

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3

по Операционной системе Linux

Планирование процессов

Студент

Лобов М.Ю.

Группа АИ-18

Руководитель

Кургасов В.В.

Доцент, к.п.н.

Липецк 2020 г.

Задание кафедры

1. Повторить команды `cat`, `head`, `tail`, `more`, `less`, `grep`, `find`.
2. Разобраться с понятиями конвейер, перенаправление ввода-вывода.
3. Ознакомиться с информацией из рекомендованных источников и других про конвейеризации.
4. Повторить назначение прав доступа. Команды `chmod`, `chown`.
5. Ознакомиться с информацией по теме процессы, посмотреть и опробовать примеры наиболее распространенных команд, изучить возможность запуска процессов в `supervisor`.
6. Изучить возможность автоматического запуска программ по расписанию.

Ход работы

Для начала разберёмся с командой `cat`.

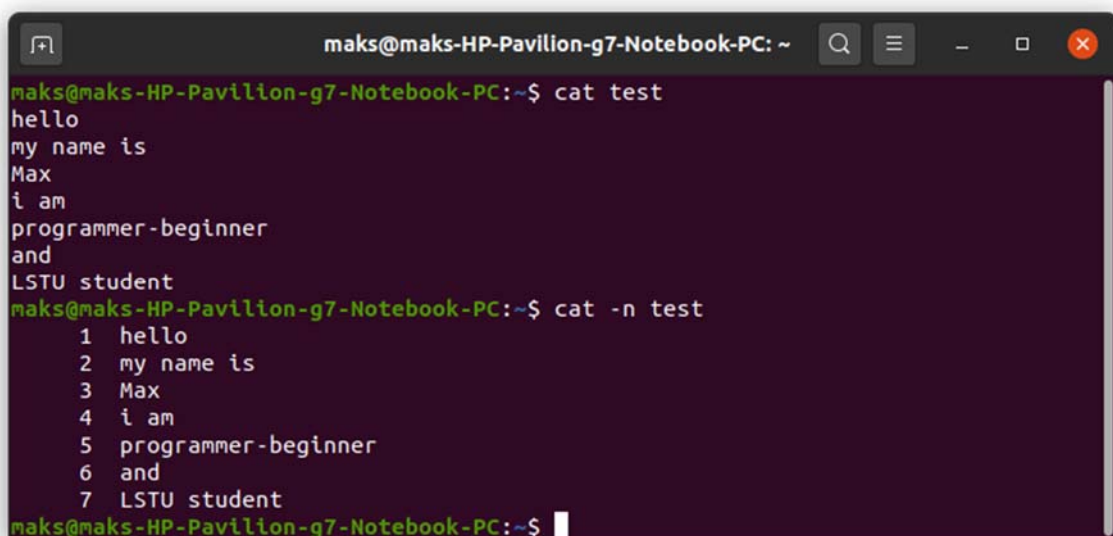
Задача команды `cat` очень проста – она читает данные из файла или стандартного ввода и выводит их на экран. Это все, чем занимается утилита. Но с помощью ее опций и операторов перенаправления вывода можно сделать очень многое. Сначала рассмотрим синтаксис утилиты:

\$ cat опции файл1 файл2 ...

Вы можете передать утилите несколько файлов и тогда их содержимое будет выведено поочередно, без разделителей. Опции позволяют очень сильно видоизменить вывод и сделать именно то, что вам нужно. Рассмотрим основные опции:

- **-b** - нумеровать только непустые строки;
- **-E** - показывать символ \$ в конце каждой строки;
- **-n** - нумеровать все строки;
- **-s** - удалять пустые повторяющиеся строки;
- **-T** - отображать табуляции в виде ^I;
- **-h** - отобразить справку;
- **-v** - версия утилиты.

Пример использования утилиты `cat`:



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ cat test  
hello  
my name is  
Max  
i am  
programmer-beginner  
and  
LSTU student  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ cat -n test  
1 hello  
2 my name is  
3 Max  
4 i am  
5 programmer-beginner  
6 and  
7 LSTU student  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 1 – Результат использования команды `cat`

Далее перейдём к команде head.

Команда head выводит начальные строки (по умолчанию — 10) из одного или нескольких документов. Также она может показывать данные, которые передает на вывод другая утилита.

