Лабораторная работа №7

Дисциплина: Операционные системы

Коновалова Татьяна Борисовна

Содержание

# Цель работы

Цель данной лабораторной работы — Ознакомление с инструментами поиска файлов и фильтрации текстовых данных. Приобретение практических навыков: по управлению процессами (и задания-ми), по проверке использования диска и обслуживанию файловых систем.

# Задание

1. Сделать отчёт по лабораторной работе №7 в формате Markdown.
2. Ознакомление с инструментами поиска файлов и фильтрации текстовых данных.

# Выполнение лабораторной работы

1). Осуществила вход в систему, используя соответствующее имя пользователя.

2). Записала, в файл file.txt название файлов, содержащийхся в определённом каталоге. Для того, чтобы записать в файл file.txt названия файлов, содержащихся в каталоге /etc, использовала команду «ls–a/etc> file.txt». Далее с помощью команды «ls-a~ >> file.txt» дописываю в этот же файл названия файлов, содержащихся в домашнем каталоге. Командой «catfile.txt» просматриваю файл, чтобы убедиться в правильности действий (алгоритм действий представлен на рис. 1 , 2).

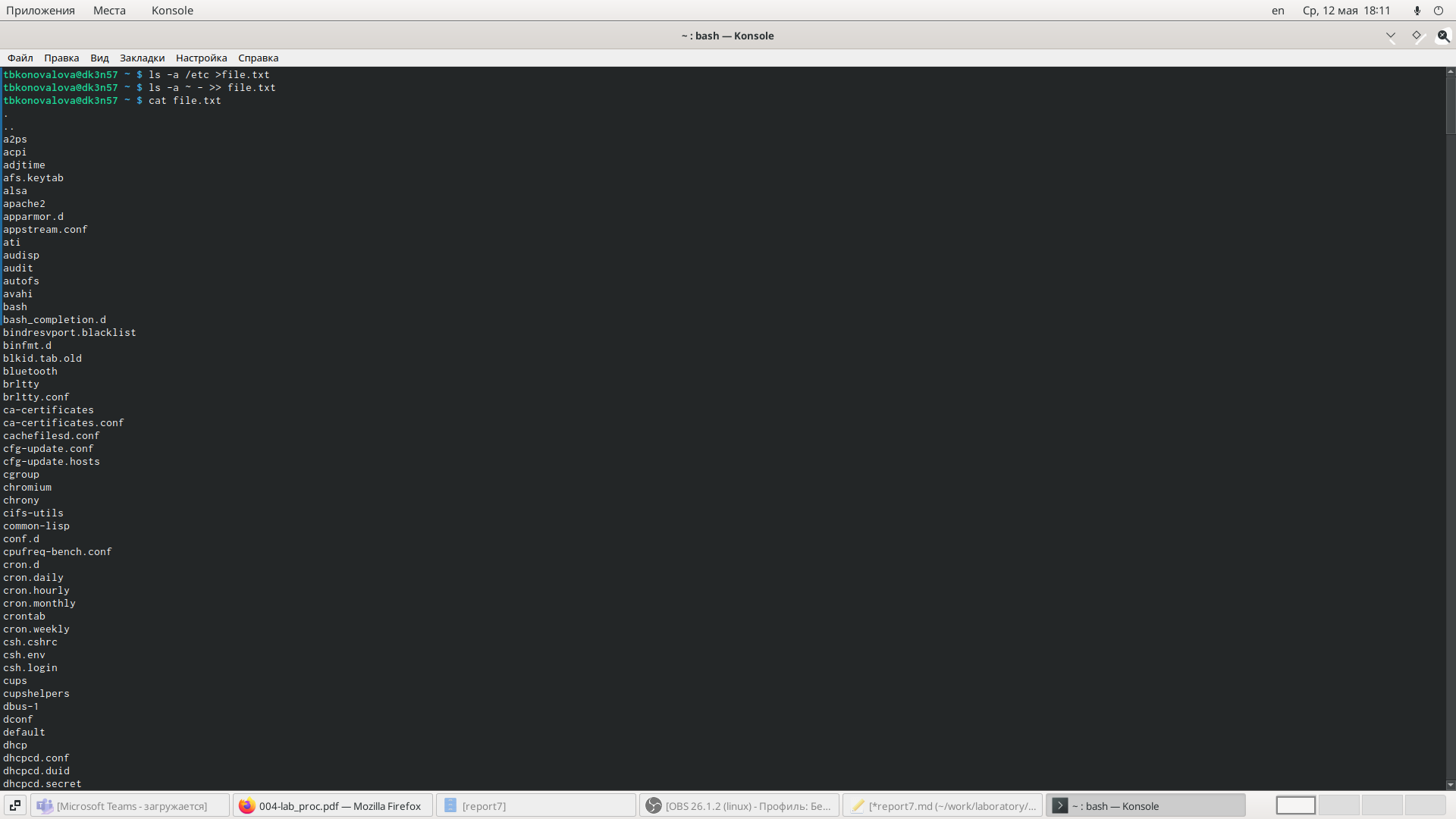


Figure 1: Запись файлов

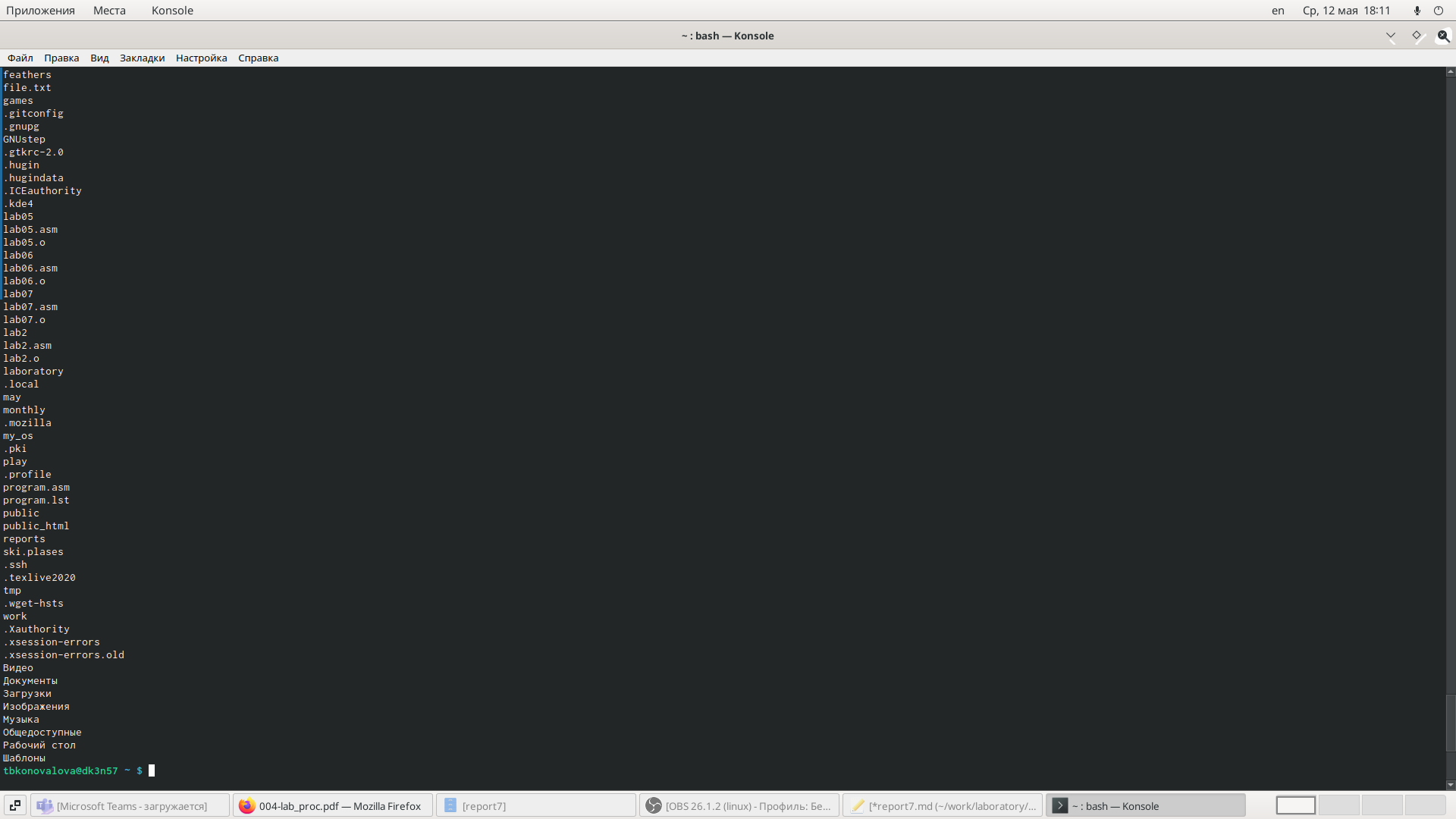


Figure 2: Файлы в файле file.txt

3). Вывела имена всех файлов из file.txt, имеющих расширение .conf и записала их в новый текстовой файл conf.txt, используя команду «grep-e‘.conf$’file.txt> conf.txt». С помощью команды «catconf.txt» проверяю правильность выполненных действий (иллюстрация на рис. 3 , 4).

