Ход занятия.

1. Работа с файлами в Линукс является одним из базовых навыков. На этом занятии мы с вами узнаем способы вывода информации на экран из файлов. Оговорюсь сразу – сейчас я подразумеваю только обычные (регулярные) файлы.

Информация внутри них может содержаться в 2 видах – текстовом ASCII или бинарном. Для каждого из этих типов информации существуют программы, способные ее выводить на экран.

Для текстовых файлов наиболее часто используются команды cat, tac, more, less, head и tail.

Рассмотрим их поподробнее:

cat filename – команда используется для вывода текстовой информации из файла на экран. Например:

```
[student@localhost student]$ cat lesson6_st.txt
Это первый текстовый файл, который Вам удалось отобразить на экране.
В нем содержиться 12 строк текстовой информации.
Такой файл можно отобразить командой cat.
В обратном порядке такой файл можно отобразить командой tac.
Посчитать строки в нем можно командой nl.


This a first text file, where You see into your screen.
His lenght is 12 lines.
This file printing with command cat.
Inverse output make command tac.
Calculate lines in file with help command nl.
[student@localhost student]$_
```

Команда cat имеет несколько параметров. Опишем наиболее используемые из них:

-b (--number-nonblank) – пронумеровать все непустые строки;

-n (--number) – пронумеровать все строки;

-s (--squeeze-blank) – отобразить несколько подряд идущих пустых строк в виде одной пустой строки;

-T (--show-tabs) – показать символы табуляции, отобразив их как “^|”;

-E (--show-ends) – показать символы конца строки как “$”.

Попробуем использовать некоторые из этих параметров:

```
[student@localhost student]$ cat -b lesson6_st.txt
1  Это первый текстовый файл, который Вам удалось отобразить на экране.
2  В нем содержиться 12 строк текстовой информации.
3  Такой файл можно отобразить командой cat.
4  В обратном порядке такой файл можно отобразить командой tac.
5  Посчитать строки в нем можно командой nl.


6  This a first text file, where You see into your screen.
7  His lenght is 12 lines.
8  This file printing with command cat.
9  Inverse output make command tac.
10  Calculate lines in file with help command nl.
[ student@localhost student]$ cat -n lesson6_st.txt
1  Это первый текстовый файл, который Вам удалось отобразить на экране.
2  В нем содержиться 12 строк текстовой информации.
3  Такой файл можно отобразить командой cat.
4  В обратном порядке такой файл можно отобразить командой tac.
5  Посчитать строки в нем можно командой nl.
6
```

```
7
8  This a first text file, where You see into your screen.
9  His lenght is 12 lines.
10  This file printing with command cat.
11  Inverse output make command tac.
12  Calculate lines in file with help command nl.
[ student@localhost student]$ cat -E lesson6_st.txt
Это первый текстовый файл, который Вам удалось отобразить на экране.$
В нем содержиться 12 строк текстовой информации.$
Такой файл можно отобразить командой cat.$
В обратном порядке такой файл можно отобразить командой tac.$
Посчитать строки в нем можно командой nl.$
$
$
This a first text file, where You see into your screen.$
His lenght is 12 lines.$
This file printing with command cat.$
Inverse output make command tac.$
Calculate lines in file with help command nl.$
[student@localhost student]$_
```

tac filename – эта команда используется для вывода на экран информации из файла в обратном порядке.

```
[student@localhost student]$ tac lesson6_st.txt
Calculate lines in file with help command nl.
Inverse output make command tac.
This file printing with command cat.
His lenght is 12 lines.
This a first text file, where You see into your screen.


Посчитать строки в нем можно командой nl.
В обратном порядке такой файл можно отобразить командой tac.
Такой файл можно отобразить командой cat.
В нем содержиться 12 строк текстовой информации.
Это первый текстовый файл, который Вам удалось отобразить на экране.
[student@localhost student]$_
```

По умолчанию разделителем записей для этой команды является символ корнца строки (/n). Но такое положение дел можно изменить, используя параметр -s (--separator=):

```
[student@localhost student]$ tac --separator=" " lesson6_st.txt
nl.
command help with file in lines tac.
Calculate command make output cat.
Inverse command with printing file lines.
This 12 is lenght screen.
His your into see You where file, text first a nl.


This командой можно нем в строки tac.
Посчитать командой отобразить можно файл такой порядке обратном cat.
В командой отобразить можно файл информации.
Такой текстовой строк 12 содержиться нем экране.
В на отобразить удалось Вам который файл, текстовый первый Это
[ student@localhost student]$
```

