РУДН. Операционные системы

Отчёт по лабораторной работе №6

Косинов Никита Андреевич, НПМбв-02-20

Содержание

# 1 Цель работы

В прошлой лабораторной работе мы познакомились со структурой файловой системы. Кроме открытия, чтения и работы с правами доступа необходимо быстро находить необходимые документы, и знать, как их записывать, какие команды и как можно применять к дереву каталогов.

Цель данной работы - приобретение теоретических и практических навыков по работе с файловой ситемой из командной строки на примере ОС **Linux**.

# 2 Ход работы

Лабораторная работа выполнена в терминале **OC Linux** и хостинге хранения проектов **Github**. Действия по лабораторной работе представлены в следующем порядке:

1. перенаправление ввода-вывода;
2. фильтрация текста;
3. поиск файла;
4. обработка информации о процессах.

По завершении отчёта, вся рабочая папка отправляется на репозиторий на *github*.

# 3 Перенаправление ввода-вывода

Перед началом работы, по обычаю, получим обновления из **git** командой *git pull*, и создадим рабочую папку **lab02-06** командой *mkdir*.

Синхронизация рабочего пространства с git

Рис. 1: Синхронизация рабочего пространства с git

Выполняем примеры из лабораторной работы по перенаправлению вывода.

1. Записываем все файлы, содержащиеся в каталоге **/etc**, используя стандартную команду *ls* вывода содержимого с применением перенаправления записи *>* в файл **file.txt**.

Перенаправление вывода

Рис. 2: Перенаправление вывода

1. Просмотрим полученный результат командой *cat*.

Просмотр содержимого документа

Рис. 3: Просмотр содержимого документа



Рис. 4: Просмотр содержимого документа

1. Дозапишем в тот же файл все файлы из домашней директории. Для дозаписи используем команду *>>*

Дозапись с перенаправлением

Рис. 5: Дозапись с перенаправлением

1. Просмотрим содержимое, чтобы убедиться, что команда выполнена корректно.

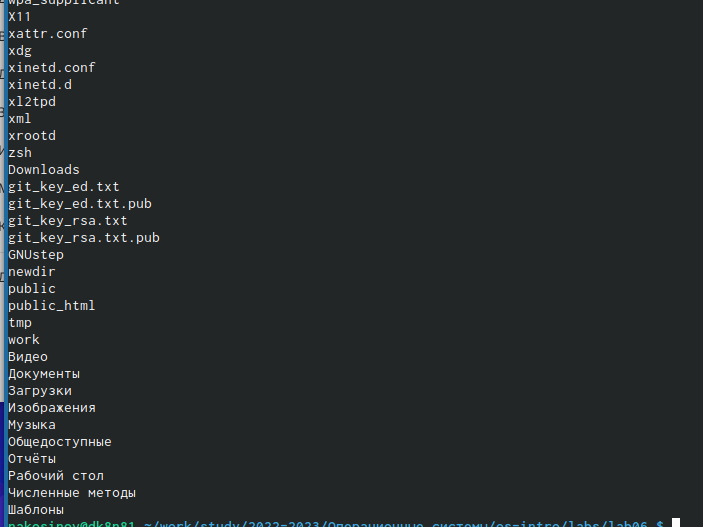


Рис. 6: Просмотр содержимого документа

# 4 Фильтрация текста

Данная часть работы посвящена поиску файлов с определёнными текстовыми маркерами.

1. Командой *grep* ищем в недавно созданном **file.txt** все файлы с расширением **.conf**: это сочетание символов должно быть в названии. Запишем результаты поиска в новый файл **conf.txt**.

Поиск файлов определённого расширения

Рис. 7: Поиск файлов определённого расширения

1. Открываем **conf.txt**, чтобы убедиться в корректности выполненния команды.

Открытие файла

Рис. 8: Открытие файла



Рис. 9: Содержимое файла

# 5 Поиск файла

Чаще при работе с файловой структорой требуется натйти определённый файл, нежели найти символы в списке. Поэтому выполним несколько заданий на поиск.

1. Первое задание: найти файлы в домашнем каталоге, начинающиеся на символ **c**. Первая мысль - сделать это командой *find*, применённой к домашней директории с фильтром \*\* c\* \*\* и вывести на экран модификатором *-print*.