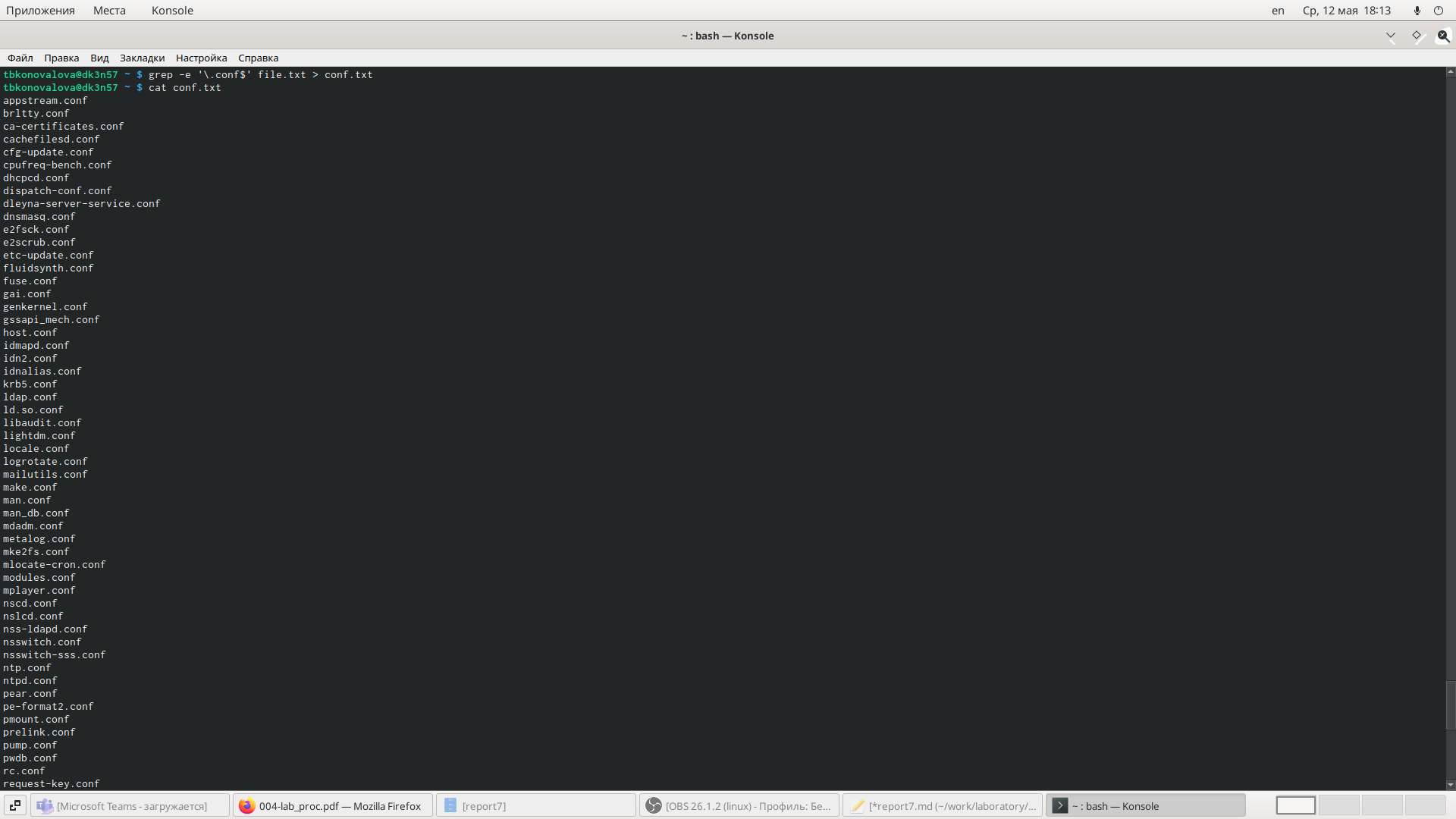


Figure 3: Вывожу имена файлов

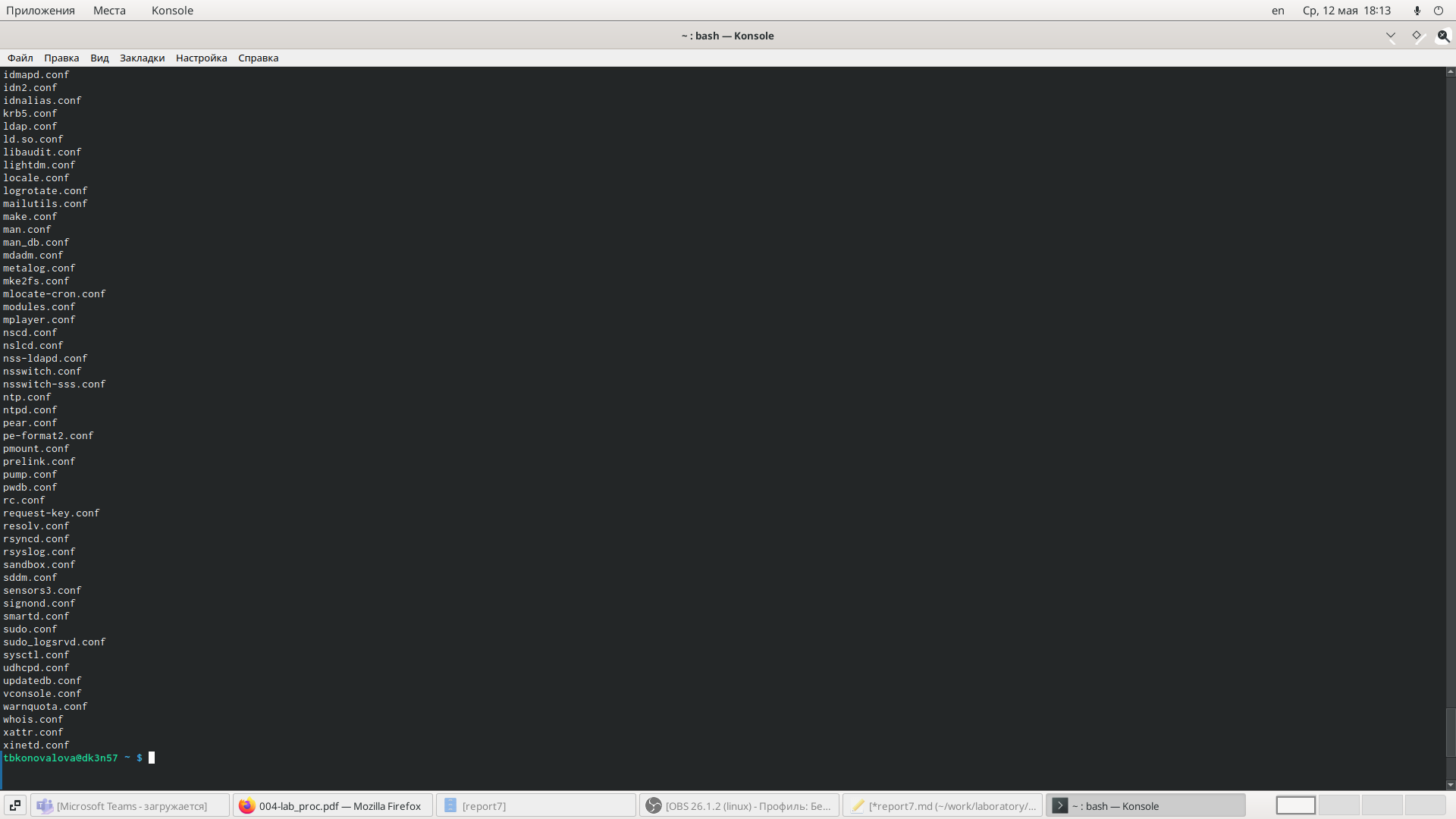


Figure 4: Файлы с расширением .conf

4). Определить, какие файлы в домашнем каталоге имеют имена, начинающиеся с символа c, можно несколькими командами: - «find~ -maxdepth1 -name“c*” -print» (опция maxdepth1 необходима для того, чтобы файлы находились строго только в домашнем каталоге); - «ls~/c*»; - «ls–a~ | grepc\*» (см. рис. 5).

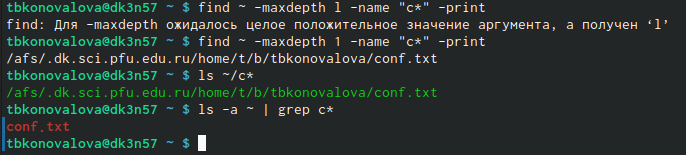


Figure 5: Создание файлов

5). Чтобы вывестина экран (постранично) имена файлов из каталога /etc, начинающиеся с символа h, я использовала команду «find /etc –maxdepth1 –name “h\*” | less» (Скриншоты 6 , 7).

Figure 6: команда «find /etc –maxdepth1 –name “h*” | less»

Figure 6: команда «find /etc –maxdepth1 –name “h\*” | less»

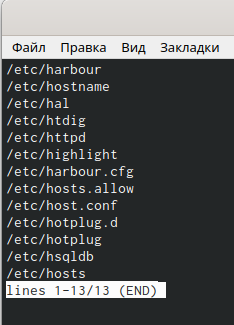


Figure 7: список файлов

6). Запускаю в фоновом режиме процесс, который будет записывать в файл ~/logfile файлы, имена которых начинаются с log, используя команду «find/ -name“log\*” > logfile&» (Скриншоты 8 , 9). Так как в фоновом режиме у меня запустился беспрерывный процесс записывания файла, я сделала скриншоты некоторых частей рабоыт процесса. Командой «catlogfile» проверяю выполненные действия (Рисунок 10). Далее удалила файл ~/logfile с помощью команды «rm logfile», у меня данный файл уже был удалён, поэтому программа вывела сообщение об этом (Рисунок 11).