Синтаксис у команды head следующий:

\$ head опции файл

Здесь:

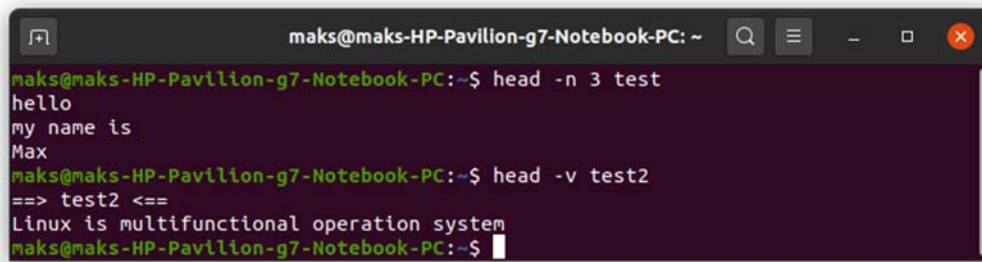
- **Опции** — это параметр, который позволяет настраивать работу команды таким образом, чтобы результат соответствовал конкретным потребностям пользователя.
- **Файл** — это имя документа (или имена документов, если их несколько). Если это значение не задано либо вместо него стоит знак «-», команда будет брать данные из стандартного вывода.

Чаще всего к команде head применяются такие опции:

- **-c (--bytes)** — позволяет задавать количество текста не в строках, а в байтах. При записи в виде --bytes=[-]NUM выводит на экран все содержимое файла, кроме NUM байт, расположенных в конце документа.
- **-n (--lines)** — показывает заданное количество строк вместо 10, которые выводятся по умолчанию. Если записать эту опцию в виде --lines=[-]NUM, будет показан весь текст кроме последних NUM строк.
- **-q (--quiet, --silent)** — выводит только текст, не добавляя к нему название файла.
- **-v (--verbose)** — перед текстом выводит название файла.
- **-z (--zero-terminated)** — символы перехода на новую строку заменяет символами завершения строк.

Переменная NUM, упомянутая выше — это любое число от 0 до бесконечности, задаваемое пользователем. Оно может быть обычным либо содержать в себе множитель.

Пример использования команды head:

A terminal window titled 'maks@maks-HP-Pavilion-g7-Notebook-PC: ~' showing two commands and their outputs. The first command is 'head -n 3 test', which outputs 'hello', 'my name is', and 'Max'. The second command is 'head -v test2', which outputs '==> test2 <==', 'Linux is multifunctional operation system', and a prompt character.

```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ head -n 3 test  
hello  
my name is  
Max  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ head -v test2  
==> test2 <==  
Linux is multifunctional operation system  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 2 – Результат выполнения команды head

Разберёмся с командой tail.

Команда tail позволяет выводить заданное количество строк с конца файла, а также выводить новые строки в интерактивном режиме.

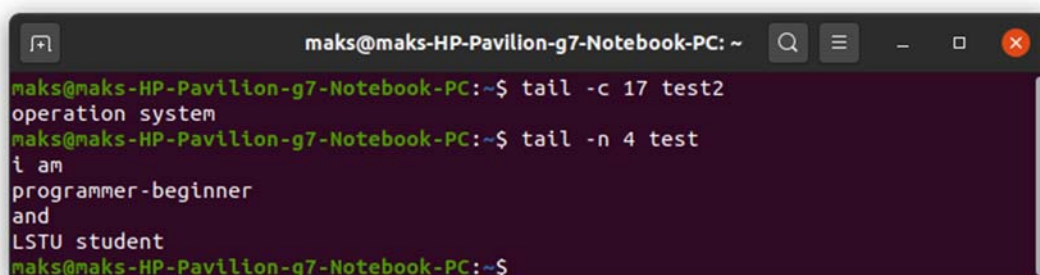
Синтаксис команды очень прост:

\$ tail опции файл

По умолчанию утилита выводит десять последних строк из файла, но ее поведение можно настроить с помощью опций:

- **-c** - выводить указанное количество байт с конца файла;
- **-f** - обновлять информацию по мере появления новых строк в файле;
- **-n** - выводить указанное количество строк из конца файла;
- **--pid** - используется с опцией **-f**, позволяет завершить работу утилиты, когда завершится указанный процесс;
- **-q** - не выводить имена файлов;
- **--retry** - повторять попытки открыть файл, если он недоступен;
- **-v** - выводить подробную информацию о файле;

Пример использования команды tail:

A terminal window titled 'maks@maks-HP-Pavilion-g7-Notebook-PC: ~' showing two commands and their outputs. The first command is 'tail -c 17 test2', which outputs 'operation system'. The second command is 'tail -n 4 test', which outputs 'i am', 'programmer-beginner', 'and', and 'LSTU student'.

```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ tail -c 17 test2  
operation system  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ tail -n 4 test  
i am  
programmer-beginner  
and  
LSTU student  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 3 – Результат выполнения команды tail

Теперь поговорим о команде `more`.

Утилита `more` предназначена для постраничного просмотра файлов в терминале Linux. Своим названием она обязана надписи `more` (в русскоязычном варианте – дальше), появляющейся внизу каждой страницы.

Команда `more` – одна из самых примитивных команд для работы с текстом.

