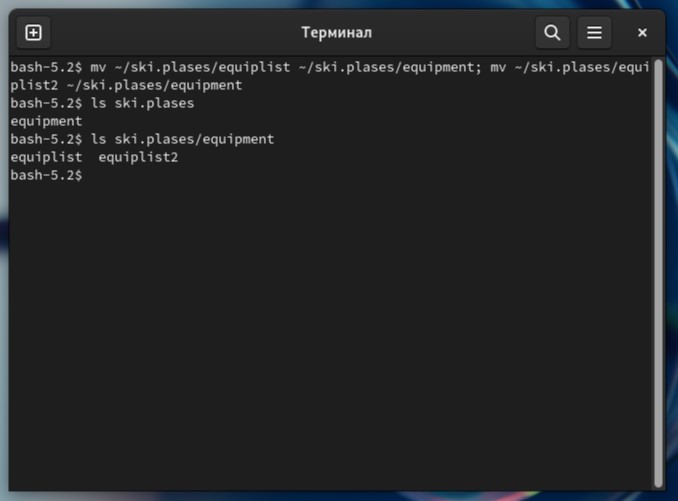


Рисунок 2.7. ~/ski.plases/equiplist и equiplist2 в каталог ~/ski.plases/equipment

2.8. Создадим и переместим каталог ~/newdir в каталог ~/ski.plases и назовем его plans (рис. 2.8):

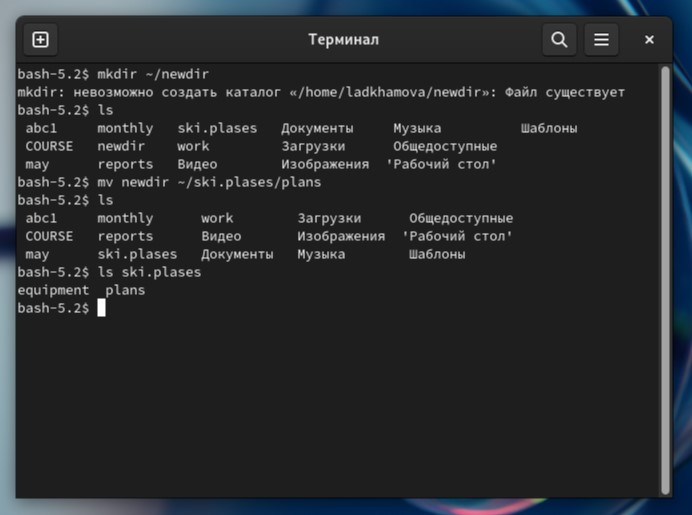


Рисунок 2.8. Создание и перемещение.

3. Определим опции команды chmod, необходимые для того, чтобы присвоить перечисленным ниже файлам выделенные права доступа, считая, что в начале таких прав нет:

3.1. drwxr--r-- ... australia (рис. 3.1):

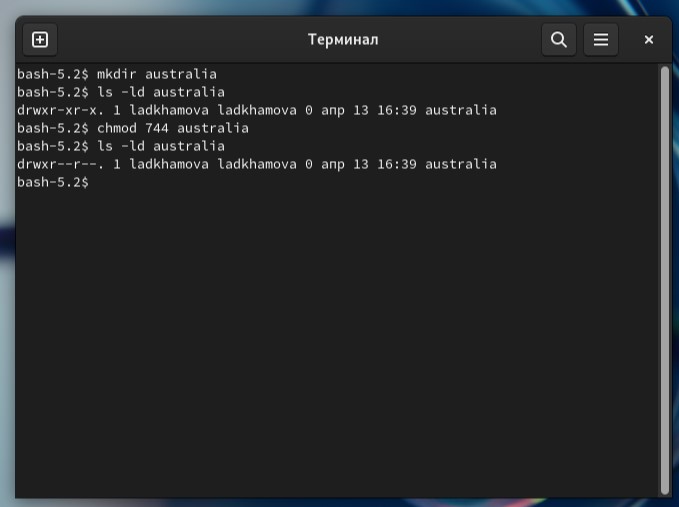


Рисунок 3.1. drwxr--r-- ... australia.

3.2. drwx--x--x ... play (рис. 3.2):

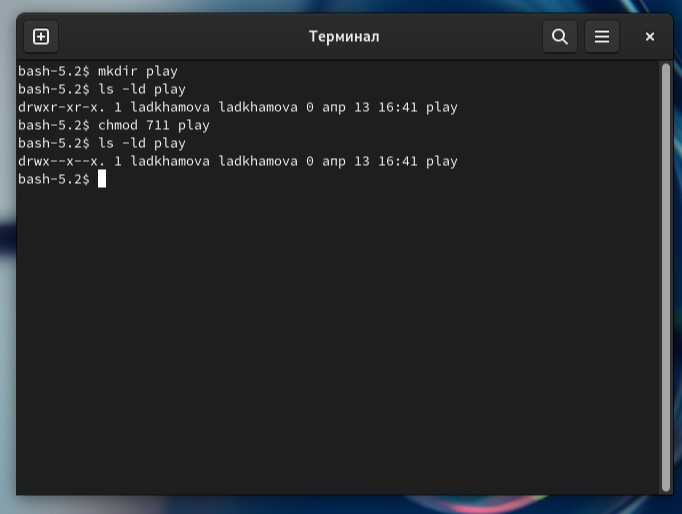


Рисунок 3.2. drwx--x--x ... play.