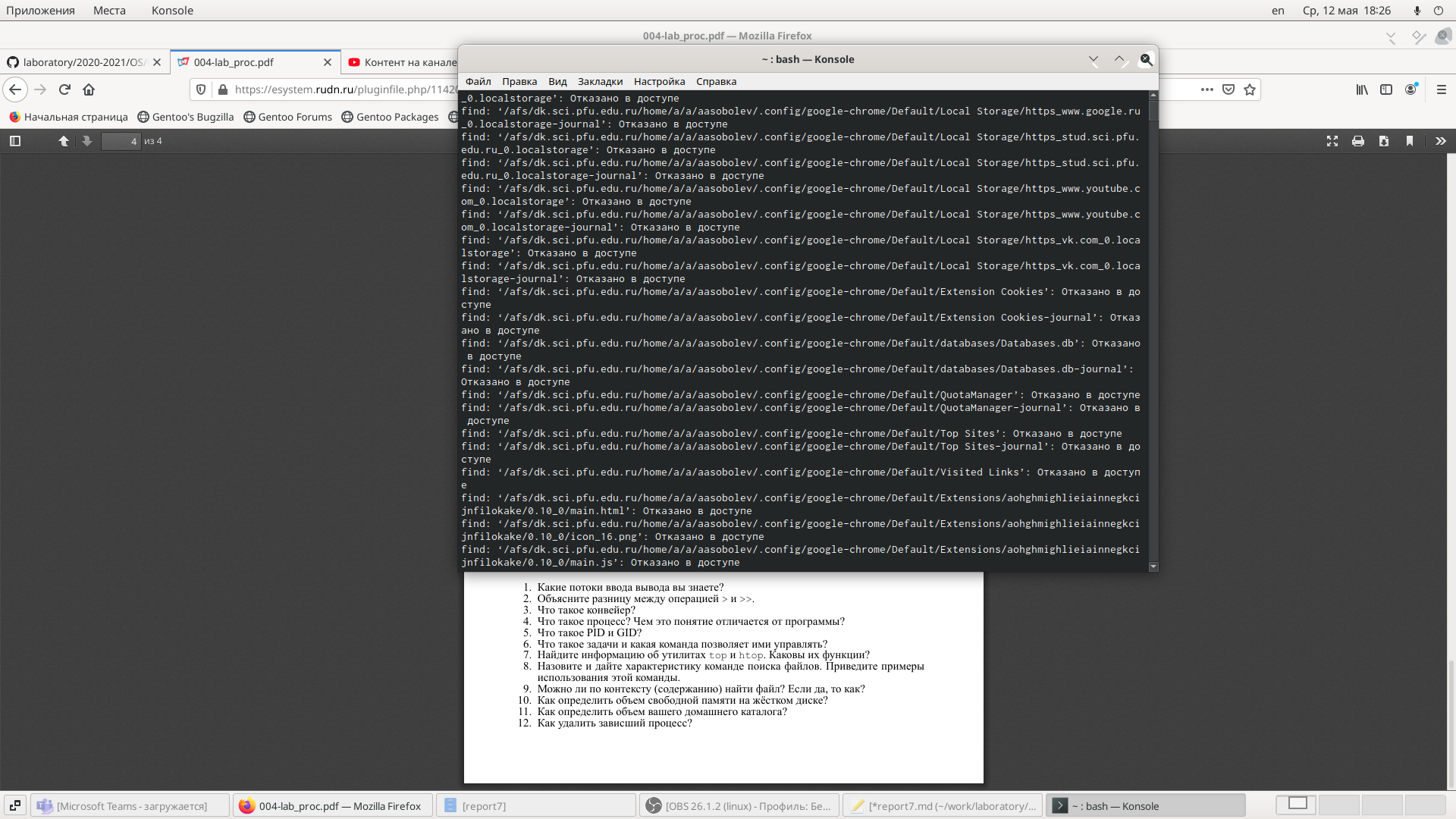


Figure 8: Фоновый режим процесса

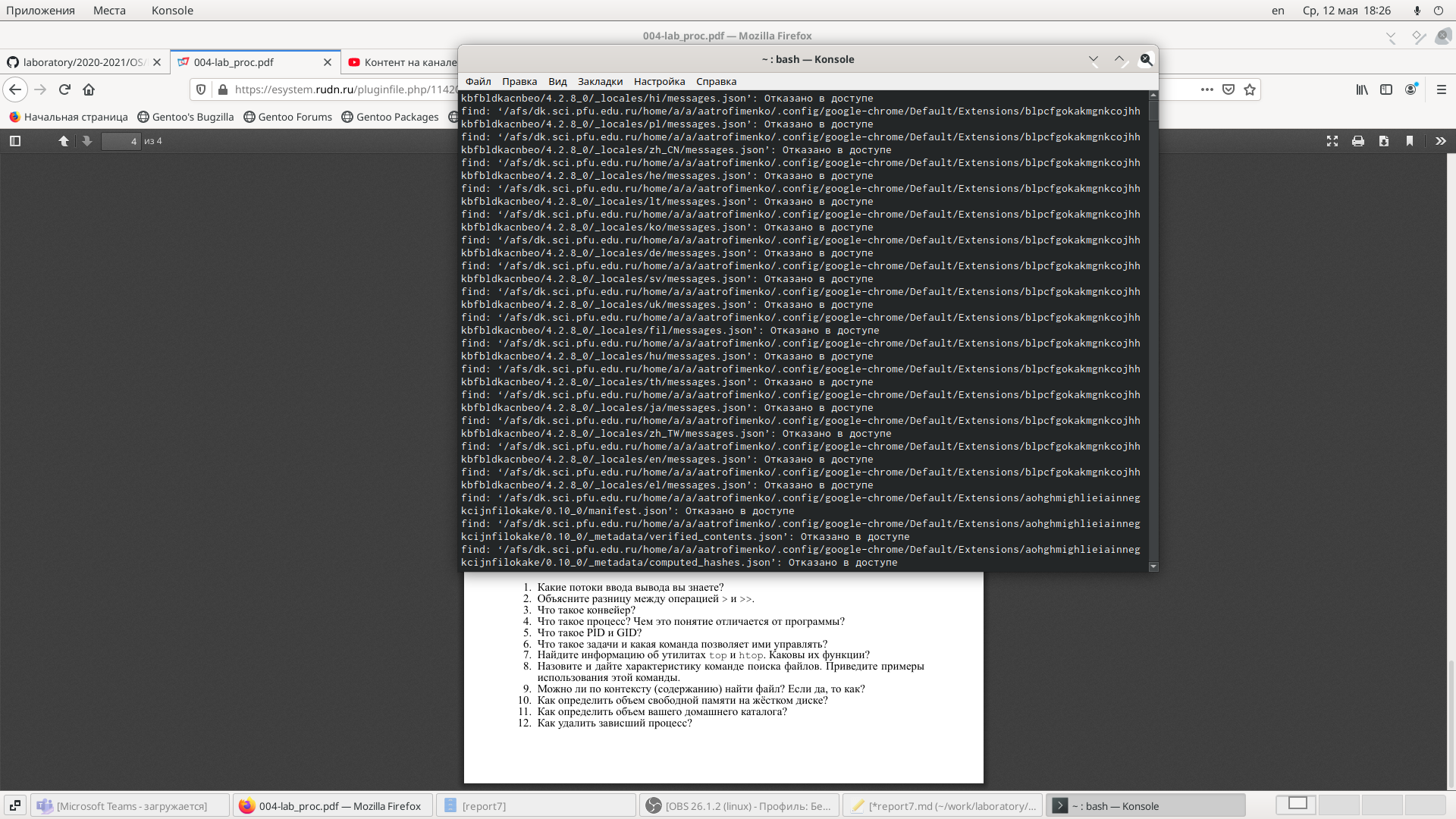


Figure 9: Фоновый режим процесса

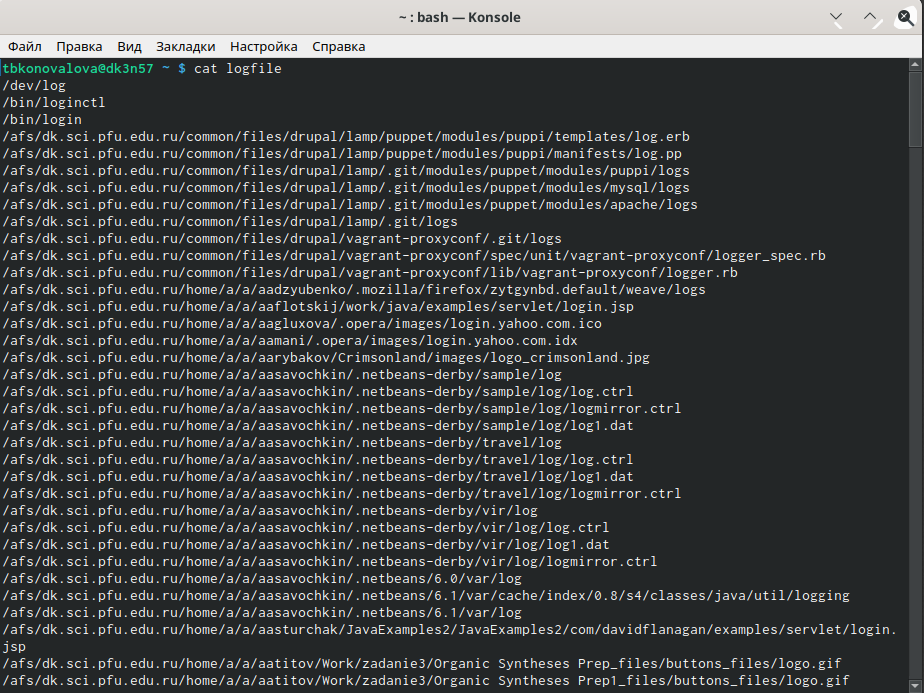


Figure 10: Удаление файла

Figure 11: Удаление файла с помощью команды rm

Figure 11: Удаление файла с помощью команды rm

7). Смотрите пункт 6.

8). Запускаю редактор gedit в фоновом режиме командой «gedit&» (Рисунок 12). После этого на экране появляется окно редактора.

Figure 12: Редактор gedit

Figure 12: Редактор gedit

9). Чтобы определить идентификатор процесса gedit, использую команду «ps| grep-i“gedit”» (Скриншот 13). Из рисунка видно, что наш процесс имеет PID 24249. Узнать идентификатор процесса можно также, используя команду «pgrep gedit»или «pidof gedit».

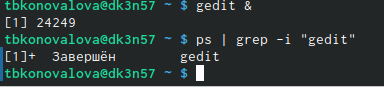


Figure 13: команда «ps| grep-i“gedit”»

