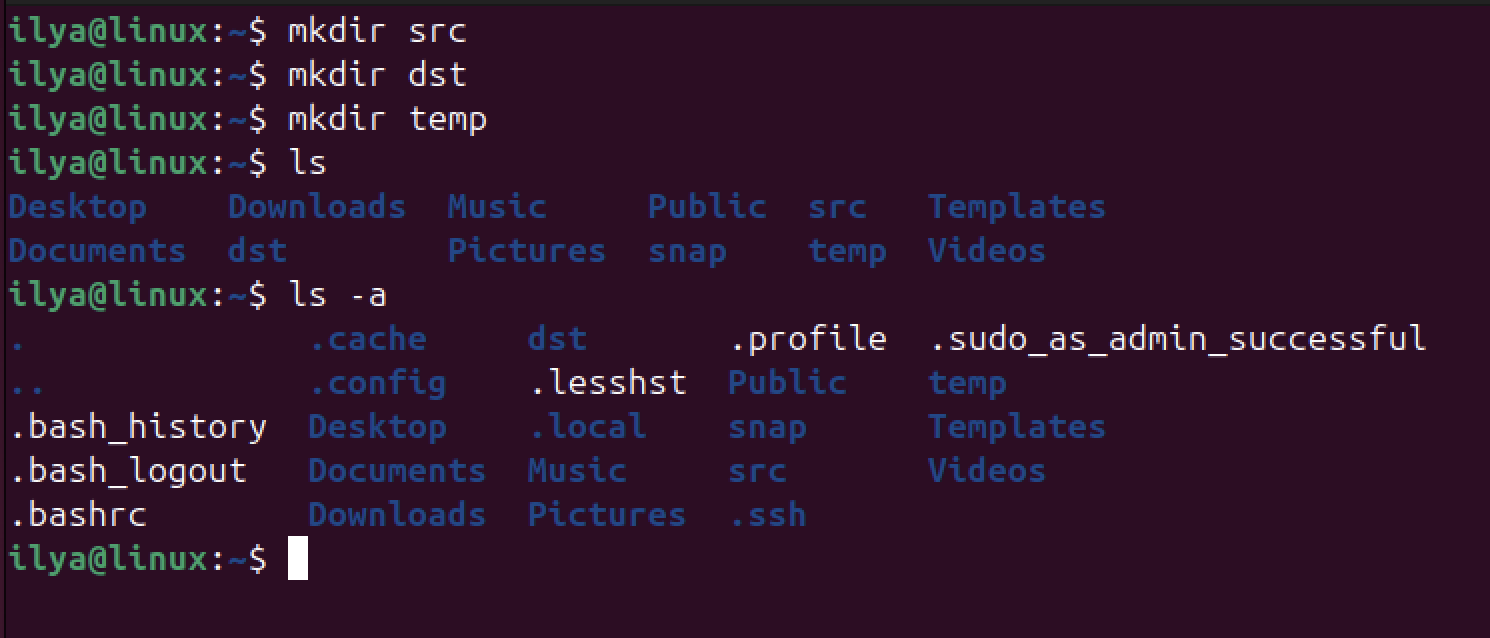
**Lx1.3. Работа с файлами.**

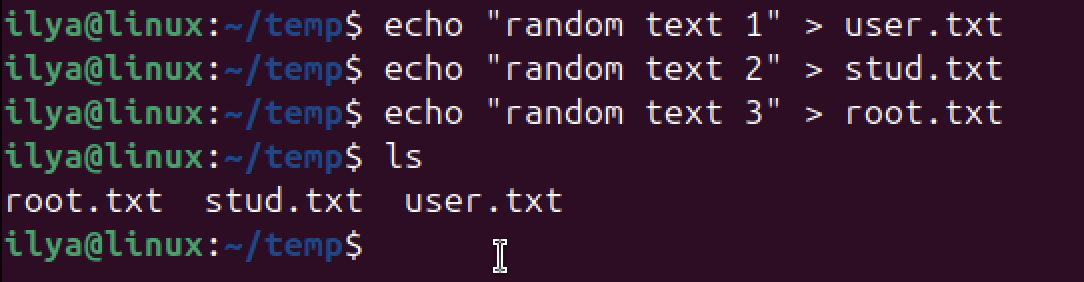
Наша главная цель семинара - ознакомиться со структурой файловой системы Линукс, типами файлов и командами управления файловой системы.