3.3. -r-xr--r-- ... my\_os (рис. 3.3):

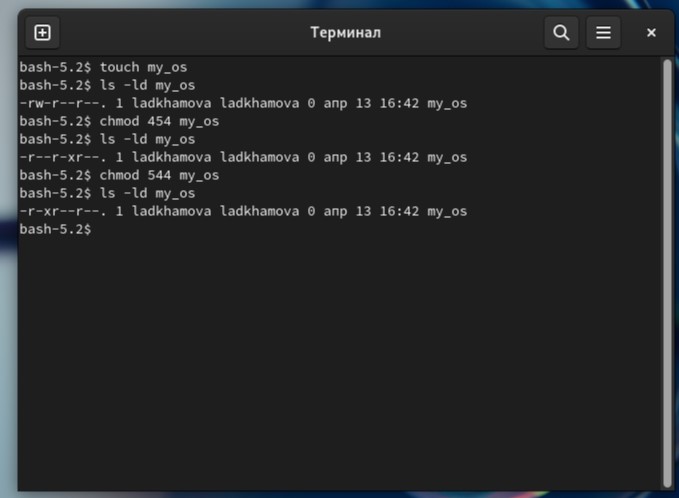


Рисунок 3.3. -r-xr--r-- ... my\_os.

3.4. -rw-rw-r-- ... feathers При необходимости создадим нужные файлы (рис. 3.4):

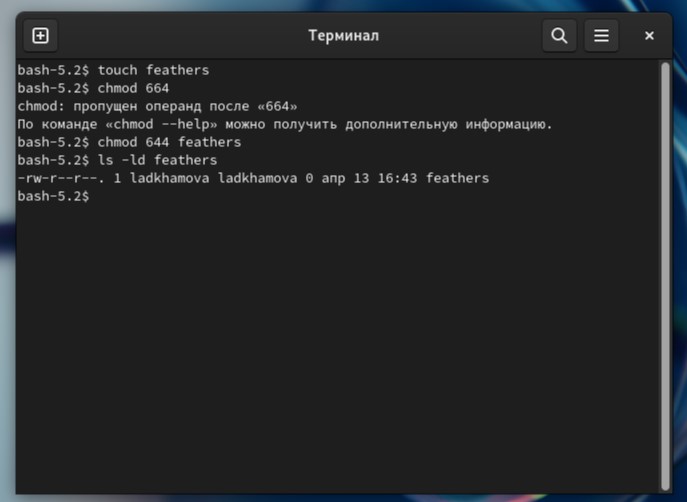


Рисунок 3.4. -rw-rw-r-- ... feathers.

4. Проделаем приведённые ниже упражнения, записывая в отчёт по лабораторной работе используемые при этом команды:

4.1. Просмотрим содержимое файла /etc/password (рис. 4.1, 4.2):

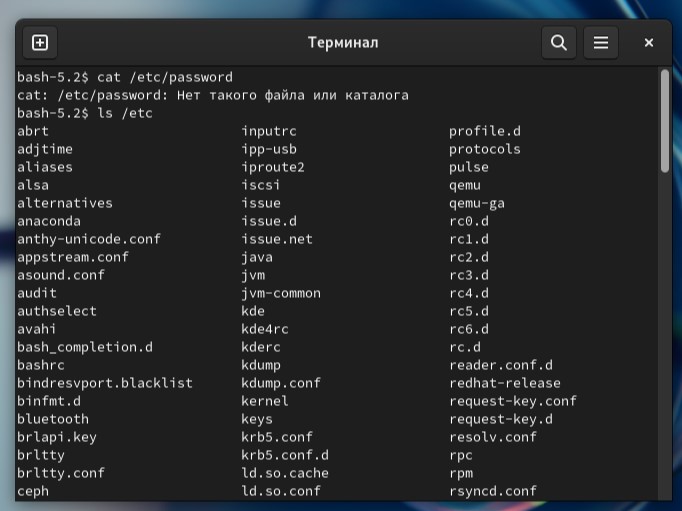


Рисунок 4.1. Выполнение команды /etc/password.