10). Прочитав информацию о команде kill с помощью команды «man kill», использую её для завершения процесса gedit (команда «kill 24249») (Алгоритм действий представлен на рис. 14 , 15 , 16).

Figure 14: команда kill

Figure 14: команда kill

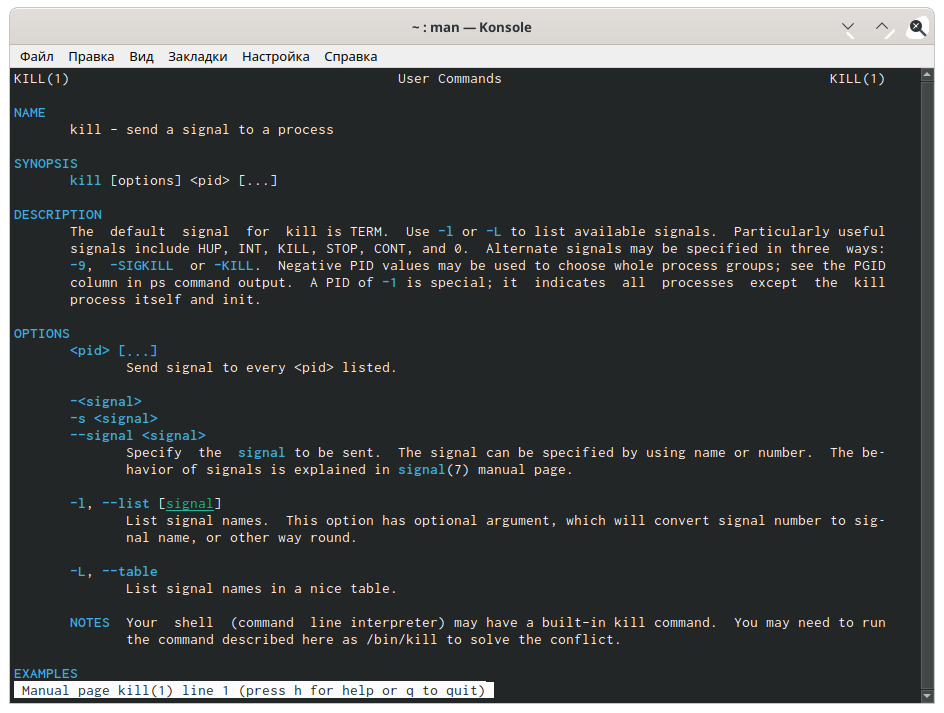


Figure 15: информация по команде kill

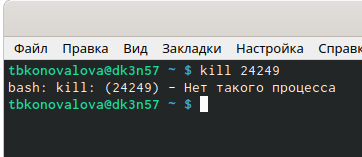


Figure 16: завершение процесса gedit

11). C помощью команд «mandf» и «mandu» узнаю информацию по необходимым командам и далее использую их (Рисунки 17 , 18 , 19 , 20 , 21). df – утилита, показывающая список всех файловых систем по именам устройств, сообщает их размер, занятое и свободное пространство и точки монтирования. Синтаксис: df[опции] устройств. du – утилита, предназначенная для вывода информации об объеме дискового пространства, занятого файлами и директориями. Она принимает путь к элементу файловой системы и выводит информацию о количестве байт дискового пространства или блоков диска, задействованных для его хранения. Синтаксис: du [опции] каталог\_или\_файл