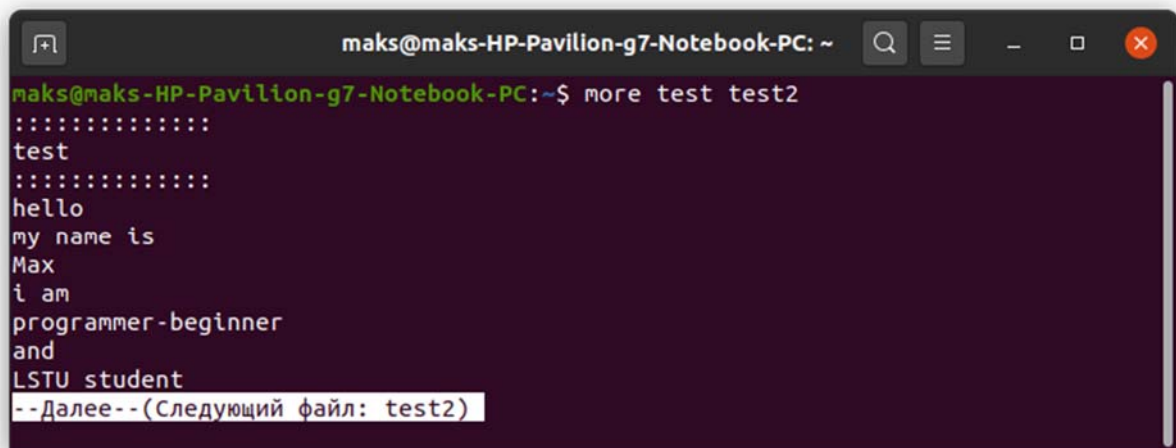
В эмуляторе терминала Linux команда записывается так:

\$ more опции файл

Список опций команды:

- **-d** — вывод информации в конце страницы о клавишах, используемых для продолжения работы, завершения её или получения инструкций;
- **-l** — игнорирование в тексте символа разрыва страницы;
- **-f** — подсчёт числа логических строк вместо экранных;
- **-p** — очистка экрана терминала для того, чтобы пользователю не пришлось пользоваться прокруткой перед выводом следующей порции текста;
- **-c** — устранение потребности в прокрутке (как и **-p**) — отображение текста, начиная с верха экрана, и стирание при этом предыдущего вывода построчно;
- **-s** — замена нескольких пустых строк, расположенных подряд, одной пустой строкой;
- **-u** — удаление подчёркивания;
- **-n** — отображение n-го количества строк;
- **+n** — отображение текста, начиная со строки с номером n;
- **+/строка** — поиск в файле указанной строки и начало вывода текста именно с неё;
- **--help** — вызов справки;
- **-v (--version)** — вывод на экран текущей версии утилиты.

Пример использования команды more:



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~$ more test test2
::::::::::::
test
::::::::::::
hello
my name is
Max
i am
programmer-beginner
and
LSTU student
--Далее--(Следующий файл: test2)
```

Рисунок 4 – Результат открытия двух файлов командой more

Похожей на more является команда less. Поговорим о ней.

Особенность less заключается в том, что команда не считывает текст полностью, а загружает его небольшими фрагментами.

Запись команды less в терминале выглядит так:

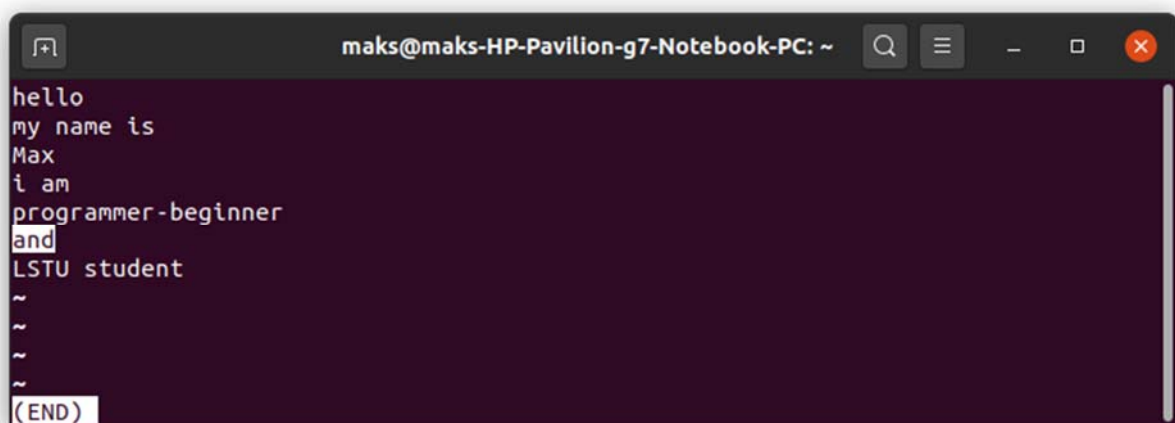
команда опции файл

Наиболее популярные опции:

- **-a, --search-skip-screen** — не осуществлять поиск в тексте, который в данный момент отображен на экране;
- **-bn, --buffers=n** — задать размер буфера памяти;
- **-c, --clear-screen** — листать текст, полностью стирая содержимое экрана (построчная прокрутка работать не будет);
- **-Dxcolor, --color=xcolor** — задать цвет отображаемого текста;
- **-E, --QUIT-AT-EOF** — выйти, когда утилита достигнет конца файла;
- **-e, --quit-at-eof** — выйти, когда утилита второй раз достигнет конца файла;
- **-F, --quit-if-one-screen** — выйти, если содержимое файла помещается на одном экране;
- **-f, --force** — открыть специальный файл;

- **-hn, --max-back-scroll=n** — задать максимальное количество строк для прокрутки назад;
- **-yn, --max-forw-scroll=n** — задать максимальное количество строк для прокрутки вперёд;
- **-i, --ignore-case** — игнорировать регистр;
- **-I, --IGNORE-CASE** — игнорировать регистр, даже если паттерн для поиска содержит заглавные буквы;
- **-jn, --jump-target=n** — указать, в какой строке должна быть выведена искомая информация;
- **-J, --status-column** — пометить строки, соответствующие результатам поиска;
- **-n, --line-numbers** — не выводить номера строк;
- **-N, --LINE-NUMBERS** — вывести номера строк;
- **-s, --squeeze-blank-lines** — заменить множество идущих подряд пустых строк одной пустой строкой;
- **-w, --hilitе-unread** — выделить первую строку нового фрагмента текста.

Пример использования команды less:



```

maks@maks-HP-Pavilion-g7-Notebook-PC: ~
hello
my name is
Max
i am
programmer-beginner
and
LSTU student
~
~
~
~
(END)

```

Рисунок 5 – Результат открытия файла командой less

Теперь займёмся командой grep.

С помощью команды grep можно искать не только строки в файлах, но и фильтровать вывод команд, и много чего ещё.

Утилита `grep` решает множество задач, в основном она используется для поиска строк, соответствующих строке в тексте или содержимому файлов. Также она может находить по шаблону или регулярным выражениям. Команда в считанные секунды найдёт файл с нужной строчкой, текст в файле или отфильтрует из вывода только пару нужных строк.

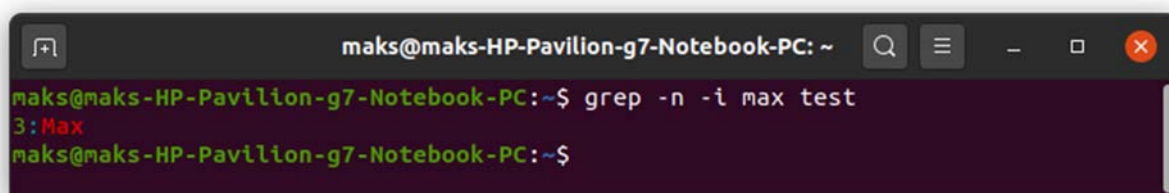
Синтаксис команды выглядит следующим образом:

\$ `grep` [опции] шаблон [имя файла...]

Давайте рассмотрим самые основные опции утилиты, которые помогут более эффективно выполнять поиск текста в файлах **`grep`**:

- **-b** - показывать номер блока перед строкой;
- **-c** - подсчитать количество вхождений шаблона;
- **-h** - не выводить имя файла в результатах поиска внутри файлов Linux;
- **-i** - не учитывать регистр;
- **-l** - отобразить только имена файлов, в которых найден шаблон;
- **-n** - показывать номер строки в файле;
- **-s** - не показывать сообщения об ошибках;
- **-v** - инвертировать поиск, выдавать все строки кроме тех, что содержат шаблон;
- **-w** - искать шаблон как слово, окружённое пробелами;
- **-e** - использовать регулярные выражения при поиске;
- **-An** - показать вхождение и n строк до него;
- **-Bn** - показать вхождение и n строк после него;
- **-Cn** - показать n строк до и после вхождения;

Пример использования команды `grep`:



```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ grep -n -i max test  
3:Max  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 6 – Результат поиска слова `max` в файле

Теперь перейдём к команде `find`.

`Find` - это одна из наиболее важных и часто используемых утилит системы Linux. 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.

Утилита `find` предустановлена по умолчанию во всех Linux дистрибутивах, поэтому вам не нужно будет устанавливать никаких дополнительных пакетов. Это очень важная находка для тех, кто хочет использовать командную строку наиболее эффективно.