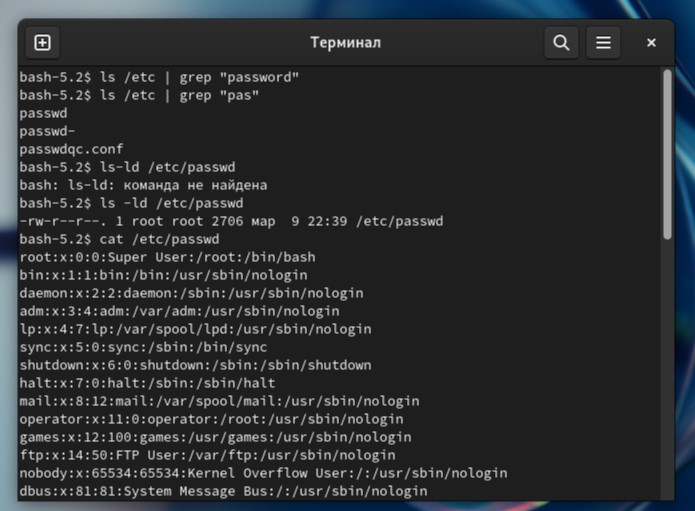


Рисунок 4.2. Продолжение.

4.2. Скопируем файл ~/feathers в файл ~/file.old (рис. 4.3, 4.4):

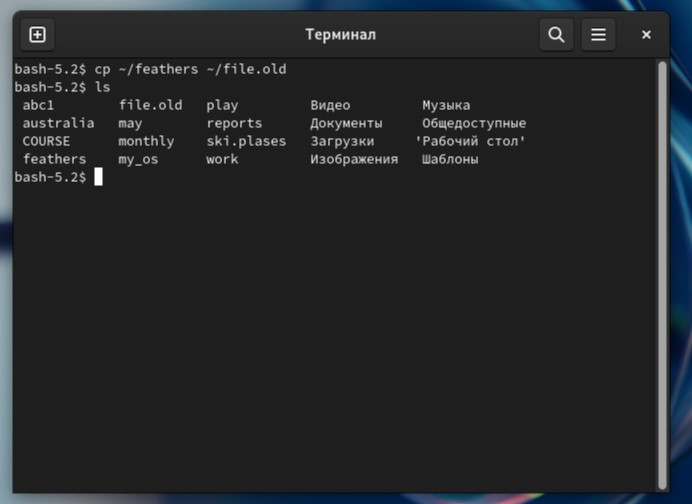


Рисунок 4.3. ~/feathers в файл ~/file.old

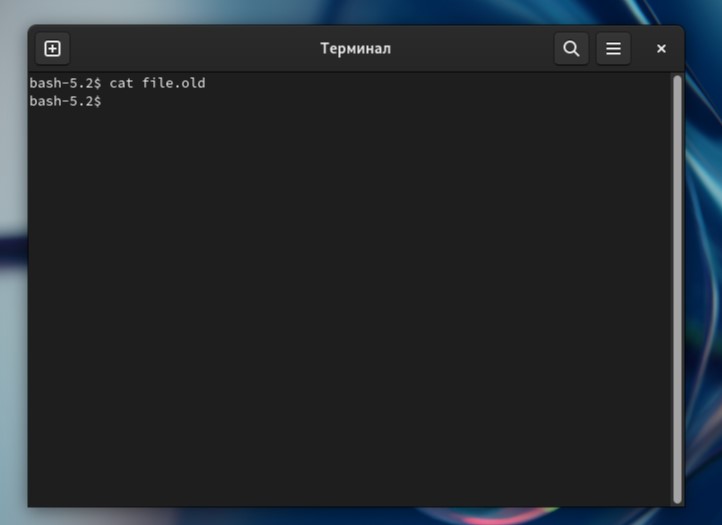


Рисунок 4.4. Продолжение.

4.3. Переместим файл ~/file.old в каталог ~/play (рис. 4.5):

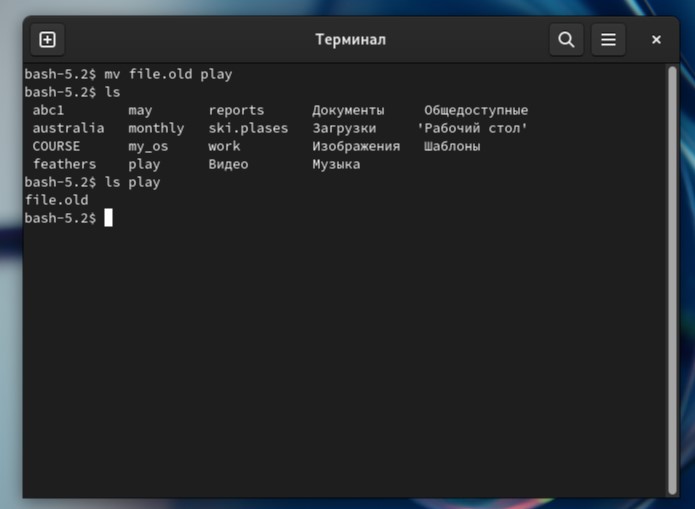


Рисунок 4.5. Перемещение файла.