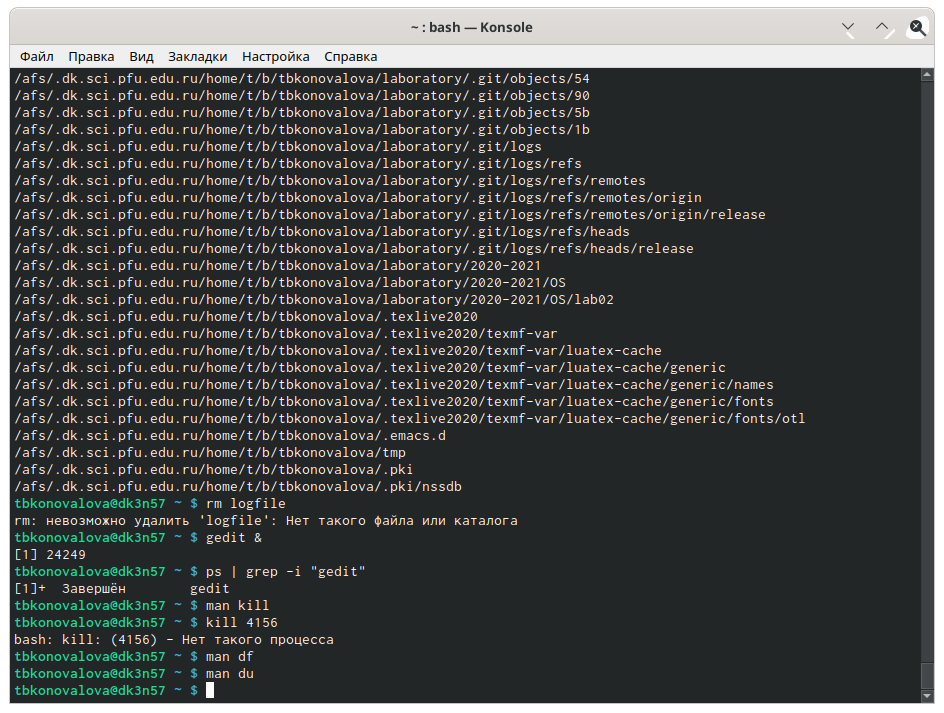


Figure 17: команды df, du

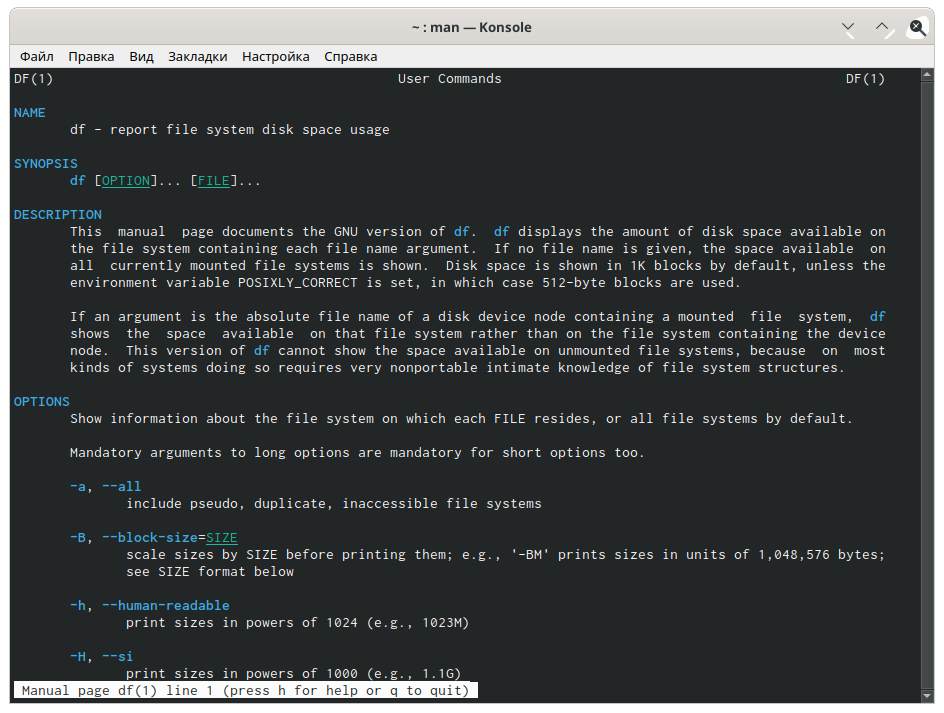


Figure 18: информация команды df

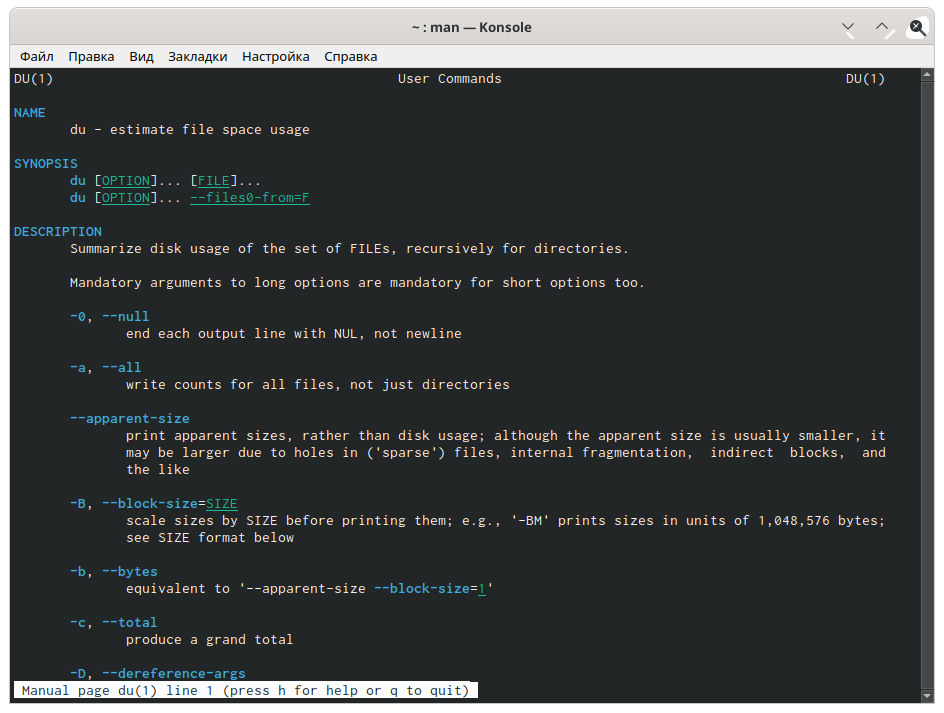


Figure 19: информация команды du

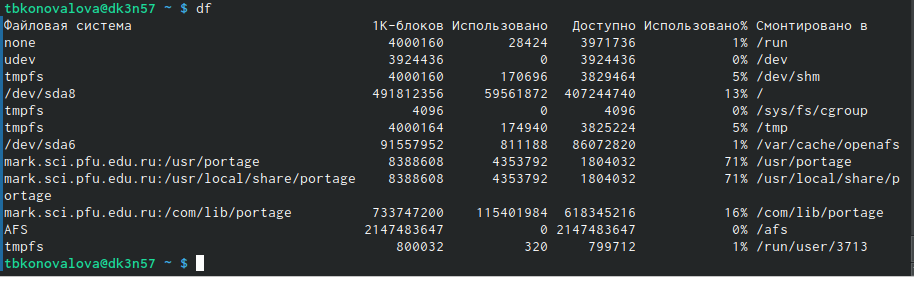


Figure 20: команда df в консоли

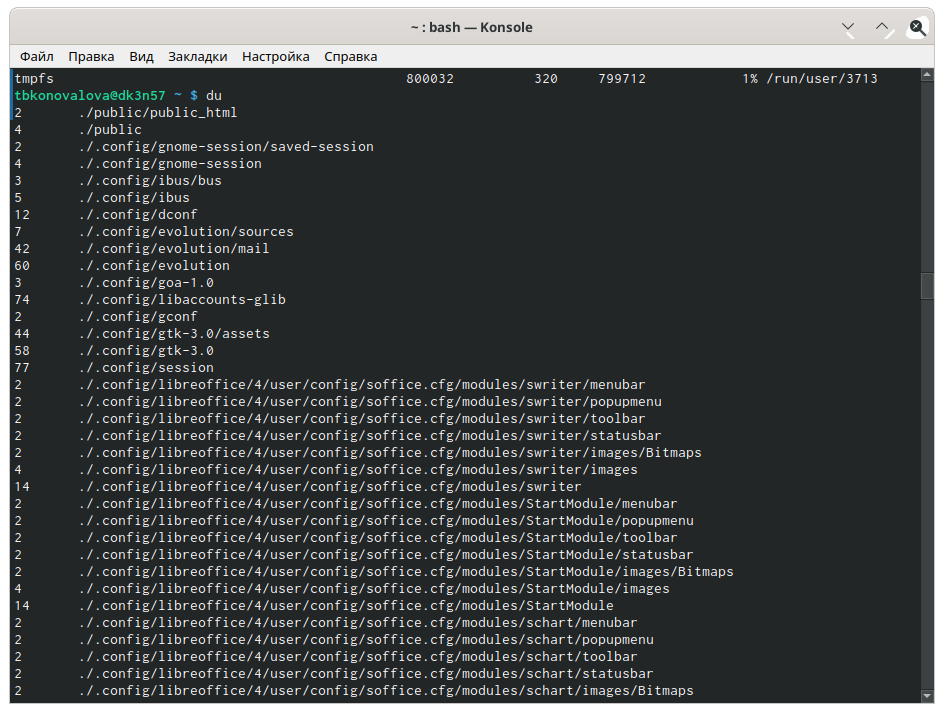


Figure 21: команда du в консоли

12). Вывела имена всех директорий, имеющихся в моем домашнем каталоге с помощью команды «find~ -typed», предварительно получив информацию с помощью команды «man find» (Скриншоты 22 , 23).