Но может возникнуть ситуация, когда размер файла намного превосходит количество строк, способных уместиться на экране. В таком случае при использовании утилиты cat вы не сможете комфортно прочесть файл, так как он будет очень быстро выведен. Справиться с этой ситуацией помогут нам команды more и less.

more filename – эта команда позволяет просматривать длинные файлы по частям. Так например:

```
[student@localhost student]$ more lesson6_bt.txt
```

```
KDE -- это интерактивная рабочая среда.
Другими словами, это набор программ,
технологий и документации, которые призваны
облегчить жизнь пользователей персональных
компьютеров. KDE нацелена на рабочие станции
под управлением Unix. Среди ее отличительных
особенностей -- "прозрачная" работа в сети и
современная философия работы.

Создатели этой рабочей среды - сообщество
разбросанных по всему миру программистов. Эти
люди - разработчики свободного программного
обеспечения - основной своей задачей считают
выпуск высококачественного программного продукта,
который позволит пользователю с легкостью
задействовать всю вычислительную мощь своего
компьютера.

KDE возникла как ответ на потребность в удобной в
использовании рабочей среде для рабочих станций под
Unix, аналогичной уже существующим системам на базе
MacOS или Windows. Основные инструменты для достижения
этой цели - это улучшенные средства межпрограммных
связей, повторное использование компонентов, технология
"drag and drop", единый внешний вид и многое другое. Таким
образом, KDE предлагает нечто большее, нежели
традиционные менеджеры окон Unix.

Стабильность, масштабируемость и открытость - вот те
--More--(56%)
```

Команда more использует для прокрутки две клавиши – пробел (показать следующий экран) и Enter (показать следующую строку). Но у more есть один недостаток – она способна прокручивать текст только вперед. То есть если вы уже смотрите второй экран, то к первому никак вернуться будет нельзя. Эту проблему с легкостью решает команда less.

less filename – позволяет просматривать файлы любой длины, прокручивая их в любую сторону. Например:

```
[student@localhost student]$ less lesson6_bt.txt
KDE -- это интерактивная рабочая среда.
Другими словами, это набор программ,
технологий и документации, которые призваны
облегчить жизнь пользователей персональных
компьютеров. KDE нацелена на рабочие станции
под управлением Unix. Среди ее отличительных
особенностей -- "прозрачная" работа в сети и
современная философия работы.

Создатели этой рабочей среды - сообщество
разбросанных по всему миру программистов. Эти
люди - разработчики свободного программного
обеспечения - основной своей задачей считают
выпуск высококачественного программного продукта,
который позволит пользователю с легкостью
задействовать всю вычислительную мощь своего
компьютера.

KDE возникла как ответ на потребность в удобной в
использовании рабочей среде для рабочих станций под
Unix, аналогичной уже существующим системам на базе
MacOS или Windows. Основные инструменты для достижения
этой цели - это улучшенные средства межпрограммных
связей, повторное использование компонентов, технология
"drag and drop", единый внешний вид и многое другое. Таким
образом, KDE предлагает нечто большее, нежели
```

```
традиционные менеджеры окон Unix.

Стабильность, масштабируемость и открытость - вот те
качества,которые сделали Unix бесспорным выбором
lesson6_bt.txt lines 1-30/47 59%
```

Команда less понимает следующие комбинации клавиш:
Enter (стрелка вниз) – перейти на одну строку вниз;
Пробел (Page Down) – перейти на страницу вниз по тексту;
Page Up – перейти на страницу вверх по тексту;
стрелка вверх – перейти на строку вверх по тексту;
/ - поиск;
Home – перейти к началу текста;
End – перейти к концу текста;
q – выйти из программы less.