4.4. Скопируем каталог ~/play в каталог ~/fun (рис. 4.6):

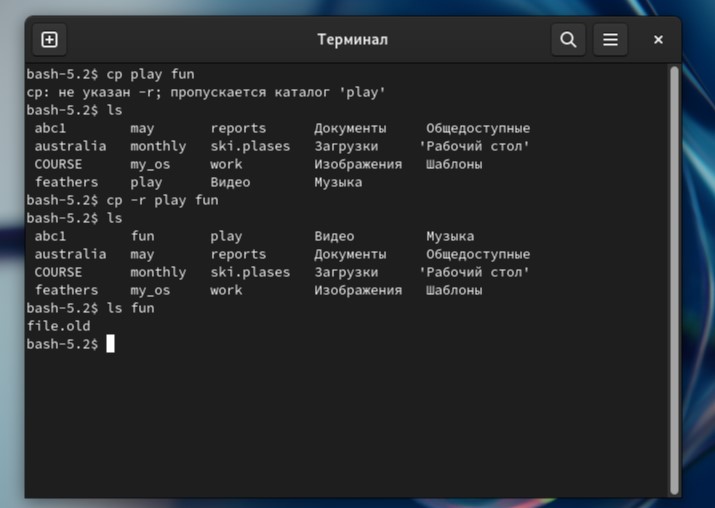


Рисунок 4.6. Копирование.

4.5. Переместим каталог ~/fun в каталог ~/play и назовем его games (рис. 4.7):

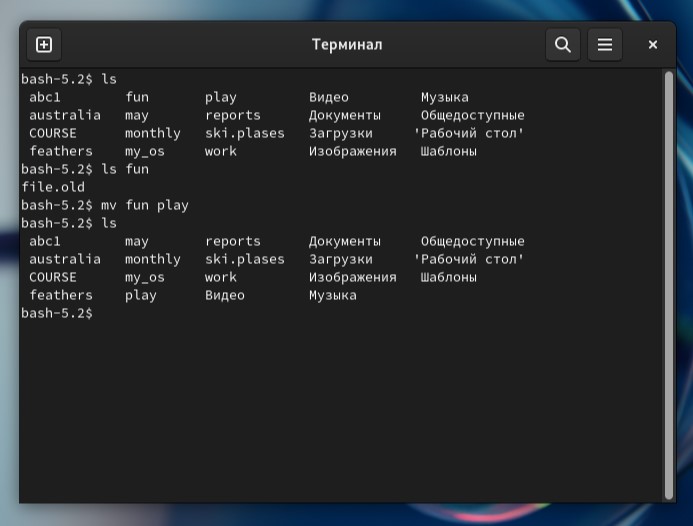


Рисунок 4.7. ~/fun в каталог ~/play.

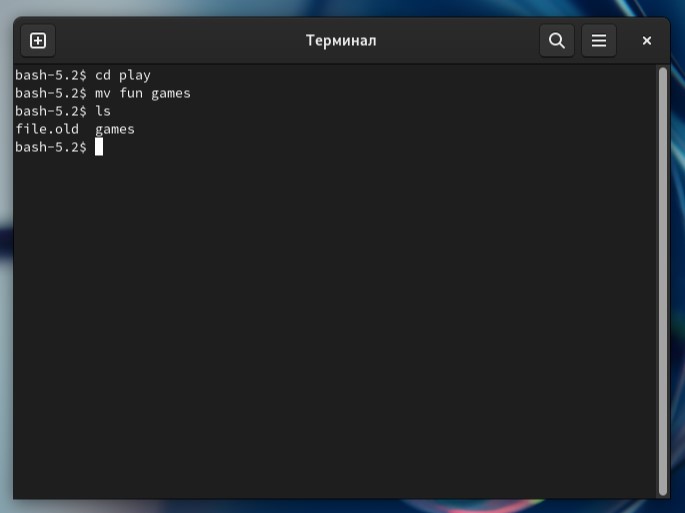


Рисунок 4.8. Называем файл games.

4.6. Лишим владельца файла ~/feathers права на чтение (рис. 4.9):

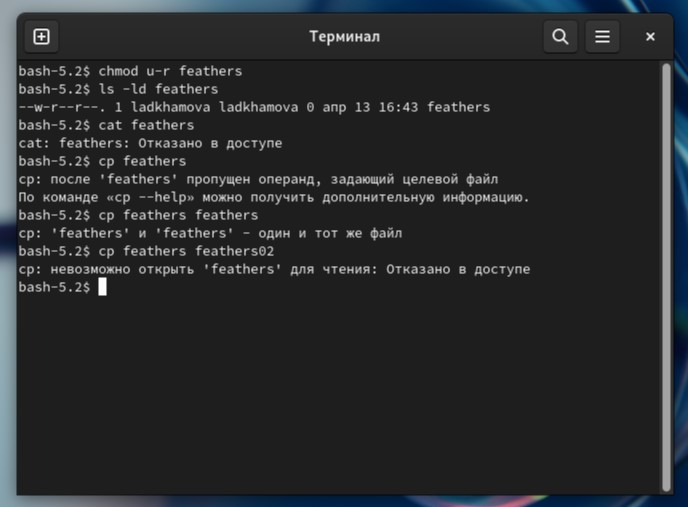


Рисунок 4.9. Лишение права на чтение.