**Основное задание.**

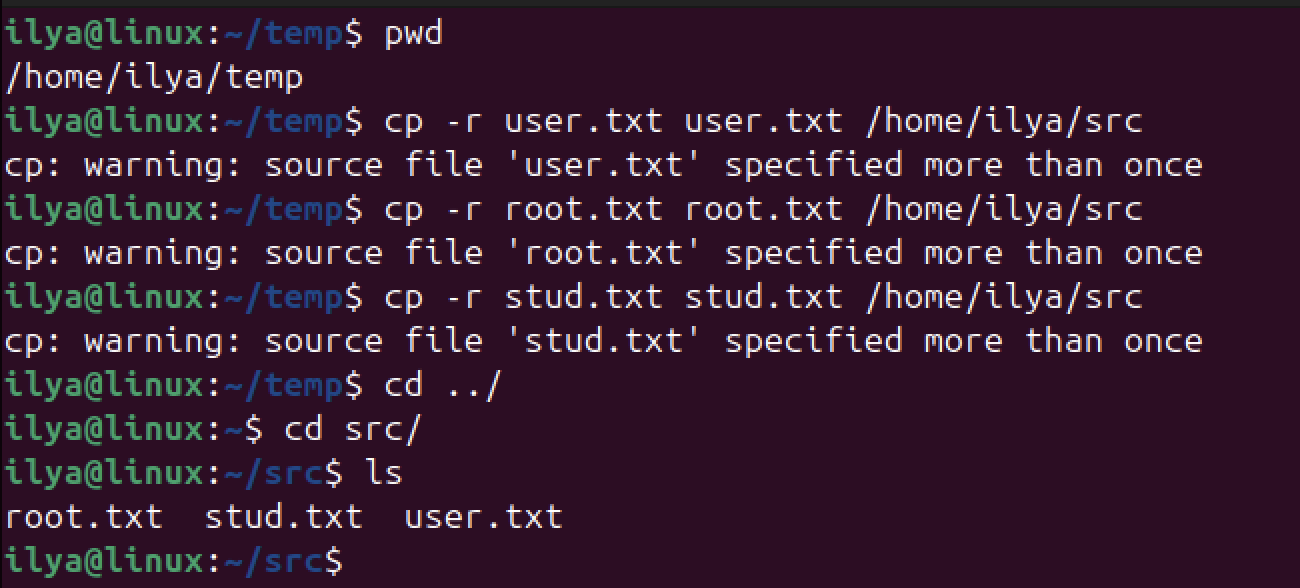
Для начала нам следует создать подкаталоги src, dst, temp. Используя команду mkdir сделаем это.



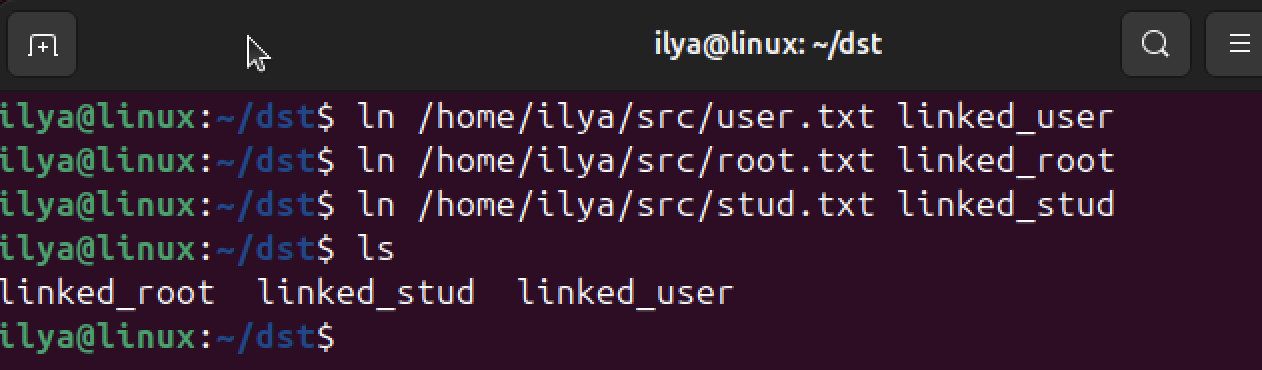
После успешного создания, переходим в каталог temp и в нем создаем файлы user.txt, root.txt, stud.txt. Команда - “touch имя файла”. И с помощью команды ls проверяем, созданные файлы.



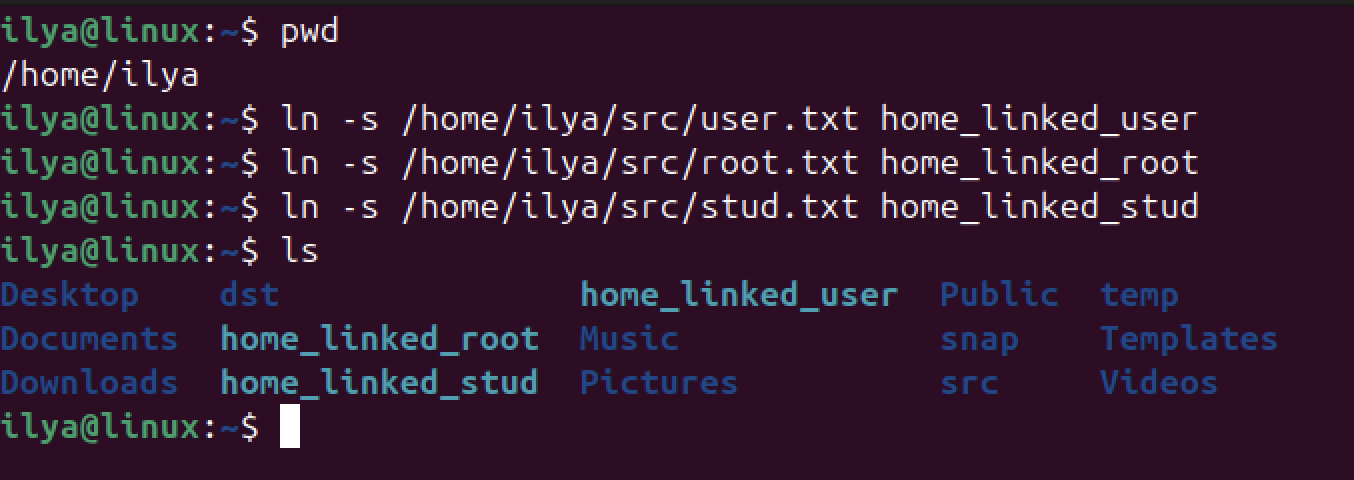
Следующим шагом нам следует скопировать только что созданные файлы в каталог src. Сделаем это с помощью команды “cp -r имя копируемого; файла имя нового файла; путь, куда записываем файл”. Названия файлов оставляем такие же. Затем перемещаемся в каталог src и с помощью команды ls просматриваем содержимое каталога.