Но если нам не нужно отображать весь файл а только необходимо убедиться что это именно то, что мы ищем, то нам помогут команды head и tail.

head filename1 [filename2 ...] – по умолчанию выводит 10 первых строк файла. Команде head можно задавать параметры, вот самые используемые из них:

-n lines – вывести не 10 а lines строк от начала файл;
-c bytes – вывести bytes байт с начала файла;
-q – не печатать заголовки файлов перез выводом (при выводе нескольких файлов сразу).

```
[student@localhost student]$ head lesson6_bt.txt
KDE -- это интерактивная рабочая среда.
Другими словами, это набор программ,
технологий и документации, которые призваны
облегчить жизнь пользователей персональных
компьютеров. KDE нацелена на рабочие станции
под управлением Unix. Среди ее отличительных
особенностей -- "прозрачная" работа в сети и
современная философия работы.

Создатели этой рабочей среды - сообщество
[student@localhost student]$ head -n 2 lesson6_bt.txt
KDE -- это интерактивная рабочая среда.
Другими словами, это набор программ,
[student@localhost student]$_
```

tail filename – по умолчанию выводит 10 последних строк файла. Вот наиболее используемые ее параметры:

-n lines – вывести не 10, а lines последних строк;
-c bytes – вывести bytes последних байт файла;
-f – войти в постоянный цикл по считыванию конца файла. Таким образом при поступлении в файл новой информации пользователь может вести мониторинг ее в реальном времени. Выход из этого режима осуществляется комбинацией клавиш <Ctrl>+<C>.

```
[student@localhost student]$ tail lesson6_bt.txt
серверных применений и в научных учреждениях, но не привлекала
обычных пользователей.

Без Unix не было бы и Интернета, по крайней мере, в нашем
теперешнем представлении. Однако до недавнего времени Unix
не отвечал запросам рядового пользователя. Этот факт особенно
неприятен, поскольку существуют такие варианты Unix, как Linux и
FreeBSD, NetBSD и т.д., которые свободно доступны в Интернете и
славятся исключительным качеством и стабильностью.
```

```
[student@localhost student]$ tail -c 100 lesson6_bt.txt
 т.д., которые свободно доступны в Интернете и
славятся исключительным качеством и стабильностью.

[student@localhost student]$
```

Вернемся к началу нашего занятия и вспомним что есть еще и второй тип содержимого файлов – бинарные данные. Для просмотра такого типа файлов нам не подойдут команды типа more или cat:

```
[student@localhost student]$ cat lesson6.bin
RIFFWAVEfmt
D┘Ddataъ────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~
[student@localhost student]$_
```

Но вот команда od предназначена как раз для просмотра файлов такого типа.

od filename – просматриват бинарные файлы в виде дампа байтов. Наиболее популярные параметры команды od:

-A radix – указать систему счисления, в которой будут указываться адреса. Параметр radix может принимать следующие значения: d – десятичная, o – восьмеричная (по умолчанию), x – шеснадцатиричная, n – не выводить адресов.

-radix – указать систему счисления, в которой будут указываться данные. Параметр radix может принимать следующие значения: d – десятичная, o – восьмеричная (по умолчанию), x – шеснадцатиричная, n – не выводить адресов.

-j bytes – пропустить bytes байт от начала файла.