4.7. Что произойдёт, если мы попытаемся просмотреть файл ~/feathers командой cat? (рис. 4.9):

4.8. Что произойдёт, если мы попытаемся скопировать файл ~/feathers? (рис. 4.9):

4.9. Дадим владельцу файла ~/feathers право на чтение (рис. 4.10):

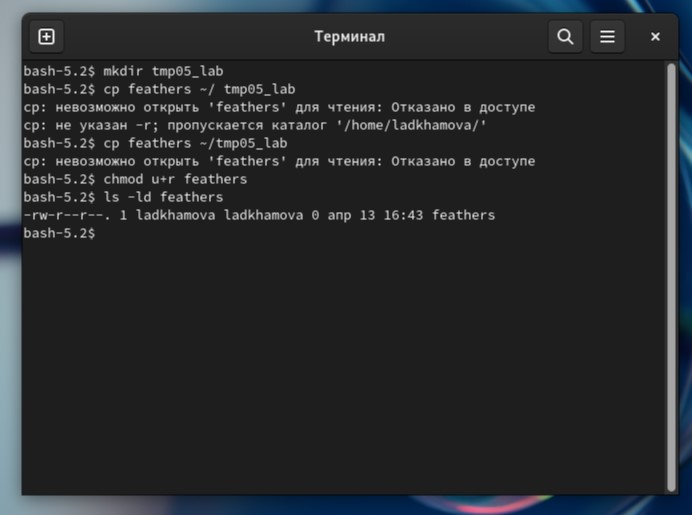


Рисунок 4.10. Дадим право владельцу на чтение.

4.10. Лишим владельца каталога ~/play права на выполнение (рис. 4.11):

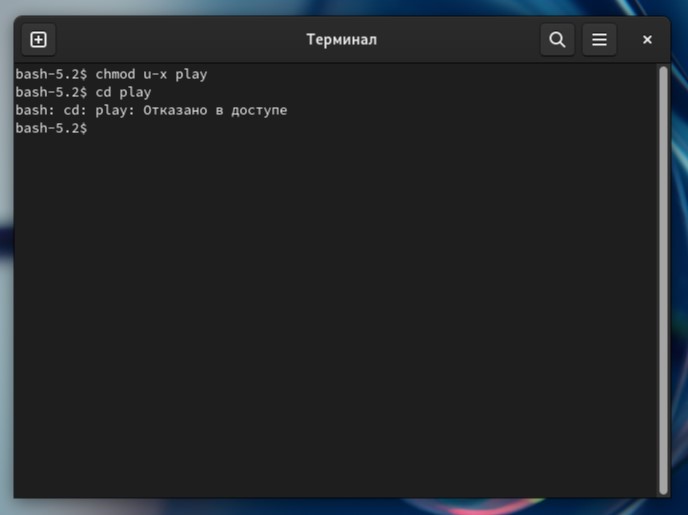


Рисунок 4.11. Лишение владельца права в каталог play.

4.11. Перейдем в каталог ~/play (рис. 4.11):

4.12. Дадим владельцу каталога ~/play право на выполнение (рис. 4.12):

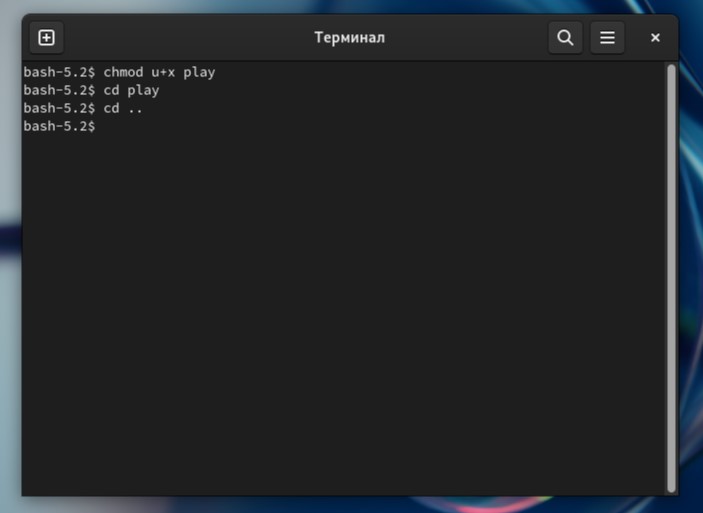


Рисунок 4.12. Дадим владельцу каталога ~/play право на выполнение.

5. Прочитаем man по командам mount, fsck, mkfs, kill и кратко их охарактеризуем, приведя примеры (рис. 5.1, 5.2, 5.3, 5.4):

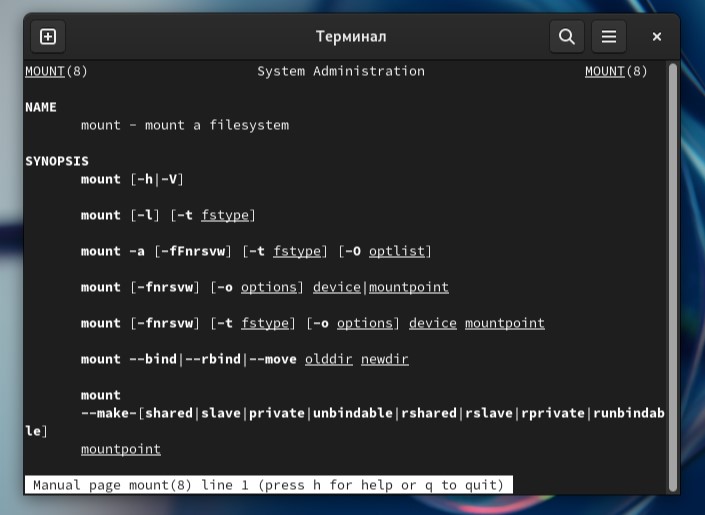


Рисунок 5.1. Выполнение с помощью команды mount.