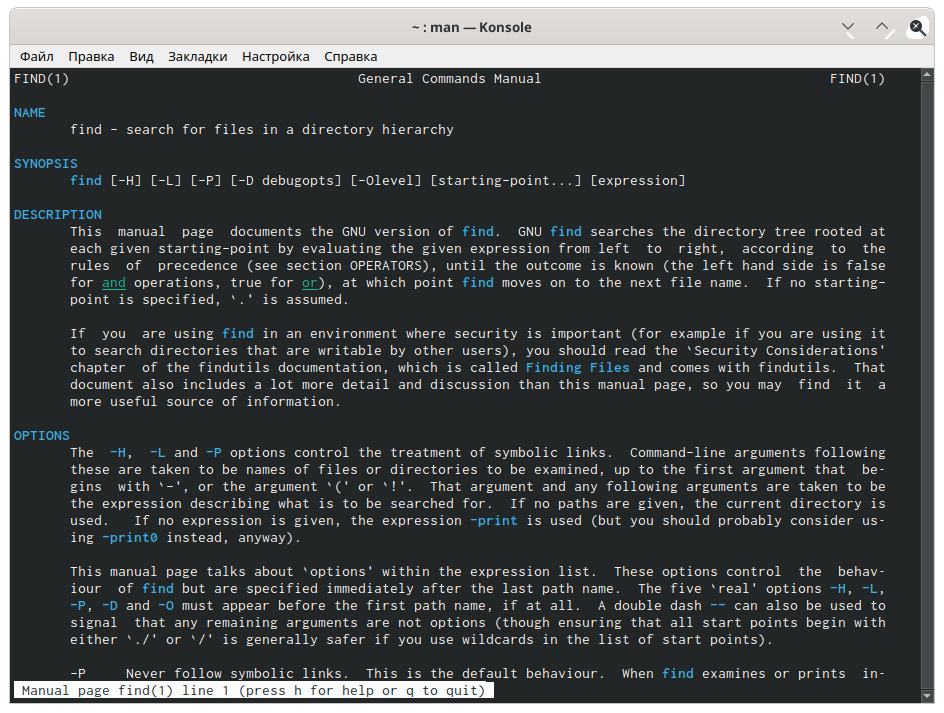


Figure 22: Информация конмады find

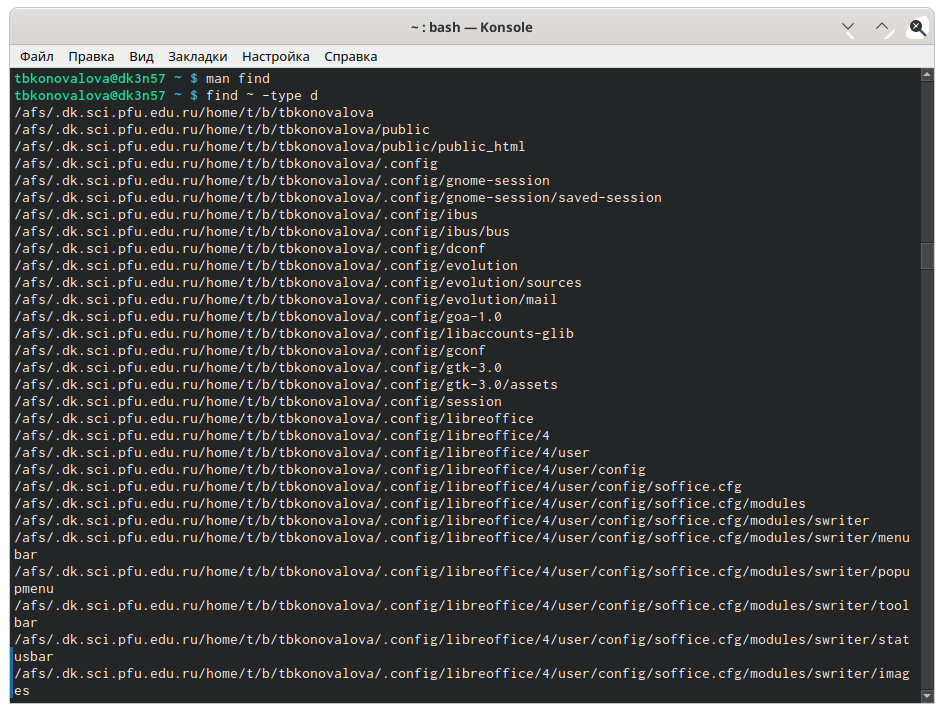


Figure 23: имена всех директорий

Ответы на контрольные вопросы:

1). В системе по умолчанию открыто три специальных потока:

–stdin − стандартный поток ввода (по умолчанию: клавиатура), файловый дескриптор 0;

–stdout − стандартный поток вывода (по умолчанию: консоль), файловый дескриптор 1;

-stderr − стандартный поток вывод сообщений об ошибках (поумолчанию: консоль), файловый дескриптор 2.

