**Второе занятие Linux:**

**mkdir** (make directory) - создавать папки

**rmdir** (remove directory) - удалять папки

**touch** - создавать файлы. Как я уже сказал, любой файл создается в Linux, как текстовый и открывается текстовым редактором, так как в нету привязки к расширению (как в windows), где система сначала нюхает расширение и делается вывод, какой программой необходимо файл запускать.

**cp** (copy) - команда копирования

**mv** (move) - команда перемещения

**rm** (remove) - команда удаления файла. Иногда может помогать нам удалять целые каталоги, но для этого нужен флаг **-rf**

**Так как командная строка не имеет функции отмены - всегда нужно понимать, что и зачем ты делаешь. Ведь “вернуть в зад” удаленный файл или папку не будет возможности.**

Открываем наш тестовый терминал по ссылке:

<https://bellard.org/jslinux/index.html>

**Заходим в папку home, как мы помним - командой cd**

140 cd /home

**создаем папку user1**

141 mkdir user1

**смотрим, что папка создана, и что еще есть у нас в этой папке**

142 ls

**переходим в папку user1**

143 cd user1

**создаем еще одну папку task1**

144 mkdir task1

**заходим в нее**

145 cd task1

**и уже в этой папке создаем файл (в данном случае, для наглядности, с расширением \*.txt)**

146 touch file1.txt

*Давайте посмотрим, где мы находимся. Какая команда нам в этом поможет?* **pwd**

147 pwd

**так же давайте проверим наши разрешения и посмотрим на размер созданного файла. Он будет нулевым, так как в нем пусто.**

148 ls -l

Флаги и аргументы. Флаги или ключи идут с минусом -ключ меняет визуальное отображение. К примеру команда ls -l (l - long) - подробная информация. Есть еще **-a** где показываются все файлы, в том числе скрытые. Есть псевдоним -ls -la

149 ls -a

151 ls -l

Если случилось страшное и наш экран забит кучей лишней информации, то всегда можно сделать команду clear и наш терминал станет девственно чистым.

Ctrl l - очистить экран и **clear**

Так же есть еще одна интересная фишка - это ветвления папки:

Помогает нам в этом команда **tree**:

**Делать мы этого не будем в корневом каталоге, так как у нас терминал начнет показывать абсолютно все файлы!**

Если использовать эту команду со слешем /, то она показывает ветвления от папки, которую ты запросишь в терминале.

**пример: tree /home/**

Важный момент! Tree есть не во всех дистрибутивах. Пробовать можно во всех, но иногда результата мы не увидим. Пугаться не стоит. Просто мы используем такую версию linux, где tree нету.

Должно получиться в папке home несколько папок. Каждый файл создается чуть быстрее.

В терминале можно посмотреть дату и время. Это, если ты вдруг забыл, сколько сейчас времени и когда там уже перерыв. Делается это командой **date**:

152 date

Так же можно посмотреть, что мы там понавводили за все занятие и в каком порядке. Для этого нам поможет команда **History**:

155 history

**Теперь давайте попробуем разобраться, как же создавать файлы и папки одной строкой, не заходя каждый раз в новосозданную папку и подпапки.**

156 mkdir -p /home/user2/task2/

157 touch /home/user2/task2/file2.txt

158 mkdir -p /home/user3/task3/

флаг **-p** создать путь сокращение от path если нужно создать папку и путь

Если просто папку mkdir то создается просто папка без вложенности.

иногда используется для неотображения ошибок в некоторых версиях Linux

Второе назначение - убедиться в наличии пути

-p используют для скриптов, чтобы не возвращать ошибку, а убеждаться в наличии путей.

**Напомню, что такое скрипт:**

Скрипт - это небольшая программа, которая содержит последовательность действий, созданных для автоматического выполнения задачи.

**создадим папку**

159 mkdir /home/user2/task2/

Увидим ошибку. Не пугаемся, так выглядит ответ, если файл или папка уже есть.

Тут важно сказать о путях. В линуксе существует Полный (он же абсолютный путь) и относительный.

* **Полный, абсолютный путь linux от корня файловой системы** - он начинается от корня "/" и описывает весь путь к файлу;
* **Относительный путь linux** - это путь к файлу относительно текущей папки, такие пути часто вызывают путаницу.

Простой пример. Сидите вы дома, и вдруг стук в дверь. Вы открываете, а там полиция. Вас кладут лицом в пол и спрашивают:

* Вася?