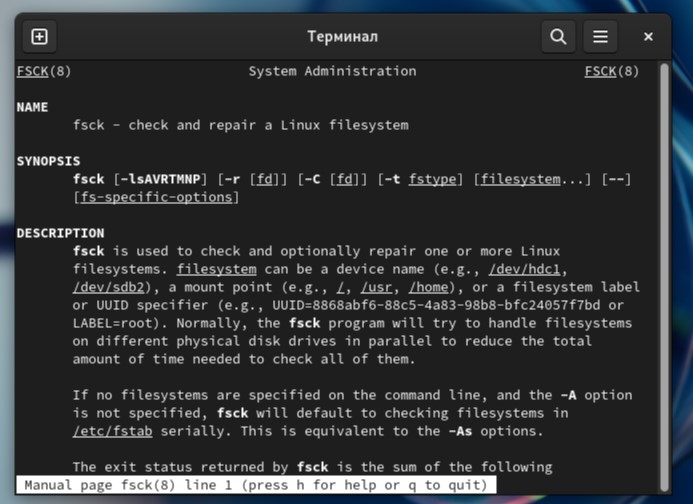


Рисунок 5.2. Выполнение с помощью команды fsck.

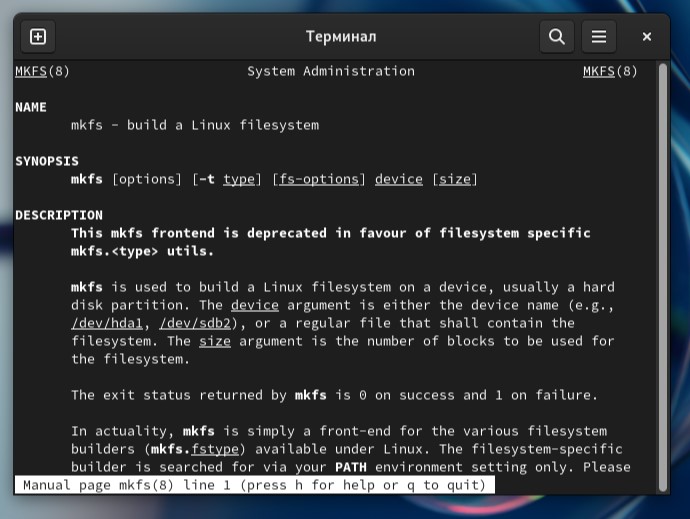


Рисунок 5.3. Выполнение с помощью команды mkfs.

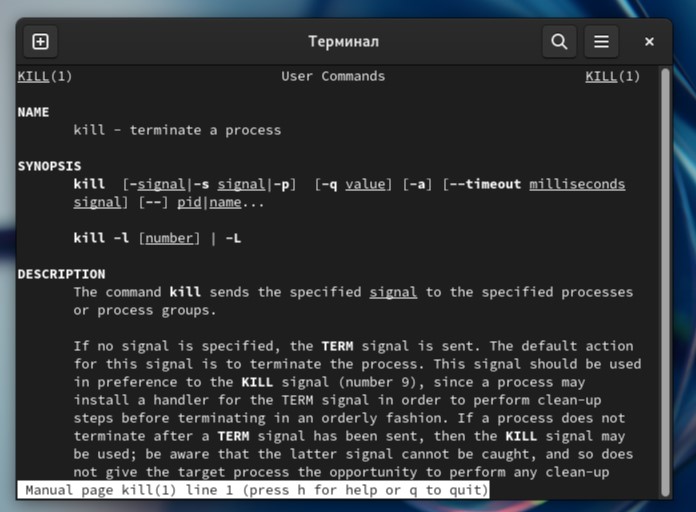


Рисунок 5.3. Выполнение с помощью команды kill.

# Вывод:

Ознакомилась с файловой системой Linux, её структурой, именами и содержанием каталогов. Приобретала практические навыки по применению команд для работы с файлами и каталогами, по управлению процессами (и работами), по проверке использования диска и обслуживанию файловой системы.