Команда `find` имеет такой синтаксис:

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

Рассмотрим самые полезные параметры:

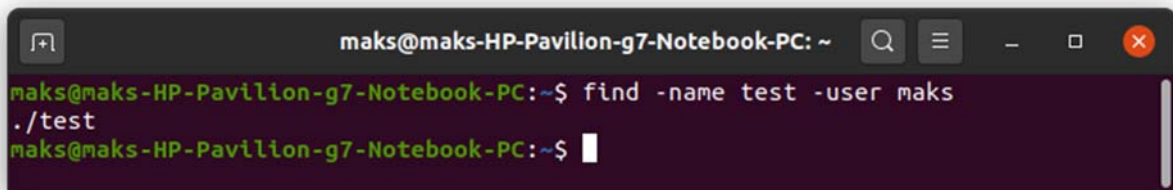
- **-P** никогда не открывать символические ссылки
- **-L** - получает информацию о файлах по символическим ссылкам. Важно для дальнейшей обработки, чтобы обрабатывалась не ссылка, а сам файл.
- **-maxdepth** - максимальная глубина поиска по подкаталогам, для поиска только в текущем каталоге установите 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
- **-type f** - искать только файлы
- **-type d** - поиск папки в Linux

Критериев у команды `find` в Linux очень много, рассмотрим основные:

- **-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

Пример использования команды `find`:



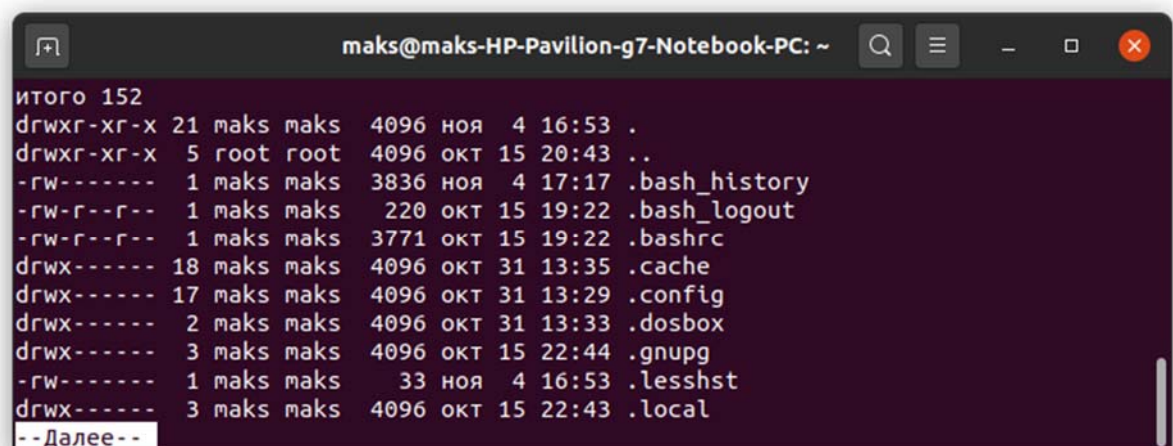
```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ find -name test -user maks
./test
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 7 – Результат использования команды `find` для поиска файла

Теперь разберёмся с понятием конвейера.

Дело в том, что Linux поддерживает использование состыкованных команд, что обычно называется конвейером. Стыковку осуществляет командная оболочка, которая `stdout` первой команды направляет на `stdin` второй команды. Для стыковки используется символ `|`.

Например, мы можем организовать постраничный просмотр какой-нибудь директории большого размера. Сделаем это с помощью состыкованных команд. Напишем в терминале `ls -la | more` и посмотрим результат:



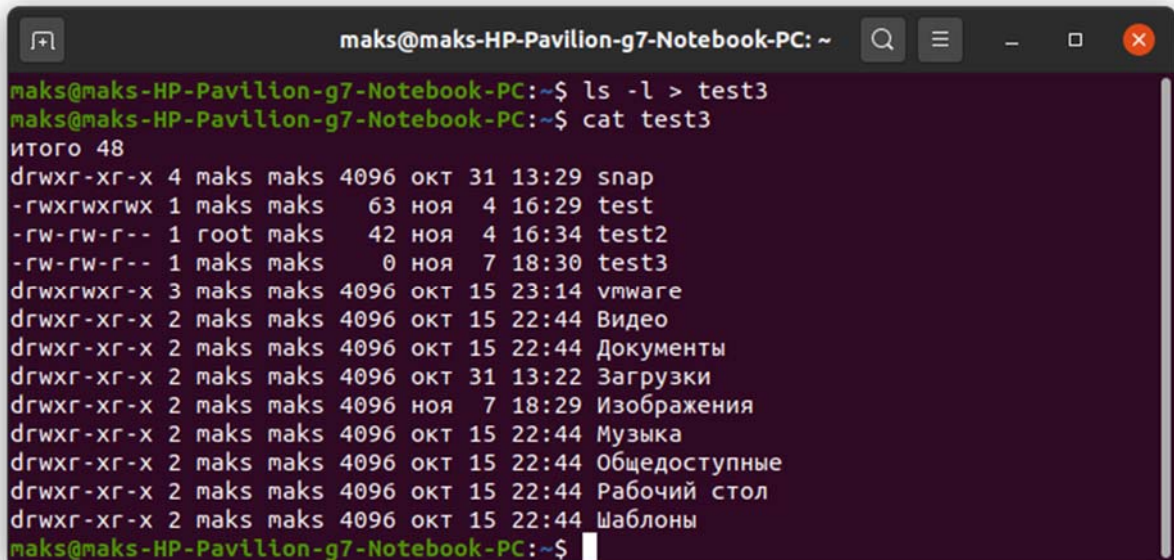
```
итого 152
drwxr-xr-x 21 maks maks 4096 ноя 4 16:53 .
drwxr-xr-x 5 root root 4096 окт 15 20:43 ..
-rw----- 1 maks maks 3836 ноя 4 17:17 .bash_history
-rw-r--r-- 1 maks maks 220 окт 15 19:22 .bash_logout
-rw-r--r-- 1 maks maks 3771 окт 15 19:22 .bashrc
drwx----- 18 maks maks 4096 окт 31 13:35 .cache
drwx----- 17 maks maks 4096 окт 31 13:29 .config
drwx----- 2 maks maks 4096 окт 31 13:33 .dosbox
drwx----- 3 maks maks 4096 окт 15 22:44 .gnupg
-rw----- 1 maks maks 33 ноя 4 16:53 .lessht
drwx----- 3 maks maks 4096 окт 15 22:43 .local
--Далее--
```

Рисунок 8 – Использование конвейера

Затем нам следует разобраться с перенаправлением ввода-вывода в ОС Linux.

Если говорить о перенаправлении вывода, то всё очень просто. Вы можете перенаправить вывод в файл с помощью символа `>`.

Например, перенаправим вывод содержимого директории в файл:

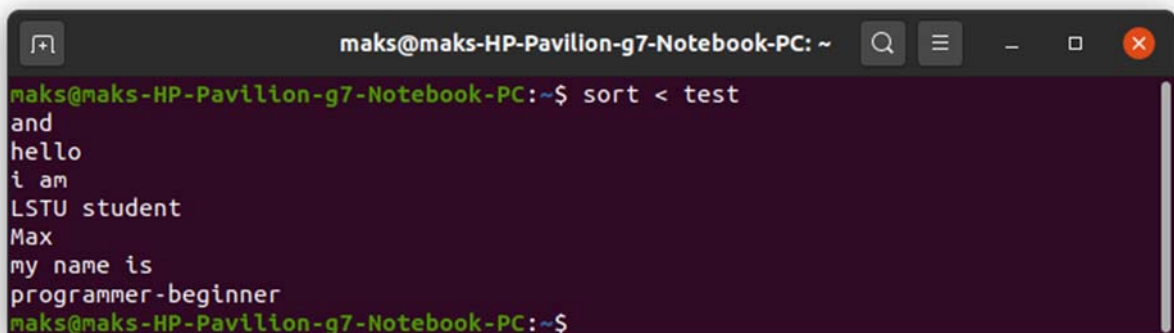
A terminal window titled 'maks@maks-HP-Pavilion-g7-Notebook-PC: ~' with search, menu, and window control icons. It shows the execution of two commands: 'ls -l > test3' and 'cat test3'. The output of 'cat test3' is a detailed directory listing for the home directory, including permissions, owner, size, date, and file names like 'snap', 'test', 'test2', 'test3', 'vmware', 'Видео', 'Документы', 'Загрузки', 'Изображения', 'Музыка', 'Общедоступные', 'Рабочий стол', and 'Шаблоны'.

```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ls -l > test3
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ cat test3
итого 48
drwxr-xr-x 4 maks maks 4096 окт 31 13:29 snap
-rwxrwxrwx 1 maks maks 63 ноя 4 16:29 test
-rw-rw-r-- 1 root maks 42 ноя 4 16:34 test2
-rw-rw-r-- 1 maks maks 0 ноя 7 18:30 test3
drwxrwxr-x 3 maks maks 4096 окт 15 23:14 vmware
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Видео
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Документы
drwxr-xr-x 2 maks maks 4096 окт 31 13:22 Загрузки
drwxr-xr-x 2 maks maks 4096 ноя 7 18:29 Изображения
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Музыка
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Общедоступные
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Рабочий стол
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Шаблоны
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 9 – Перенаправление вывода

Перенаправление ввода же реализуется с помощью символа `<`.