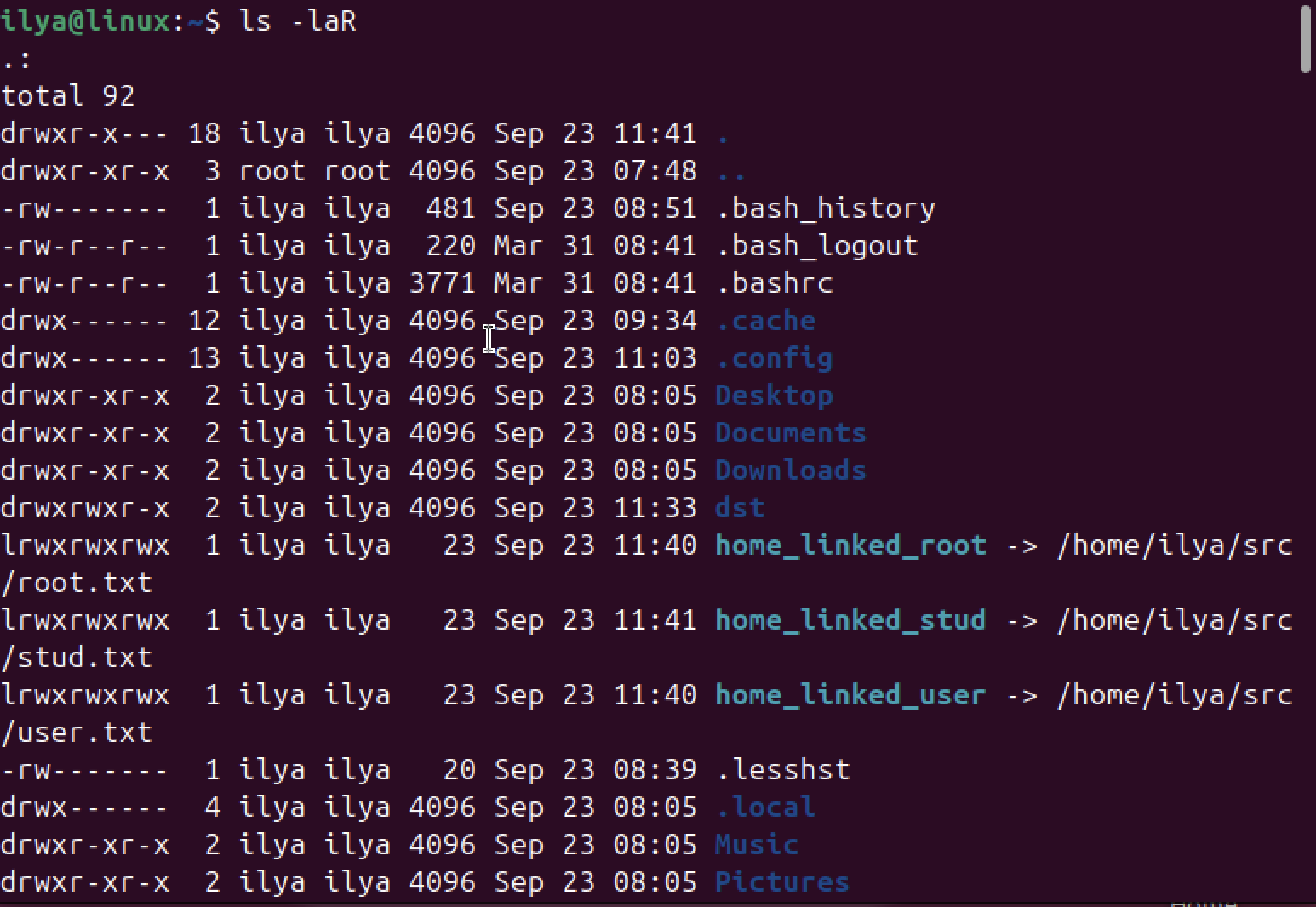
Далее нам требуется создать жестки ссылки в каталоге dts. Сначала перейдем в каталог dts, далее пропишем команду для создания жестких ссылок - “ln путь к файлу/имя файла новое имя файла”. Также с помощью ls сделаем проверку на наличие жестких ссылок. Как мы видим, все прошло успешно.



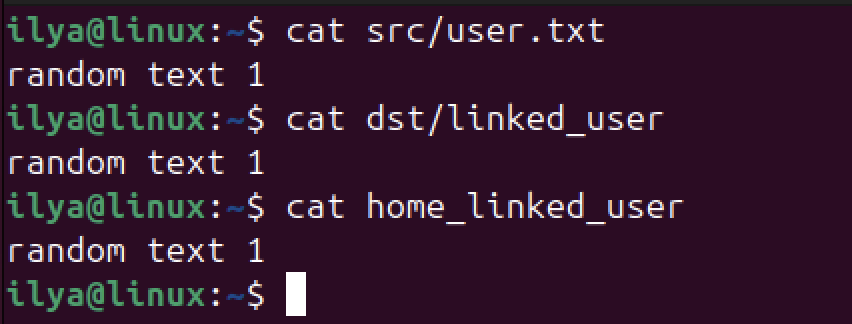
Теперь перейдем в домашний каталог и в нем нам потребуется создать мягкие ссылки на файлы, находящиеся в каталоге src. Отличие синтаксиса мягкой ссылки от жесткой в параметре -s, который идет после команды ln. Полная команда выглядит так: “ln -s путь до файла/имя файла новое имя файла”. При вводе команды ls мы видим, что обычные файлы у нас подсвечиваются синим цветом, а мягкие ссылки - бирюзовым.