-v – включить вывод последовательных одинаковых строк. По умолчанию od выводит только первую строку из множества, а вместо остальных – символ “*”.

```
[student@localhost student]$ od -x -j 100 lesson6.bin
0000144 8080 8080 8080 8080 8080 8080 8080 8080
*
0000224 8080 7f80 7f7f 807f 7f80 7f7f 7f7f 807f
0000244 8080 8080 7f80 7f7f 7f7f 7f7f 7f7f 7f7f
0000264 7f7f 7f7f 807f 7f80 7f7f 7f7f 7f7f 7f7f
0000304 7f7f 7f7f 807f 8080 8080 8080 8080 7f80
0000324 7f7f 7f7f 7f7f 7f7f 807f 8080 8080 7f80
0000344 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
*
0000404 7f7f 7f7f 7f7f 7f7f 7f7f 807f 8080 8080
0000424 7f80 7f7f 7f7f 7f7f 807f 8080 7f80 7f7f
0000444 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
*
0000504 7f7f 7f7f 807f 8080 7f80 7f7f 7f7f 7f7f
0000664 7e7e 7e7e 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f
0000704 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e
0000724 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e
0000744 7e7e 7e7e 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f
0000764 7e7f 7e7e 7e7e 7e7e 7e7e 7e7e 7f7e 7f7f
0001004 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e 7e7e 0a7e
0001024
[student@localhost student]$ od -x -j 100 -v lesson6.bin
0000144 8080 8080 8080 8080 8080 8080 8080 8080
0000164 8080 8080 8080 8080 8080 8080 8080 8080
0000204 8080 8080 8080 8080 8080 8080 8080 8080
0000224 8080 7f80 7f7f 807f 7f80 7f7f 7f7f 807f
```

```
0000244 8080 8080 7f80 7f7f 7f7f 7f7f 7f7f 7f7f
0000264 7f7f 7f7f 807f 7f80 7f7f 7f7f 7f7f 7f7f
0000304 7f7f 7f7f 807f 8080 8080 8080 8080 7f80
0000324 7f7f 7f7f 7f7f 7f7f 807f 8080 8080 7f80
0000344 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
0000364 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
0000404 7f7f 7f7f 7f7f 7f7f 7f7f 807f 8080 8080
0000424 7f80 7f7f 7f7f 7f7f 807f 8080 7f80 7f7f
0000444 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
0000464 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
0000504 7f7f 7f7f 807f 8080 7f80 7f7f 7f7f 7f7f
0000524 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f
0000544 7f7f 7f7f 7f7f 7e7f 7e7e 7e7e 7e7e 7e7e
0000564 7e7e 7e7e 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f
0000604 7f7f 7e7f 7e7e 7e7e 7e7e 7e7e 7e7e 7e7e
0000624 7f7e 7f7f 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e
0000644 7e7e 7e7e 7f7e 7f7f 7e7f 7e7e 7e7e 7e7e
0000664 7e7e 7e7e 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f
0000704 7f7f 7f7f 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e
0000724 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e
0000744 7e7e 7e7e 7e7e 7f7e 7f7f 7f7f 7f7f 7f7f
0000764 7e7f 7e7e 7e7e 7e7e 7e7e 7e7e 7f7e 7f7f
0001004 7f7f 7f7f 7f7f 7f7f 7e7f 7e7e 7e7e 0a7e
0001024
[student@localhost student]$
```

2. Если вам понадобилось в Линукс получить справку по какой либо команде, то стоит воспользоваться справочной системолй ***man***. В Линукс при выводе справки man ведет себя аналогично less.

man command

man поддерживает несколько параметров. Наиболее часто используются следующие:

-k keyword – показывает все страницы руководства, на которых встречается слово keyword;

<число> - указывает брать руководство из раздела с номером <число>

Имеются следующие разделы man:

Номер	***Значение***
1	Команды и приложения пользовательского уровня
2	Системные вызовы и коды ошибок ядра
3	Библиотечные функции
4	Драйверы устройств и сетевые протоколы
5	Стандартные форматы файлов
6	Игры и демонстрационные программы
7	Различные файлы и документы
8	Команды системного администрирования
9	Внутренние интерфейсы и спецификации ядра

Подробное руководство по команде man можно получить используюя команду man man.

3. Во многих случаях при работе с файлами нам необходимо будет использовать вывод команды. Решеним проблемы перенаправления вывода занимаются командные оболочки, которые мы с вами еще обсудим на занятиях.

В работе с коммандной строкой Linux есть понятия стандартных устройств ввода, вывода и вывода ошибок.

stdin – стандартное устройство ввода. Имеет файловый указатель №0. Автоматически открывается всеми процессами.

stdout – стандартное устройство вывода. Имеет файловый указатель №1. Автоматически открывается всеми процессами.

stderr – стандартный поток ошибок (специальное устройство вывода для

сообщений об ошибках. Имеет файловый указатель №2.
Автоматически открывается всеми процессами.

По умолчанию практически все команды Linux используют для ввода информации stdin, а для вывода stdout и stderr, если им параметрами не указано обратное.

Операторы перенаправления способны изменять направление вывода и ввода информации. Так оператор:

> - перенаправляет стандартный поток в файл (другой поток). При этом если файл существует, то он перезаписывается, если не существует – создается.

>> - перенаправляет стандартный поток в файл. При этом если файл существует, то информация добавляется в конец, если не существует – файл создается.

< - перенаправляет содержимое указанного файла на стандартный ввод программе.