Поиск файла в каталоге по маске

Рис. 10: Поиск файла в каталоге по маске

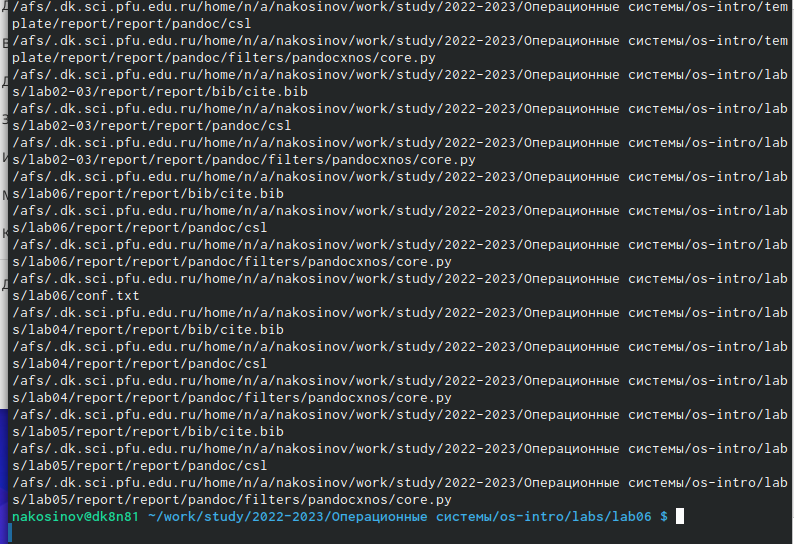


Рис. 11: Результат поиска

1. Как видно выше, команда *find* ищет вообще все файлы внутри каталога, обладающие нужным свойством, не только в корне этого каталога. Поэтому попробуем сделать по другому - командой *grep* в предварительно созданном файле *home.txt*

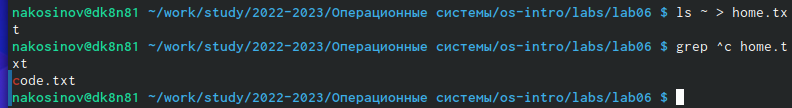


Рис. 12: Поиск файла в созданном списке

1. Однако, создавать список в отдельный файл нецелесообразно. Здесь на помощь приходит конвейер, позволяющий передать в *grep* список файлов напрямую, без промежуточной записи.

Поиск файла по маске конвейером

Рис. 13: Поиск файла по маске конвейером

1. Аналогичным образом, с помощью *grep* и конвейера выведем файлы, начинающиеся на **h** в каталоге **/etc**.

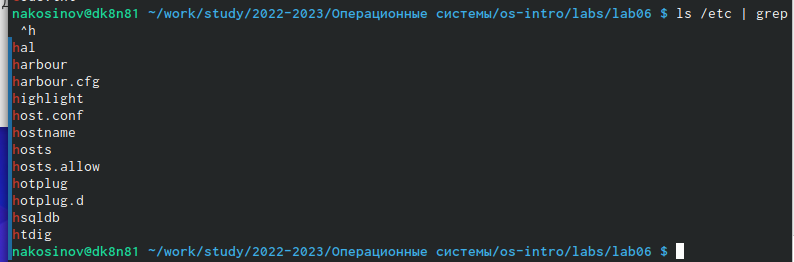


Рис. 14: Поиск файла по маске конвейером

# 6 Обработка информации о процессах

При работе с ЭВМ зачастую требуется управлять процессами: например, чтобы развести их по потокам для параллельной работы. Эффект хорошо виден, когда, например, на мониторе открыто несколько окон. Данная часть работы посвящена управлению и посику процессов

1. Запустим процесс, записыающий в файл **logfile** все файлы каталога **/etc**, начинающиеся с *log*. Для этого поставим в конец команды *&*.