Большинство используемых в консоли команд и программ записывают результаты своей работы в стандартный поток вывода stdout.

2). ‘>’ Перенаправление вывода в файл

‘»’ Перенаправление вывода в файл и открытие файла в режиме добавления (данные добавляются в конец файла)/

3). Конвейер (pipe) служит для объединения простых команд или утилит в цепочки, в которых результат работы предыдущей команды передаётся последующей.

Синтаксис следующий:

команда1|команда2 (это означает, что вывод команды 1 передастся на ввод команде 2)

4). Процесс рассматривается операционной системой как заявка на потребление всех видов ресурсов, кроме одного − процессорного времени. Этот последний важнейший ресурс распределяется операционной системой между другими единицами работы − потоками, которые и получили свое название благодаря тому, что они представляют собой последовательности (потоки выполнения) команд.

Процесс − это выполнение программы. Он считается активной сущностью и реализует действия, указанные в программе.

Программа представляет собой статический набор команд, а процесс это набор ресурсов и данных, использующихся при выполнении программы.

5). pid: идентификатор процесса (PID) процесса (processID), к которому вызывают метод

gid: идентификатор группы UNIX, в котором работает программа.

6). Любую выполняющуюся в консоли команду или внешнюю программу можно запустить в фоновом режиме. Для этого следует в конце имени команды указать знак амперсанда &.

Запущенные фоном программы называются задачами (jobs). Ими можно управлять с помощью команды jobs, которая выводит список запущенных в данный момент задач.

7). top − это консольная программа, которая показывает список работающих процессов в системе. Программа в реальном времени отсортирует запущенные процессы по их нагрузке на процессор.

htop − это продвинутый консольный мониторинг процессов. Утилита выводит постоянно меняющийся список системных процессов, который сортируется в зависимости от нагрузки на ЦПУ. Если делать сравнение сtop, то htop показывает абсолютно все процессы в системе, время их непрерывного использования, загрузку процессоров и расход оперативной памяти.

8). find − это команда для поиска файлов и каталогов на основе специальных условий. Ее можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и другим подобным критериям.

Команда find имеет такой синтаксис:

find[папка][параметры] критерий шаблон [действие]

Папка − каталог в котором будем искать

Параметры − дополнительные параметры, например, глубина поиска, и т д.

Критерий − по какому критерию будем искать: имя, дата создания, права, владелец и т д.

Шаблон – непосредственно значение по которому будем отбирать файлы.

Основные параметры:

-P никогда не открывать символические ссылки

-L - получает информацию о файлах по символическим ссылкам. Важно для дальнейшей обработки, чтобы обрабатывалась не ссылка, а сам файл.

-maxdepth - максимальная глубина поиска по подкаталогам,для поиска только в текущем каталоге установите 1.

-depth - искать сначала в текущем каталоге, а потом в подкаталогах

-mount искать файлы только в этой файловой системе.

-version - показать версию утилиты find

-print - выводить полные имена файлов

-typef - искать только файлы

-typed - поиск папки в Linux

Основные критерии:

-name - поиск файлов по имени

-perm - поиск файлов в Linux по режиму доступа

-user - поиск файлов по владельцу

-group - поиск по группе

-mtime - поиск по времени модификации файла

-atime - поиск файлов по дате последнего чтения

-nogroup - поиск файлов, не принадлежащих ни одной группе

-nouser - поиск файлов без владельцев

-newer - найти файлы новее чем указанный

-size - поиск файлов в Linux по их размеру

Примеры:

find~ -type d поиск директорий в домашнем каталоге

find~ -type f -name ".\*" поиск скрытых файлов в домашнем каталоге

9). Файл по его содержимому можно найти с помощью команды grep: «grep -r" слово/выражение, которое нужно найти"».

10). Утилита df, позволяет проанализировать свободное пространство на всех подключенных к системе разделах.

11). При выполнении команды du (без указания папки и опции) можно получить все файлы и папки текущей директории с их размерами. Для домашнего каталога: du ~/

12). Основные сигналы (каждый сигнал имеет свой номер), которые используются для завершения процесса:

* SIGINT–самый безобидный сигнал завершения, означает Interrupt. Он отправляется процессу, запущенному из терминала с помощью сочетания клавиш Ctrl+C. Процесс правильно завершает все свои действия и возвращает управление;
* SIGQUIT–это еще один сигнал, который отправляется с помощью сочетания клавиш, программе, запущенной в терминале. Он сообщает ей что нужно завершиться и программа может выполнить корректное завершение или проигнорировать сигнал. В отличие от предыдущего, она генерирует дамп памяти. Сочетание клавиш Ctrl+/;
* SIGHUP–сообщает процессу, что соединение с управляющим терминалом разорвано, отправляется, в основном, системой при разрыве соединения с интернетом;
* SIGTERM–немедленно завершает процесс, но обрабатывается программой, поэтому позволяет ей завершить дочерние процессы и освободить все ресурсы;
* SIGKILL–тоже немедленно завершает процесс, но, в отличие от предыдущего варианта, он не передается самому процессу, а обрабатывается ядром. Поэтому ресурсы и дочерние процессы остаются запущенными.

Также для передачи сигналов процессам в Linux используется утилита kill, её синтаксис: kill [-сигнал] [pid\_процесса] (PID – уникальный идентификатор процесса). Сигнал представляет собой один из выше перечисленных сигналов для завершения процесса.

Перед тем, как выполнить остановку процесса, нужно определить его PID. Для этого используют команды ps и grep. Команда ps предназначена для вывода списка активных процессов в системе и информации о них. Команда grep запускается одновременно с ps (вканале) и будет выполнять поиск по результатам команды ps.

Утилита pkill – это оболочка для kill, она ведет себя точно так же, и имеет тот же синтаксис, только в качестве идентификатора процесса ей нужно передать ег оимя.

killall работает аналогично двум предыдущим утилитам. Она тоже принимает имя процесса в качестве параметра и ищет его PID в директории /proc. Но эта утилита обнаружит все процессы с таким именем и завершит их.

# Выводы

В ходе выполнения данной лабораторной работы я изучила инструменты поиска файлови фильтрации текстовых данных, а также приобрела практические навыки: по управлению процессами (и заданиями), по проверке использования диска и обслуживанию файловых систем.