>& - перенаправляет стандартные потоки вывода и ошибок друг в друга.

```
Пример1:
[gserg@WEBMEDIA Занятие 7]$ ls
lesson7_1.txt
[gserg@WEBMEDIA Занятие 7]$ head -n 5 lesson7_1.txt > fivelines.txt
[gserg@WEBMEDIA Занятие 7]$ ls
fivelines.txt  lesson7_1.txt
[gserg@WEBMEDIA Занятие 7]$ cat fivelines.txt
В сочетании со свободными версиями Unix, KDE дарит
миру открытую и абсолютно свободную от всех
ограничений рабочую среду для домашнего или
профессионального применения.
Эта платформа доступна всем желающим бесплатно,
[gserg@WEBMEDIA Занятие 7]$_

Пример2:
[gserg@WEBMEDIA Занятие 7]$ tail -n 1 fivelines.txt >> fivelines.txt
[gserg@WEBMEDIA Занятие 7]$ cat fivelines.txt
В сочетании со свободными версиями Unix, KDE дарит
миру открытую и абсолютно свободную от всех
ограничений рабочую среду для домашнего или
профессионального применения.
Эта платформа доступна всем желающим бесплатно,
Эта платформа доступна всем желающим бесплатно,
[gserg@WEBMEDIA Занятие 7]$

Пример3:
[gserg@WEBMEDIA Занятие 7]$ cat < fivelines.txt
В сочетании со свободными версиями Unix, KDE дарит
миру открытую и абсолютно свободную от всех
ограничений рабочую среду для домашнего или
профессионального применения.
Эта платформа доступна всем желающим бесплатно,
Эта платформа доступна всем желающим бесплатно,
[gserg@WEBMEDIA Занятие 7]$_
```

Пример4:

```
[gserg@WEBMEDIA Занятие 7]$ cat nofile.txt > result.out 2>&1
```

```
[gserg@WEBMEDIA Занятие 7]$ cat result.out
cat: nofile.txt: No such file or directory
[gserg@WEBMEDIA Занятие 7]$_
```

Такой способ перенаправления вывода хорош при работе с файлами. Но как быть, если вывод какой-либо программы нам необходимо перенаправить на вход другой? Для этого в Linux существует такое понятие как каналы.

Канал – программный интерфес, позволяющий процессам обмениваться данными (односторонний поток).

Организацией канала занимается shell. Для управления каналом существует оператор “|”. Приведу пример:

Пример5:

```
[gserg@WEBMEDIA Занятие 7]$ cat lesson7_1.txt | tail –n 3 | less
в течение многих лет с удовольствием используют ученые и
профессионалы–компьютерщики во всем мире.

lines 1–3/3 (END)
```

Рассмотрим поподробнее все, что произошло при выполнении данной нами группы команд:

- Команда cat прочитала файл lesson7_1.txt и передала его содержание на стандартный ввод команды tail
- Команда tail исходя из заданных ей параметров взяла 3 последние строки текстового файла и передала их на стандартный ввод команде less
- Команда less вывела информацию со стандартного ввода на экран и стала ожидать действий пользователя.

Таким образом одна команда передавала по ***каналу*** информацию другой команде.

4. Иногда необходимо вывести информацию, содержание которой вы знаете, а вот расположение – нет. Именно для таких случаев существуют регулярные выражения. Перечислю наиболее используемые из них, хотя оговорюсь сразу, что на самом деле регулярных выражений намного больше.

Регулярное выражение – средство указания шаблона для поиска его в тексте.

^ - начало строки

$ - конец строки

[] - любой символ из заключённых в скобки. Поддерживает диапазоны, например [0-9] – цифры, [a-zA-Z] -все буквы латинского алфавита

[^] - любой символо за исключением заключенных в скобки

\ - отменяет действие любого метасимвола. Например \$ - обозначает символ $, а не \ в конце строки, а \\$ - символ \ в конце строки.

. - любой один символ.

* - 0 или более раз в тексте встречается предыдущий шаблон. Так например выражение .* означает любой набор символов.

Регулярные выражения поддержкиваются практически всеми текстовыми редакторами Linux. Существует также программа фильтрации текста grep. Она также использует регулярные выражения. Её мы с Вами и рассмотрим.

grep regexp file – утилита фильтрации текста. Ищет в файле file строки, в которых встречается выражение, соответствующее шаблону regexp и выводит их на стандартный вывод.

Пример6:

```
[gserg@WEBMEDIA Занятие 7]$ grep KDE lesson7_1.txt
В сочетании со свободными версиями Unix, KDE дарит
что комбинация Unix и KDE наконец подарит пользователю
[gserg@WEBMEDIA Занятие 7]$_
```

Пример7:
```
[gserg@WEBMEDIA Занятие 7]$ grep ^Э.*  lesson7_1.txt
Эта платформа доступна всем желающим бесплатно,
[gserg@WEBMEDIA Занятие 7]$_
```

Как правило, утилиту grep используют не только для фильтрации текстовых файлов, но и, например, для фильтрации вывода каких-либо команд.