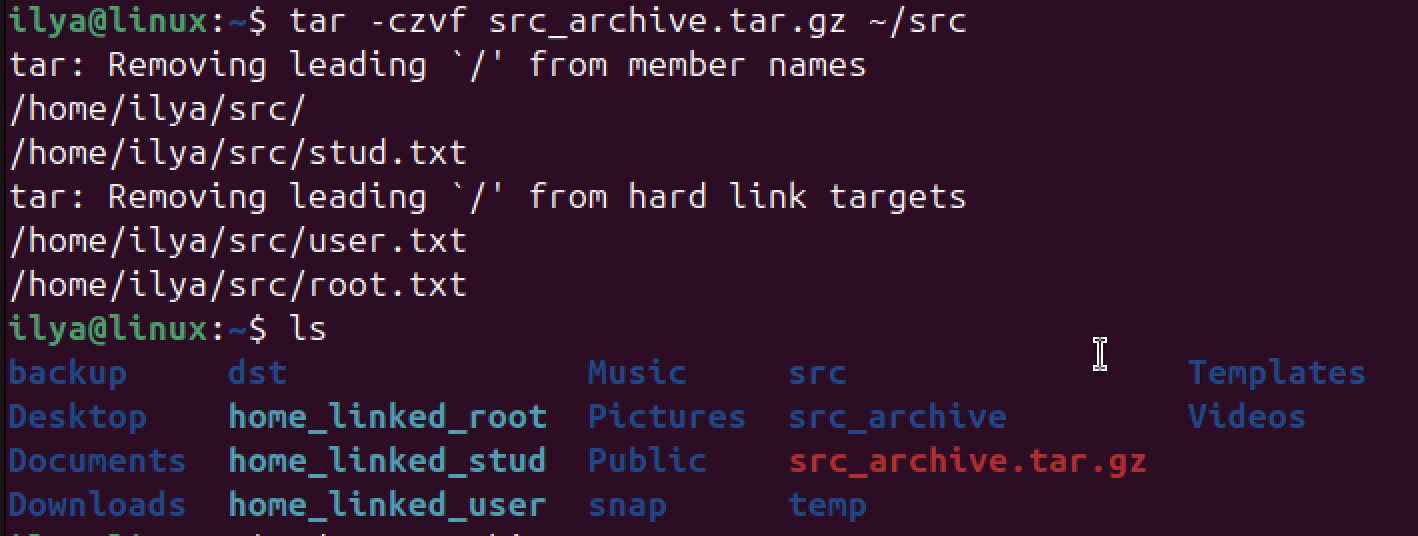
Теперь отобразим содержимое домашнего каталога, показывая рекурсивно расширенную информацию. Используем команду “ls -laR”, - где l отвечает за короткое отображение файлов, a - за подробную информацию о файле, R - за рекурсивный вывод. Следует заметить, что справа от названий ссылок указан путь, на который они ссылаются.



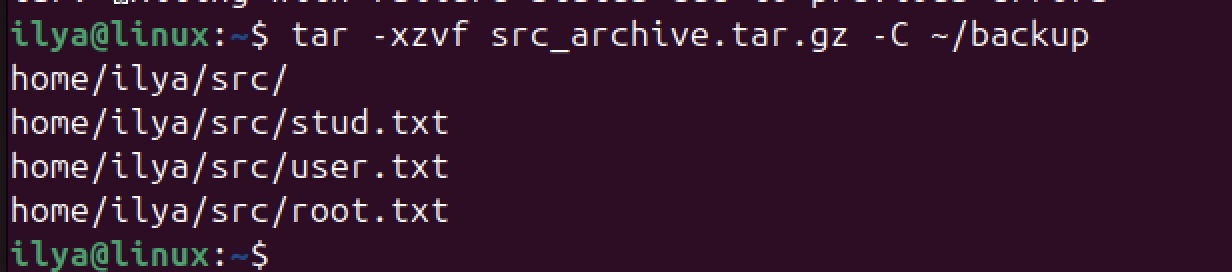
Теперь из домашнего каталога выполним команды: cat src/user.txt, cat dst/user.txt, cat user.txt. Как мы видим, при вызове каждой команды мы получаем одинаковое содержимое файла. Потому что в каталоге dst и home мы используем жесткую и мягкую ссылку соответственно.

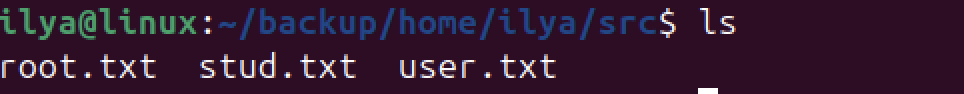


Для того, чтобы заархивировать содержимое каталога нам следует применить команду “tar -czvf имя архива.tar.gz -c путь до папки.”. Разберемся что за что отвечает. Tar - создает архив, -с - создает новый архив, z - сжимает архив с помощью gzip, v - выводит список файлов, которые мы будем записывать в архив, f - указывает имя файла архива.