А вы им - НЕТ! Вася живет на два этажа выше во второй квартире слева! - это относительный путь.

Но если вы скажете - Вася живет на пятом этаже во второй квартире слева! - это уже будет полный.

* **Путь относительно домашний папки текущего пользователя.** - путь в файловой системе, только не от корня, а от папки текущего пользователя.

Так вот. Если ты не уверен в правильности пути - используй полный, проверяя себя табом каждый шаг. Проверено, помогает!

160 ls -la /home/user2/task2/file2.txt

161 pwd

162 cd /home

**тут мы смотрим созданную папку:**

165 ls user1

**а тут создаем новую:**

167 mkdir -p user3/task3

**Cp - копирование два аргумента, что и куда.**

**скопируем file2.txt из** /home/user2/task2/file2.txt **в home/user3/task3/ но с переименованием**

cp /home/user2/task2/file2.txt user3/task3/file3.txt

**создадим папку task4**

169 mkdir -p /home/user4/task4

**создадим файл file4**

170 touch /home/user4/task4/file4.txt

**проверим все пути**

171 ls -a /

172 ls /home/

173 ls /home/user1/task1/file1.txt

174 ls /home/user2/task2/file2.txt

175 ls /home/user3/task3/file3.txt

176 ls /home/user4/task4/file4.txt

**посмотрим на историю**

177 history

**скопируем file2 в папку /tmp**

178 cp /home/user2/task2/file2.txt /tmp

179 ls /tmp/file2.txt

**так же скопируем**

180 cp /home/user3/task3/file3.txt /tmp

181 cp /home/user2/task2/file2.txt /tmp/newfile

**проверим**

182 ls /tmp/newfile

183 mv /tmp/newfile /opt

184 cp /home/user2 /tmp/newfile

**выдаст ошибку, поэтому используем флаг -r**

**все дело, что эта команда копирует только файлы. А значит, что нам приходится использовать этот флаг**

cp -r /home/user2 /tmp/newfile

185 cp /home/user2 /tmp

186 cp -R /home/user2 /tmp

187 ls /tmp/

**Перенос и переименование**

Перемещение и переименование. При помощи команды mv - тоже старое название - новое название. Переименовывая ты переносишь файл в ту же локацию, только с другим именем.

**Переносить можно так же с переименованием:**

188 mv /opt/newfile /opt/newfile.txt

189 mv /opt/newfile.txt /tmp/new\_text

190 rm /tmp/file2.txt

191 ls /tmp

195 history

**вывод истории и запись ее на свой компьютер:**

history > /tmp/history.txt

export\_file /tmp/history.txt

0 cd ..

1 whoami

2 ls

3 ls --help

4 ls

5 ls -l

6 ls -a

7 ls -la

8 ls -laL1

9 clear

10 cd

11 cd /

12 ls

13 cd /home/

14 mkdir user1

15 ls

16 cd /user1/

17 cd /home/user1/

18 mkdir task1

19 ls

20 ls /bin

21 ls -l /bin

22 clear

23 cd task1/

24 touch file.txt

25 cd ..

26 history

27 cd /opt

28 cd opt

29 cd /home/

30 tree /

31 clear

32 tree

33 pwd

34 mkdir /home/user2/task2

35 mkdir -p /home/user2/task2/

36 tree

37 cd

38 touch /home/user2/task2/file2.txt

39 tree

40 tree /home

41 mkdir /home/user3 /home/user4 /home/user5 /opt/ivan

42 mkdir /home/user6 && touch /home/user6

43 tree /home/

44 tree /home

45 ls /home/

46 ls /home/user6

47 ls /home/user5

48 ls /home/user4

49 ls /home/user3

50 ls /home/user2

51 history

52 ls /home/

53 mkdir /home/user6 && touch /home/user6.txt

54 ls /home/

55 mkdir /home/user6 touch /home/user6.txt

56 ls /home/

57 clear

58 tree /home

59 cp /home/user1/task1/file.txt /tmp/1.txt

60 ls /tmp/

61 mv /home/user2/task2/file2.txt /tmp/

62 ls /tmp/

63 tree /home/

64 mv /tmp/1.txt /tmp/2.txt

65 ls /tmp/

66 rmdir --help

67 rmdir /home/user1

68 rmdir -p /home/user1

69 rmdir -- /home/user1

70 rm -rf /home/user1/

71 clear

72 history

73 history > history.txt