В примере ниже мы с вами попытаемся найти все файлы, начинающиеся на букву “f” в каталога bin:

Пример8:
```
[gserg@WEBMEDIA Занятие 7]$ ls /bin | grep ^f.*
false*
fbresolution*
fgrep@
find*
[gserg@WEBMEDIA Занятие 7]$_
```

Подробнее о команде grep можно узнать из страницы справочного руководства man (man grep).

5. Как известно, при работе в любой операционной системе часто возникает необходимоть создавать резервные копии важной информации. Для такой работы необходимы программы-архиваторы и программы сжатия.

В комплект поставки ОС Linux входят как правило сразу несколько программ архивирования и/или сжатия. Но стандартом de facto для unix-like ОС стали архиватор tar и программа сжатия gzip (GNU zip).

tar (GNU tar – GNU tape arhiver) – программа для создания архивов. Современный tar поддерживает сжатие, но для обеспечения совместимости с более ранними версиями Linux и Unix советуется использовать насжатый архив tar, либо сжимать его после создания одной из утилит сжатия.

tar имеет множество параметров и опций. Мы с вами рассмотрим наиболее употребительные из них:

-***c*** – создать архив
-***r*** – добавить файлы в архив
-***A*** – добавить содержимое tar-файлов в архив
--***delete*** – удалить файлы из архива (невозможно использование на архивных лентах)
-***t*** – вывести список файлов в архиве
-***x*** – извлечь файлы из архива
-***f*** – информация будет извлекаться из файла
-***v*** – расширенный вывод информации о выполняемых действиях

Пример9:
```
[gserg@WEBMEDIA Занятие 7]$ tar -xvf lesson_2.tar
./lesson_2/
./lesson_2/.quota
./lesson_2/arch
./lesson_2/chain.b
./lesson_2/ida/
./lesson_2/hda1
tar: ./lesson_2/hda1: Cannot mknod: Operation not permitted
tar: Выход, отложенный по результатам предыдущих ошибок
[gserg@WEBMEDIA Занятие 7]$_
```

Пример10:
```
[gserg@WEBMEDIA Занятие 7]$ tar -tf lesson_2.tar
./lesson_2/
./lesson_2/.quota
./lesson_2/arch
```

```
./lesson_2/chain.b
./lesson_2/ida/
./lesson_2/hda1
[gserg@WEBMEDIA Занятие 7]$_
```

Пример11:

```
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2/  lesson_2.tar  lesson7_1.txt
[gserg@WEBMEDIA Занятие 7]$ tar -cf lesson7.tar lesson7_1.txt
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2/  lesson_2.tar  lesson7_1.txt  lesson7.tar
[gserg@WEBMEDIA Занятие 7]$ tar -tf lesson7.tar
lesson7_1.txt
[gserg@WEBMEDIA Занятие 7]$_
```

gzip options filename – утилита компрессии (сжатия) файлов. Производит сжатие| декомпрессию файла filename или другие сопутствующие действия в соответствии с опциями options:

только имя файла – сжимает этот файл.

-***d*** – декомпрессия файла

-***t*** – проверяет целостность архива

-***v*** – расширенный вывод информации о выполняемых действиях

Пример12:

```
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2.tar  lesson7_1.txt  pict.png.gz
[gserg@WEBMEDIA Занятие 7]$ gzip -dv pict.png.gz
pict.png.gz:              0.3% -- replaced with pict.png
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2.tar  lesson7_1.txt  pict.png
[gserg@WEBMEDIA Занятие 7]$_
```

Пример13:

```
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2.tar  lesson7_1.txt  pict.png
[gserg@WEBMEDIA Занятие 7]$ gzip pict.png
[gserg@WEBMEDIA Занятие 7]$ ls
lesson_2.tar  lesson7_1.txt  pict.png.gz
[gserg@WEBMEDIA Занятие 7]$_
```

Пример14:

```
[gserg@WEBMEDIA Занятие 7]$ gzip -tv pict.png.gz
pict.png.gz:              OK
[gserg@WEBMEDIA Занятие 7]$_
```

Для создания сжатого архива из множества файлов используйте сразу обе эти утилиты путем передачи информации каналами или последовательно запуская их из командной строки.