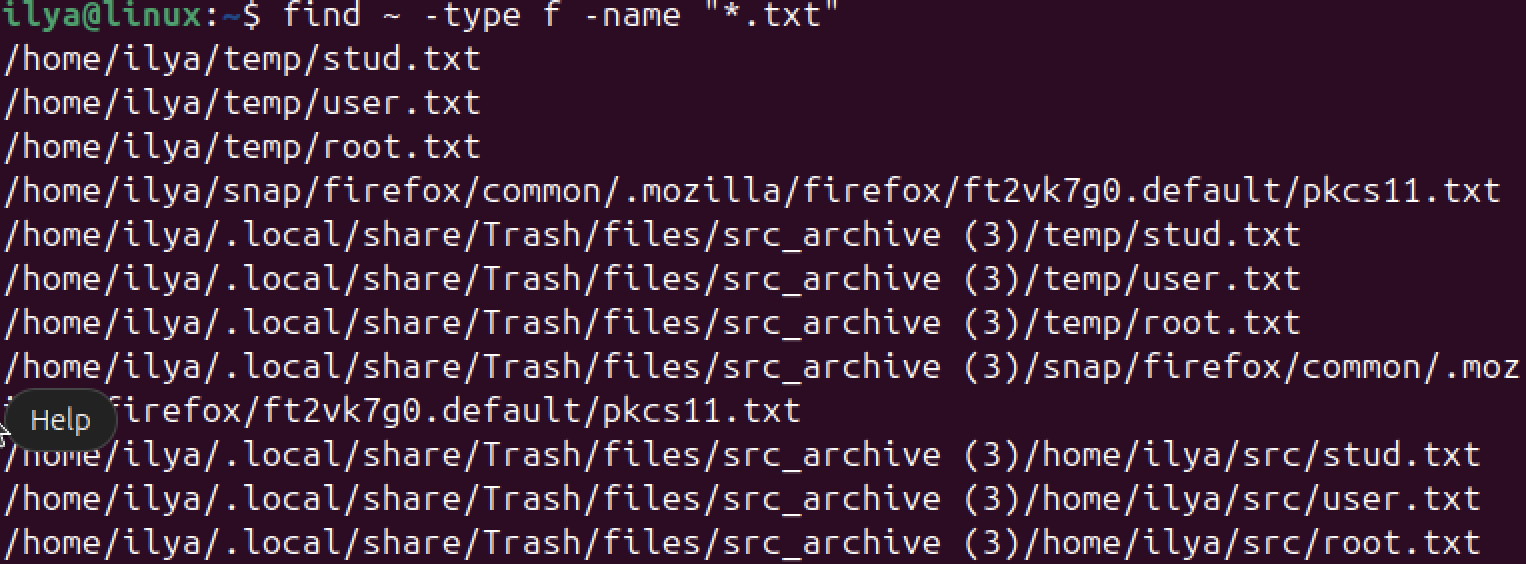


Теперь разархивируем наш архив в каталог backup, который мы создаем в домашней директории. Команда для разархивирования схожа с прошлой, только отличие в первом параметре - здесь это -x - указывает на необходимость извлечь файлы.

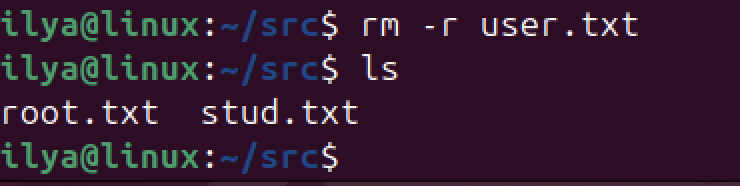


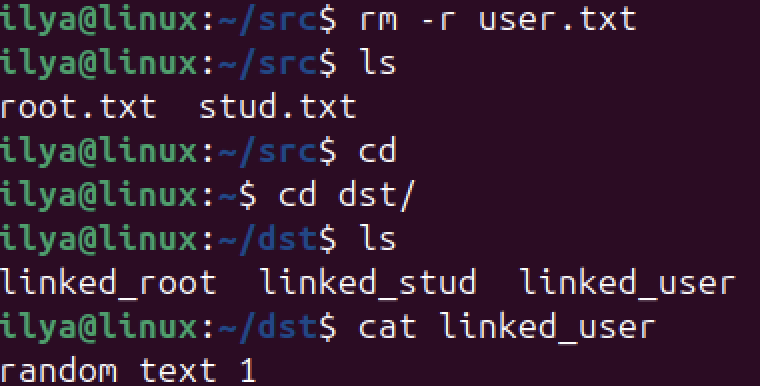


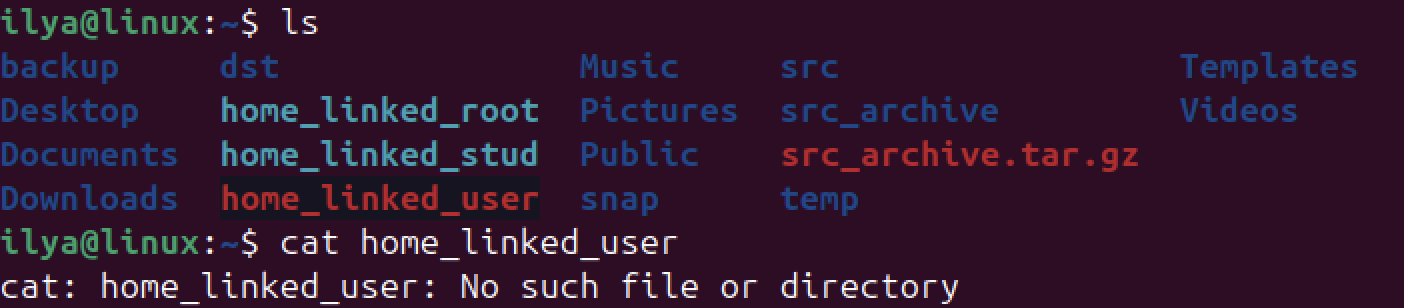
Далее нам нужно отобразить все файлы в домашнем каталоге, для этого используем команду “find ~ -type f -name ”\*.txt” ”. Здесь ~ - поиск по всему домашнему каталогу, -type f - отбираем только тип: файл, -name - отбираем все файлы с расширением .txt.