# Ответы на контрольные вопросы:

1. Дайте характеристику каждой файловой системе, существующей на жёстком диске компьютера, на котором вы выполняли лабораторную работу:



* proc. Файловая система proc является важным источником информации о вашей Linux-системе, который попросту нельзя игнорировать. Вообще, proc является псевдо- или виртуальной файловой системой, которая предоставляет пользователям доступ к внутренним структурам ядра Linux. Другими словами, proc не является реальной файловой системой в обычном смысле; она располагается исключительно в оперативной памяти, а не на диске. При этом она автоматически монтируется системой.
* Sysfs - отправляет данные в пространство пользователя с помощью виртуальных файлов. Эти данные содержат данные о различных подсистемах ядра, аппаратных устройствах и связанных с ними драйверах устройств.
* tmpfs и devtmpfs - они относятся к энергозависимой памяти.
* devpts - обеспечивает доступ к терминалам pseudo (PTY).
* cgroup2 - Неверно ведущий себя процесс может создавать тонны процессов через ветвления, запуская некую бомбу ветвлений и сокрушая своё ядро. Это означает, что нам требуется ввести некий способ контроля ресурсов для процессов в пределах заданного пространства имён. Это достигается через механизм, носящий название групп контроля (control groups), обычно именуемых cgroups. cgroups работают под понятием контроллеров cgroup и представляются в файловой системе с названием cgroupfs в самом ядре Linux. В настоящее время применяется cgroup v2 версия cgroups.
* pstore - был введен в Linux для записи информации (например, dmesg tail) при выключении. Pstore не зависит от kdump и может запускаться до него. В определенных сценариях (например, хосты/гости с корневыми файловыми системами на NFS/iSCSI, где произошел сбой сетевого программного и/или аппаратного обеспечения), pstore может содержать информацию, доступную для посмертной отладки, которая не может быть получена иным образом.
* bpf - это псевдо-файловая система, существующая только в памяти, которая позволяет создавать файлы, ссылающиеся на объекты BPF.
* btrfs - файловая система для Linux, основанная на структурах B-деревьев и работающая по принципу «копирование при записи» (copy-on-write). Опубликована корпорацией Oracle в 2007 году под лицензией GNU General Public License.
* selinux - Как и файловая система /proc, /selinux является псевдофайловой системой. Новая реализация SE Linux использует расширенные атрибуты для хранения контекста безопасности.
* tracefs - файловая система для задач трассировки Linux
* debugfs - DebugFS является самой известной утилитой, предназначенной для работы с файловыми системами EXT2FS и EXT3FS.
* hugetlb - использует страницы большого размера, что позволяет кэшировать больше адресов за раз.
* mqueue - обеспечивает необходимую поддержку ядра для библиотеки пользовательского пространства, которая реализует интерфейсы очереди сообщений POSIX.
* fusectl - это простой интерфейс для программ пользовательского пространства для экспорта виртуальной файловой системы в ядро Linux.
* configf - Виртуальная файловая система, представляющая состояние ядра операционной системы и запущенных процессов в виде файлов.
* ext4 - журналируемая файловая система, используемая преимущественно в операционных системах с ядром Linux, созданная на базе ext3 в 2006 году.
* fuse.gv, fuse.po, fuse.je - FUSE (файловая система в пользовательском пространстве) — это интерфейс для программ пользовательского пространства для экспорта файловой системы в ядро ​​Linux.

1. Приведите общую структуру файловой системы и дайте характеристику каждой ди- ректории первого уровня этой структуры.

/ — root каталог. Содержит в себе всю иерархию системы;

/bin — здесь находятся двоичные исполняемые файлы. Основные общие команды, хранящиеся отдельно от других программ в системе (прим.: pwd, ls, cat, ps);

/boot — тут расположены файлы, используемые для загрузки системы (образ initrd, ядро vmlinuz);

/dev — в данной директории располагаются файлы устройств (драйверов). С помощью этих файлов можно взаимодействовать с устройствами. К примеру, если это жесткий диск, можно подключить его к файловой системе. В файл принтера же можно написать напрямую и отправить задание на печать;

/etc — в этой директории находятся файлы конфигураций программ. Эти файлы позволяют настраивать системы, сервисы, скрипты системных демонов;

/home — каталог, аналогичный каталогу Users в Windows. Содержит домашние каталоги учетных записей пользователей (кроме root). При создании нового пользователя здесь создается одноименный каталог с аналогичным именем и хранит личные файлы этого пользователя;

/lib — содержит системные библиотеки, с которыми работают программы и модули ядра;

/lost+found — содержит файлы, восстановленные после сбоя работы системы. Система проведет проверку после сбоя и найденные файлы можно будет посмотреть в данном каталоге;

/media — точка монтирования внешних носителей. Например, когда вы вставляете диск в дисковод, он будет автоматически смонтирован в директорию /media/cdrom;

/mnt — точка временного монтирования. Файловые системы подключаемых устройств обычно монтируются в этот каталог для временного использования;

/opt — тут расположены дополнительные (необязательные) приложения. Такие программы обычно не подчиняются принятой иерархии и хранят свои файлы в одном подкаталоге (бинарные, библиотеки, конфигурации);

/proc — содержит файлы, хранящие информацию о запущенных процессах и о состоянии ядра ОС;

/root — директория, которая содержит файлы и личные настройки суперпользователя;

/run — содержит файлы состояния приложений. Например, PID-файлы или UNIX-сокеты;