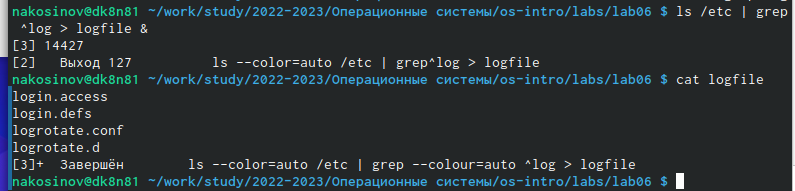


Рис. 15: Запуск фонового процесса

1. Откроем файл для просмотра. На предыдущем скриншоте хорошо видно, что процесс завершился автоматически, как только мы открыли файл на чтение. Действительно, нельзя одновременно и записывать, и читать документ.

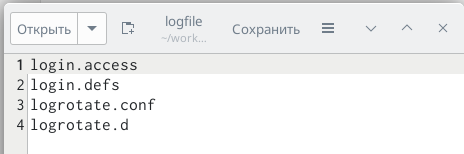


Рис. 16: Просмотр содержимого logfile

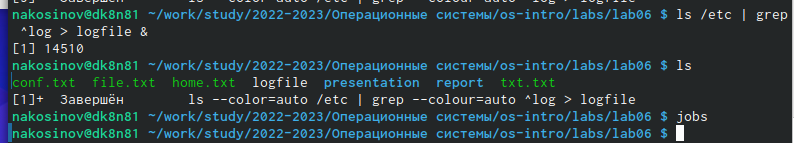


Рис. 17: Проверка запущенных процессов командой jobs

1. Удалим файл **logfile**.

Удаление файла

Рис. 18: Удаление файла

1. Запустим **gedit** в фоновом режиме. Окно редактора откроется поверх консоли, а процессу присвоится идентификатор, который видно внизу скриншота - 14877.

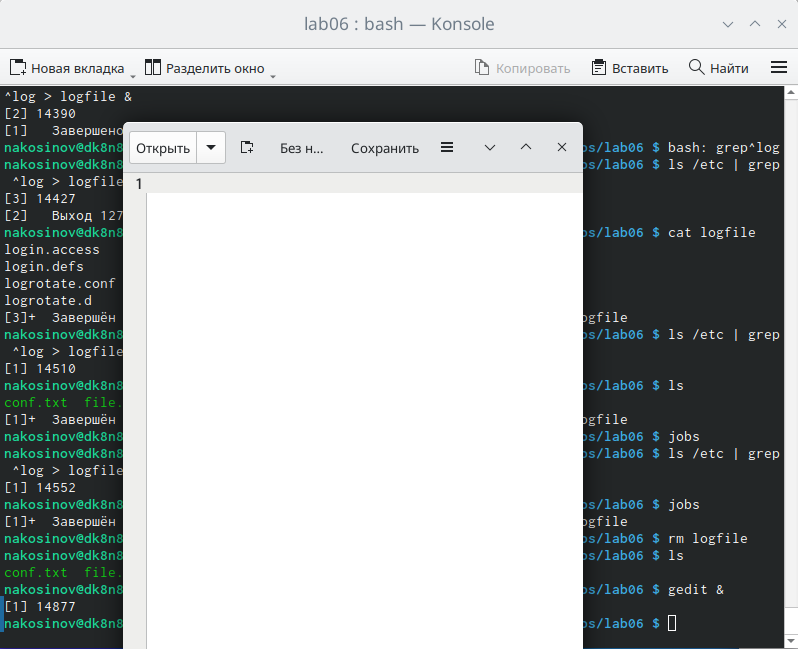


Рис. 19: Запуск фонового процесса

1. Можно получить идентификатор недавно запущенного процесса командой *ps aux*, и промотав в конец списка.

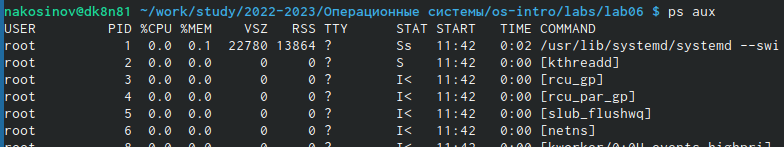


Рис. 20: Вывод идентификаторов процессов

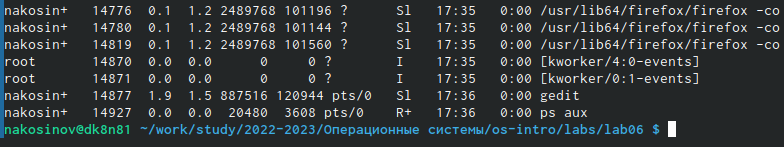


Рис. 21: Вывод идентификаторов процессов

1. Также, можно получить идентификатор командой *grep*, соединённой конвейером с *ps aux*: так можно определить идентификаторы всех процессов, включающих в название **gedit**.