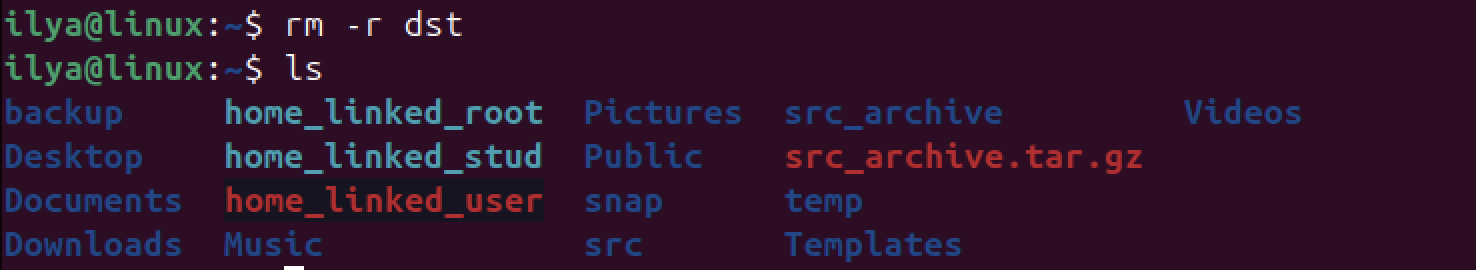
По заданию удаляем файл user.txt, и мы видим, что с удалением файла с каталога temp, жесткие ссылки на документ остались и содержимое фала остается. Пока последний экземпляр файла не будет удален, жесткая ссылка будет работать. А если обратить внимание на мягкие ссылки, то она при выводе команды ls начала подсвечиваться красным и при обращении к ней с помощью команды cat home\_linked\_user и получим, что такой файл отсутствует.



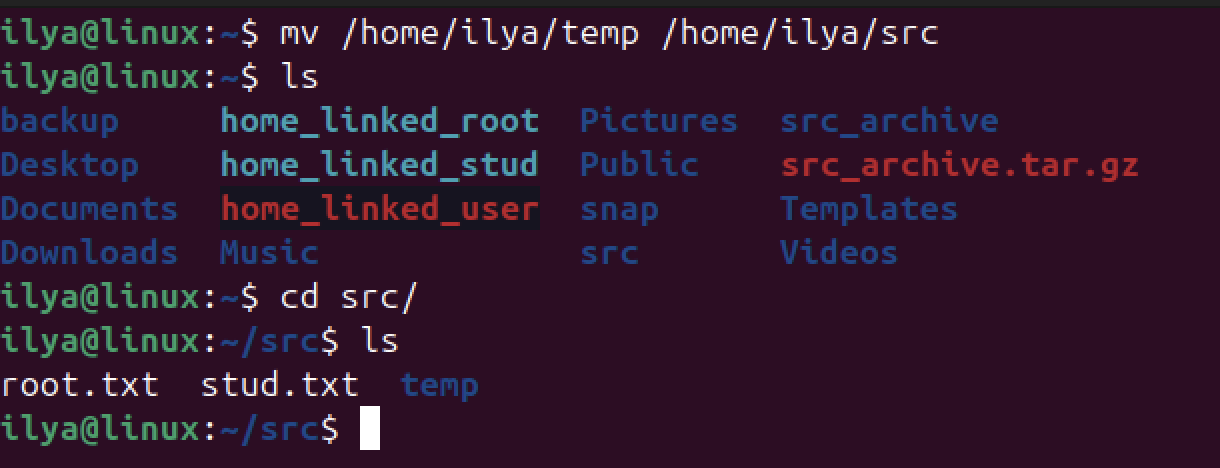




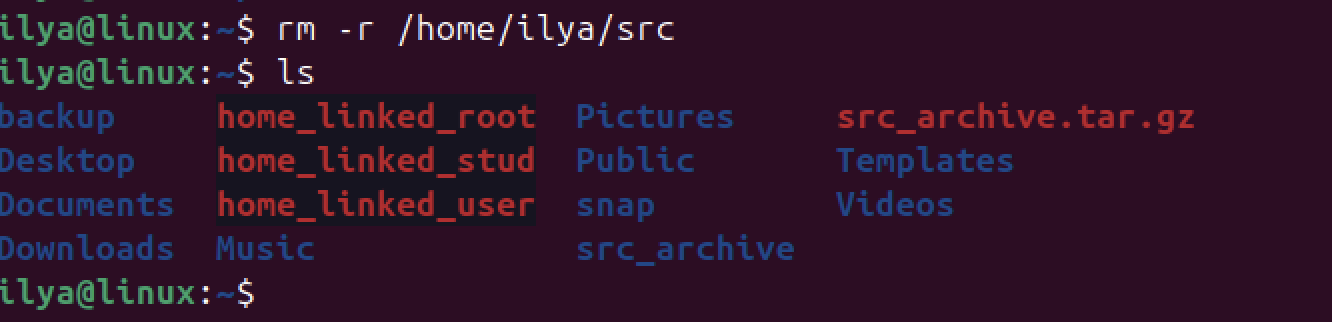
По заданию удаляем каталог dst.