/sbin — аналогично /bin содержит бинарные файлы. Утилиты нужны для настройки и администрирования системы суперпользователем;

/srv — содержит файлы сервисов, предоставляемых сервером (прим. FTP или Apache HTTP);

/sys — содержит данные непосредственно о системе. Тут можно узнать информацию о ядре, драйверах и устройствах;

/tmp — содержит временные файлы. Данные файлы доступны всем пользователям на чтение и запись. Стоит отметить, что данный каталог очищается при перезагрузке;

/usr — содержит пользовательские приложения и утилиты второго уровня, используемые пользователями, а не системой. Содержимое доступно только для чтения (кроме root). Каталог имеет вторичную иерархию и похож на корневой;

/var — содержит переменные файлы. Имеет подкаталоги, отвечающие за отдельные переменные. Например, логи будут храниться в /var/log, кэш в /var/cache, очереди заданий в /var/spool/ и так далее.

1. Какая операция должна быть выполнена, чтобы содержимое некоторой файловой системы было доступно операционной системе?

Монтирование тома.

1. Назовите основные причины нарушения целостности файловой системы. Как устранить повреждения файловой системы?

Отсутствие синхронизации между образом файловой системы в памяти и ее данными на диске в случае аварийного останова может привести к появлению следующих ошибок:

1. Один блок адресуется несколькими mode (принадлежит нескольким файлам).  
  
 2. Блок помечен как свободный, но в то же время занят (на него ссылается onode).  
  
 3. Блок помечен как занятый, но в то же время свободен (ни один inode на него не ссылается).  
  
 4. Неправильное число ссылок в inode (недостаток или избыток ссылающихся записей в каталогах).  
  
 5. Несовпадение между размером файла и суммарным размером адресуемых inode блоков.  
  
 6. Недопустимые адресуемые блоки (например, расположенные за пределами файловой системы).  
  
 7. "Потерянные" файлы (правильные inode, на которые не ссылаются записи каталогов).  
  
 8. Недопустимые или неразмещенные номера inode в записях каталогов.

1. Как создаётся файловая система?

* mkfs - позволяет создать файловую систему Linux.

Создать файловую систему linux, семейства ext, на устройстве можно с помощью команды mkfs. Ее синтаксис выглядит следующим образом:

sudo mkfs -t тип устройство

Доступны дополнительные параметры:

* с - проверить устройство на наличие битых секторов
* b - размер блока файловой системы
* j - использовать журналирование для ext3
* L - задать метку раздела
* v - показать подробную информацию о процессе работы
* V - версия программы

1. Дайте характеристику командам для просмотра текстовых файлов.

* cat <имя\_файла> - Это самая простая и, пожалуй, самая популярная команда для просмотра файла в Linux. Cat просто печатает содержимое файла на стандартном экране, т.е. на экране. В основном используется для небольших файлов.
* Less <имя\_файла>. Команда Less просматривает файл по одной странице за раз.
* Head <имя\_файла>. Команда Head — это еще один способ просмотра текстового файла, но с небольшой разницей. Команда head отображает первые 10 строк текстового файла по умолчанию. Вы можете изменить это поведение, используя опции с командой head, но основной принцип остается тем же: команда head начинает работать с заголовка (начала) файла.
* Tail <имя\_файла>. Команда Tail в Linux аналогична и все же противоположна команде head. В то время как команда head отображает файл с начала, команда tail отображает файл с конца. По умолчанию команда tail отображает последние 10 строк файла. Команды Head и Tail могут быть объединены для отображения выбранных строк из файла. Вы также можете использовать команду tail для просмотра изменений, внесенных в файл в режиме реального времени.

1. Приведите основные возможности команды cp в Linux.

Это сокращение от copy, и она делает именно то, что предполагает ее название: она копирует. cp используется для копирования файлов из одного местоположения в другое. cp также можно использовать для копирования всех каталогов в новое место. Можно использовать эту команду для копирования нескольких файлов и каталогов.

1. Приведите основные возможности команды mv в Linux.

Команда mv используется для перемещения файлов из одного каталога в другой. Также команда mv используется для переименования файла в системах Linux.

1. Что такое права доступа? Как они могут быть изменены?

Права доступа определяют, какие действия конкретный пользователь может или не может совершать с определенным файлами и каталогами.

Каждый файл можно изменять по трём параметра доступа. Вот они:

* Чтение - разрешает получать содержимое файла, но на запись нет. Для каталога позволяет получить список файлов и каталогов, расположенных в нем;
* Запись - разрешает записывать новые данные в файл или изменять существующие, а также позволяет создавать и изменять файлы и каталоги;
* Выполнение - вы не можете выполнить программу, если у нее нет флага выполнения. Этот атрибут устанавливается для всех программ и скриптов, именно с помощью него система может понять, что этот файл нужно запускать как программу.

Чтобы получить доступ к файлам в Linux, используются разрешения. Эти разрешения назначаются трем объектам: файлу, группе и другому объекту (то есть всем остальным). Изменить права доступа можно при помощи команды **chmod**:

chmod <параметры изменения> <имя\_файла/каталога>