Например, нам нужно отсортировать содержимое файла. Сделаем это с помощью перенаправления ввода:

A terminal window titled 'maks@maks-HP-Pavilion-g7-Notebook-PC: ~' with search, menu, and window control icons. It shows the execution of the command 'sort < test'. The output is the content of the 'test' file, sorted alphabetically: 'and', 'hello', 'i am', 'LSTU student', 'Max', 'my name is', and 'programmer-beginner'.

```
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ sort < test
and
hello
i am
LSTU student
Max
my name is
programmer-beginner
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 10 – Перенаправление ввода

Теперь поговорим о командах `chmod` и `chown` (мы уже говорили о них первой лабораторной работе). Начнём с команды `chmod`.

Эта команда имеет типичный для команд `linux` синтаксис, сначала команда, затем опции, а в конце файл или папка, к которой ее нужно применить:

\$ chmod опции права /путь/к/файлу

Сначала рассмотрим какими бывают права доступа `linux` и как они устанавливаются. Пред этим рекомендую прочитать статью про права, ссылка на которую есть выше. Есть три основных вида прав:

- **r** - чтение;
- **w** - запись;
- **x** - выполнение;
- **s** - выполнение от имени суперпользователя (дополнительный);

Права можно назначать следующим образом: для начала нам нужно определить, для какой категории пользователя (владелец файла, группа файла, остальные пользователи) какие нужно установить права, затем установим права с помощью восьмеричного шаблона:

- **0** - никаких прав;
- **1** - только выполнение;
- **2** - только запись;
- **3** - выполнение и запись;
- **4** - только чтение;
- **5** - чтение и выполнение;
- **6** - чтение и запись;
- **7** - чтение, запись и выполнение.

Приведём пример выполнения команды `chmod`:

```
maks@maks-HP-Pavilion-g7-Notebook-PC: ~  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ls -l  
итого 48  
drwxr-xr-x 4 maks maks 4096 окт 31 13:29 snap  
-rw-rw-r-- 1 maks maks 63 ноя 4 16:29 test  
-rw-rw-r-- 1 maks maks 42 ноя 4 16:34 test2  
drwxrwxr-x 3 maks maks 4096 окт 15 23:14 vmware  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Видео  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Документы  
drwxr-xr-x 2 maks maks 4096 окт 31 13:22 Загрузки  
drwxr-xr-x 2 maks maks 4096 ноя 5 17:31 Изображения  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Музыка  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Общедоступные  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 'Рабочий стол'  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Шаблоны  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ chmod 777 test  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ls -l  
итого 48  
drwxr-xr-x 4 maks maks 4096 окт 31 13:29 snap  
-rwxrwxrwx 1 maks maks 63 ноя 4 16:29 test  
-rw-rw-r-- 1 maks maks 42 ноя 4 16:34 test2  
drwxrwxr-x 3 maks maks 4096 окт 15 23:14 vmware  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Видео  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Документы  
drwxr-xr-x 2 maks maks 4096 окт 31 13:22 Загрузки  
drwxr-xr-x 2 maks maks 4096 ноя 5 17:31 Изображения  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Музыка  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Общедоступные  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 'Рабочий стол'  
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Шаблоны  
maks@maks-HP-Pavilion-g7-Notebook-PC:~$
```

Рисунок 11 – Изменение прав на файл

Здесь мы установили права на чтение, запись и исполнение файла для всех категорий пользователя: первая семёрка открывает доступ для создателя файла, вторая – для группы файла, третья – для остальных пользователей.

Теперь поговорим о команде `chown`.

Синтаксис `chown` прост:

\$ `chown` пользователь опции /путь/к/файлу

В поле пользователь надо указать пользователя, которому мы хотим передать файл.

Вот основные опции, которые могут вам понадобится:

- **-c, --changes** - подробный вывод всех выполняемых изменений;
- **-f, --silent, --quiet** - минимум информации, скрыть сообщения об ошибках;

- **--dereference** - изменять права для файла к которому ведет символическая ссылка вместо самой ссылки (поведение по умолчанию);
- **-h, --no-dereference** - изменять права символических ссылок и не трогать файлы, к которым они ведут;
- **--from** - изменять пользователя только для тех файлов, владельцем которых является указанный пользователь и группа;
- **-R, --recursive** - рекурсивная обработка всех подкаталогов;
- **-H** - если передана символическая ссылка на директорию - перейти по ней;
- **-L** - переходить по всем символическим ссылкам на директории;
- **-P** - не переходить по символическим ссылкам на директории (по умолчанию).

Изменим владельца файла:

```

maks@maks-HP-Pavilion-g7-Notebook-PC: ~
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ls -l
итого 48
drwxr-xr-x 4 maks maks 4096 окт 31 13:29 snap
-rwxrwxrwx 1 maks maks 63 ноя 4 16:29 test
-rw-rw-r-- 1 maks maks 42 ноя 4 16:34 test2
drwxrwxr-x 3 maks maks 4096 окт 15 23:14 vmware
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Видео
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Документы
drwxr-xr-x 2 maks maks 4096 окт 31 13:22 Загрузки
drwxr-xr-x 2 maks maks 4096 ноя 7 14:22 Изображения
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Музыка
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Общедоступные
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 'Рабочий стол'
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Шаблоны
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ sudo chown root test2
[sudo] пароль для maks:
maks@maks-HP-Pavilion-g7-Notebook-PC:~$ ls -l
итого 48
drwxr-xr-x 4 maks maks 4096 окт 31 13:29 snap
-rwxrwxrwx 1 maks maks 63 ноя 4 16:29 test
-rw-rw-r-- 1 root maks 42 ноя 4 16:34 test2
drwxrwxr-x 3 maks maks 4096 окт 15 23:14 vmware
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Видео
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Документы
drwxr-xr-x 2 maks maks 4096 окт 31 13:22 Загрузки
drwxr-xr-x 2 maks maks 4096 ноя 7 14:22 Изображения
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Музыка
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Общедоступные
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 'Рабочий стол'
drwxr-xr-x 2 maks maks 4096 окт 15 22:44 Шаблоны
maks@maks-HP-Pavilion-g7-Notebook-PC:~$

```

Рисунок 12 – Использование chown

Теперь перейдём к рассмотрению наиболее распространённых команд для процессов и поговорим о команде `top`.

Она, в отличие от команды `ps`, позволяет выводить информацию о системе, а также список процессов динамически обновляя информацию о потребляемых ими ресурсах.

Поговорим про интерактивные команды, которые вы можете выполнять во время работы программы.

- **h** - вывод справки по утилите;
- **q** или **Esc** - выход из `top`;
- **A** - выбор цветовой схемы;
- **d** или **s** - изменить интервал обновления информации;
- **H** - выводить потоки процессов;
- **k** - послать сигнал завершения процессу;
- **W** - записать текущие настройки программы в конфигурационный файл;
- **Y** - посмотреть дополнительные сведения о процессе, открытые файлы, порты, логи и т.д.;
- **Z** - изменить цветовую схему;
- **I** - скрыть или вывести информацию о средней нагрузке на систему;
- **m** - выключить или переключить режим отображения информации о памяти;
- **x** - выделять жирным колонку, по которой выполняется сортировка;
- **y** - выделять жирным процессы, которые выполняются в данный момент;
- **z** - переключение между цветным и одноцветным режимами;
- **c** - переключение режима вывода команды, доступен полный путь и только команда;
- **F** - настройка полей с информацией о процессах;
- **o** - фильтрация процессов по произвольному условию;
- **u** - фильтрация процессов по имени пользователя;
- **V** - отображение процессов в виде дерева;

- **i** - переключение режима отображения процессов, которые сейчас не используют ресурсы процессора;
- **n** - максимальное количество процессов, для отображения в программе;
- **L** - поиск по слову;
- **< >** - перемещение поля сортировки вправо и влево;

Теперь поговорим о возможности запуска процессов в supervisor.

Supervisor – это менеджер процессов, который существенно упрощает управление долго работающими программами, предоставляя простой и понятный интерфейс. Данная утилита упрощает работу с процессами и позволяет администратору контролировать процессы. Помимо этого, он способен обеспечить бесперебойную работу веб-сервиса.

Если говорить о возможности запуска процессов по расписанию, то для этого используется утилита cron.

Cron – программа, предназначенная для выполнения заданий в определенное время, или через определенные промежутки времени.

Для настройки времени, даты и интервала, когда нужно выполнять задание, используется команда crontab, которую рекомендуется выполнять с опцией -e.

Давайте рассмотрим синтаксис настройки одной задачи cron:

минута час день месяц день_недели /путь/к/исполняемому/файлу

Нужно сказать, что обязательно нужно писать полный путь к команде, потому что для команд, запускаемых от имени cron переменная среды PATH будет отличаться, и сервис просто не сможет найти вашу команду. Это вторая самая распространенная причина проблем с Cron. Дата и время указываются с помощью цифр или символа '*'. Этот символ означает, что нужно выполнять каждый раз, если в первом поле - то каждую минуту и так далее.

Данная программа является ультимативным решением для планирования автоматических задач.

Вывод

В ходе выполнения лабораторной работы были закреплены знания по командам `cat`, `head`, `tail`, `more`, `less`, `grep`, `find`, `chown`, `chmod`. Также изучено понятие конвейера и основные принципы перенаправления ввода-вывода. Изучены утилиты `top` для управления процессами, `cron` для планирования процессов и `supervisor` для администрирования.