Далее переместим каталог temp в каталог src. Используем команду “mv путь до каталога temp путь до каталога src”



Рекурсивно удаляем каталог src - “rm -r /home/ilya/src”



**Контрольные вопросы.**

1. Чем отличаются жесткие и символические ссылки?

Главное отличие жестких от символических в том, что при удалении оригинального файла, данные все равно остаются доступными. А у символических - ссылка на удаленный оригинальный файл становиться недействительной. Также есть отличие в синтаксисе команд. Для жестких это просто ln, а для символичных - добавляется параметр -s. Символические ссылку могут вести на целую директорию, а жесткие ссылки - только на файлы.

1. Что произойдет с жесткой ссылкой, если удалить исходный файл? А с символической?

Если остался еще один экземпляр удаленного файла, то жесткая ссылка продолжит существовать и выводить данные файла, а символическая перестанет работать.

1. Совпадает ли размер символьных и жестких ссылок?

Нет, не совпадает, так как жесткие ссылки имеют такой же размер, как и оригинальный файл, фактически размер жесткой ссылке равен размеру данных, на которые она ссылается. А размер символьной ссылки зависит от длины пути к файлу, они представляют собой отдельные файлы.

1. Пусть в системе имеется каталог “~/test”. Что произойдет при попытке выполнить следующую команду: “cp ~/test ~/test/newdir”?

Произойдет ошибка, так как мы не указали параметр для копирования - “-r”, для рекурсивного копирования. И также мы хотим скопировать несуществующий каталог newdir, поэтому выдаст еще одну ошибку.

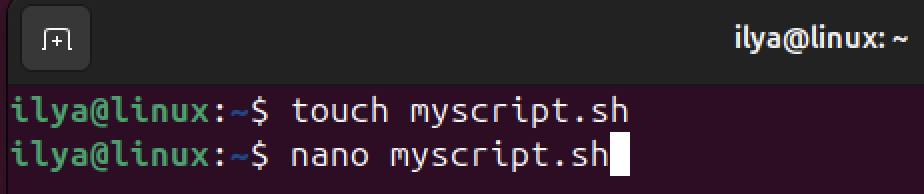
1. Какие основные форматы архивов принято использовать в Linux? Какими командами можно пользоваться для создания/открытия этих архивов?

К основным форматам можно отнести те, с которыми мы работали: tar и gzip. Для создания архива нужно использовать параметр “-c”, а для чтения из архива - “-x”.

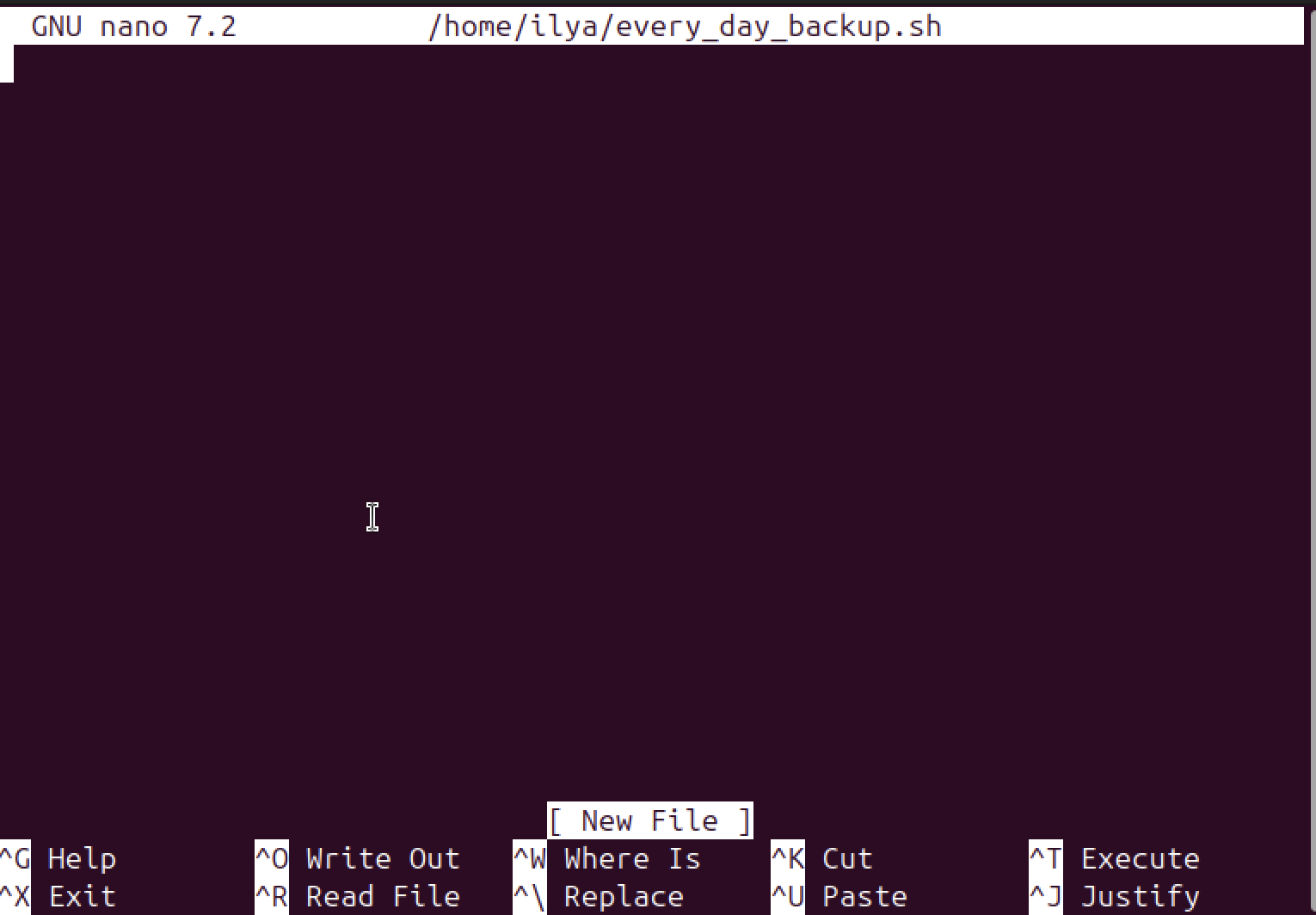
**Задания для самостоятельного выполнения.**