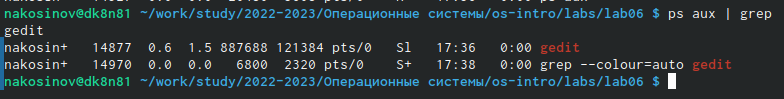


Рис. 22: Поиск идентификатора конкретного процесса

1. Проверим командой *jobs* запущенные процессы. Видим, что нашему присвоен номер **1** - пока это единственный наш фоновый процесс. Завершим его командой *kill*. Снова проверим список наших процессов. Видим, что команда выполнена успешно.

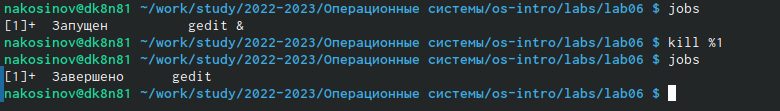


Рис. 23: Завершение процесса

1. Просмотрим, какие файловые системы занимают больше всего места. Для этого воспользуемся командой *df*. Модификатор *-vi* покажет инод с игнорируемыми системами, а модификатор *-l* - предел локальной системы.

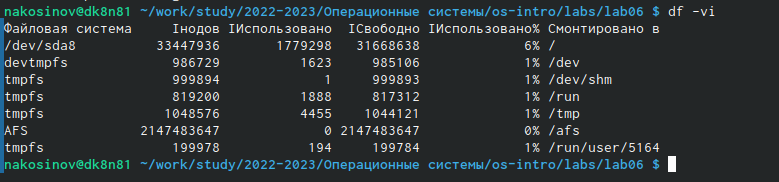


Рис. 24: Использование команды df

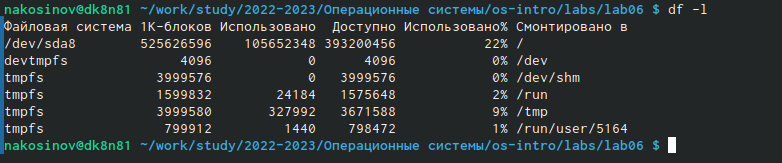


Рис. 25: Использование команды df

1. Командой *du* можно увидеть отчёт об использовании пространства конкретными файлами. Модификатор *-a* покажет все файлы, а не только директории.

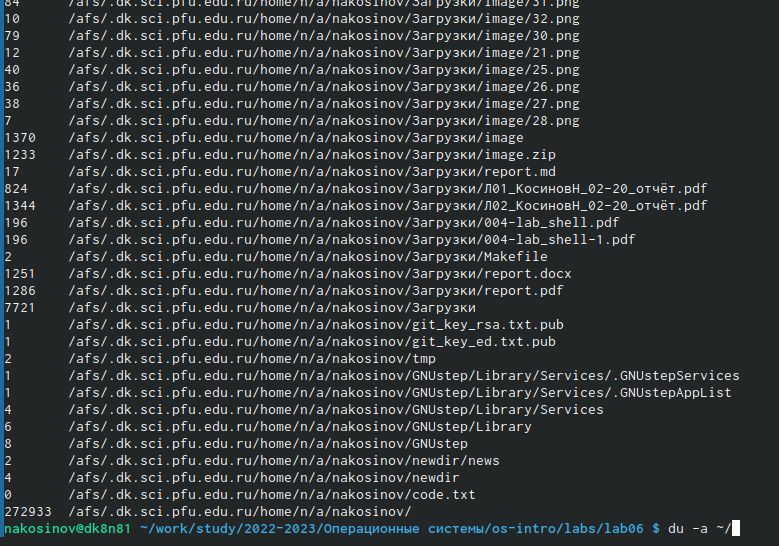


Рис. 26: Использование команды du

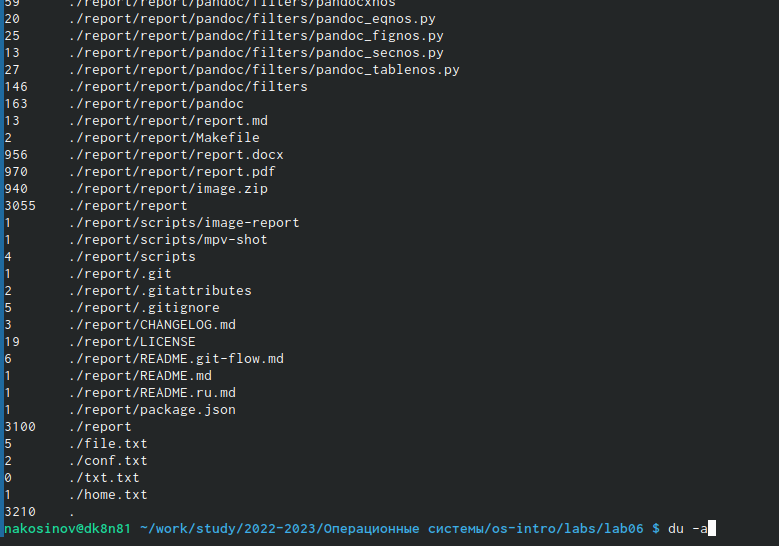


Рис. 27: Использование команды du

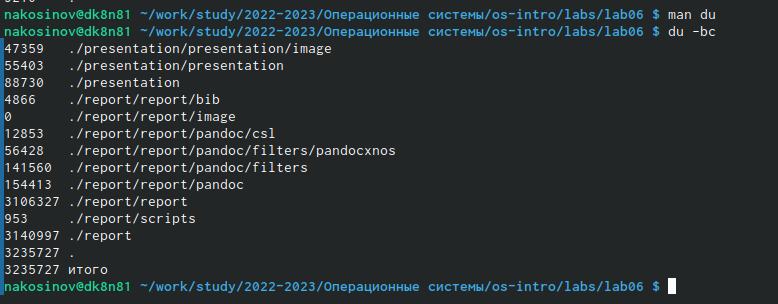


Рис. 28: Использование команды du

1. Прочтя мануал команды *find*, и применив модификатор *type -d*, выведем все каталоги, лежащие в домашней директории.

Чтение документпции команды

Рис. 29: Чтение документпции команды

Вызов find с модификатором

Рис. 30: Вызов find с модификатором

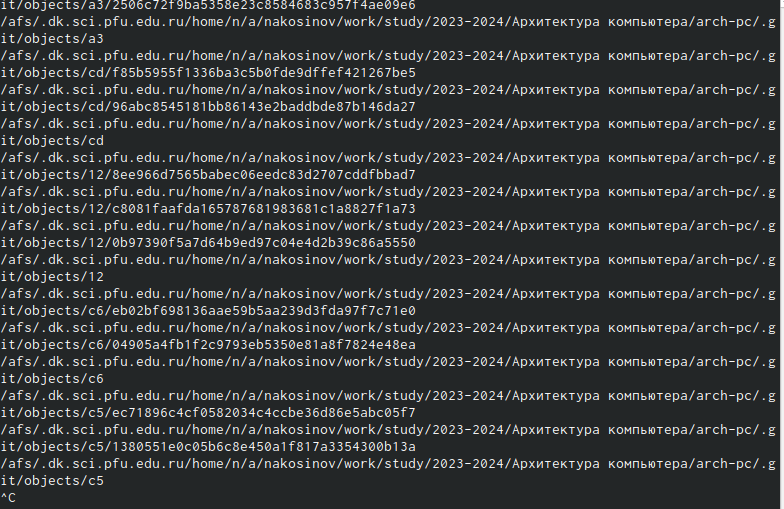


Рис. 31: Результат: каталоги домашней директории

# 7 Выводы

Терминал предоставляет пользователю возможность получение информации о состоянии файловой системы: поиск файлов, перенаправление вывода полученной информации, управление процессами и т.п.