Требуется настроить автоматическую инкрементальную архивацию домашнего каталога. Для этого создаем команду архивирования всех файлов, измененных за последние сутки, и добавим ее на автоматическое выполнение при помощи службы cron.

В первую очередь, для создания скрипта, напишем команду “touch имя\_срипта” с расширением sh (файл, написанный на языке bash) и используем команду nano, написав в ней имя нашего скрипта.

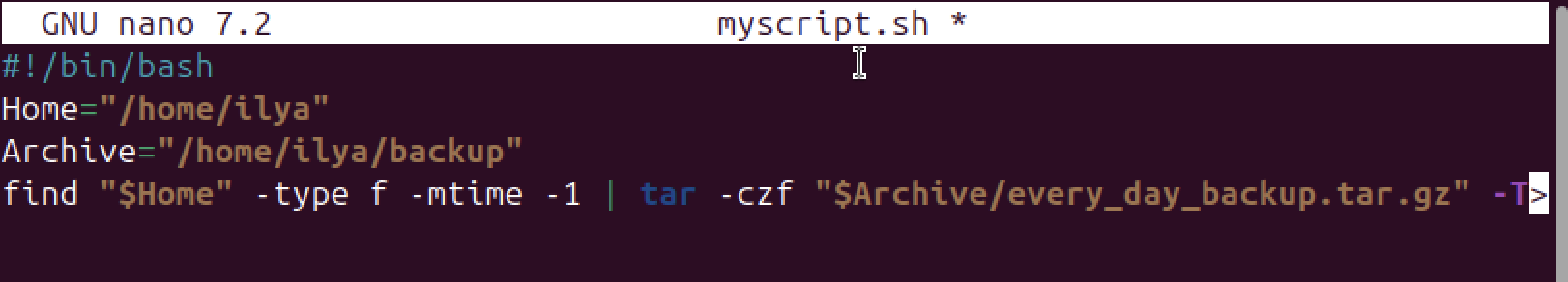


Откроется следующее окно:

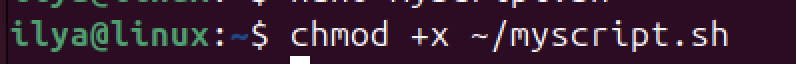


Открывается окно для написания скрипта. В первой строке, открытого окна, мы пропишем последовательность символов, которая называется shebang - такая последовательность служит для определения интерпретатора, для Bash используется следующая: #!/bin/bash.

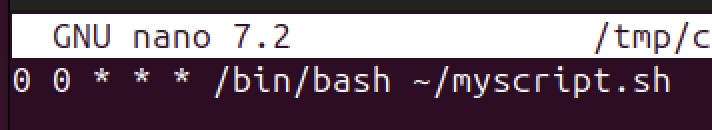
Далее поэтапно пропишем команды для выполнения нашего скрипта. Для начала укажем в переменной Home путь к домашнему каталогу, с которого будут архивироваться данные. В следующей строке, в переменной Archive, пропишем путь до каталога, в котором будет храниться наш архив. Теперь с помощью команды find будем искать все файлы в домашнем каталоге, измененные за последние сутки. Аргумент “|” используется для комбинирования двух команд, для удобства записи. Далее через прямую черту пропишем команду для архивации, с помощью tar и укажем параметры -czf (-c создание архива, z - производит сжатие с помощью gzip, f - отображает архивированные файлы). В конце указываем -T - для динамической передачи списка файлов в команду “tar”.



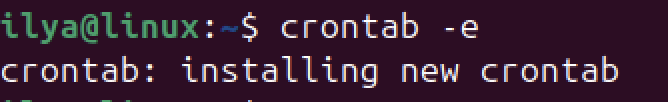
После, мы выходим из редактора скрипта, и вводим команду для активации нашего скрипта. Сделаем это с помощью команды chmod, тем самым добавляя право выполнения для владельца файла.



Теперь, используя cron, как сказано в задании, добавим этот скрипт в cron для автоматического выполнения. Пропишем команду “crontab -e”, в которой crontab, при первом открытии будет отвечать за создание нового файла в cron. -e отвечает за редактирование файла.



Первый и второй нули отвечают за выполнение команды ежедневно в 0 минут 0 часов по установленному времени на устройстве. /bin/bash - здесь также является интерпретатором, для запуска скрипта. И в конце у нас указывается имя скрипта, который мы хотим выполнить.



Мы получили уведомление об установке нового скрипта в cron.