Отчет о прохождении внешних курсов

3 этап

Кижваткина Анна Юрьевна

Содержание

# 1 Цель работы

Ознакомиться с функционалом операционной системы Linux.

# 2 Задание

Просмотреть видео и на основе полученной информации пройти тестовые задания.

# 3 Теоретическое введение

Линукс - в части случаев GNU/Linux — семейство Unix-подобных операционных систем на базе ядра Linux, включающих тот или иной набор утилит и программ проекта GNU, и, возможно, другие компоненты. Как и ядро Linux, системы на его основе, как правило, создаются и распространяются в соответствии с моделью разработки свободного и открытого программного обеспечения. Linux-системы распространяются в основном бесплатно в виде различных дистрибутивов — в форме, готовой для установки и удобной для сопровождения и обновлений, — и имеющих свой набор системных и прикладных компонентов, как свободных, так и проприетарных.

# 4 Выполнение лабораторной работы

3 Этап: (рис. fig. 1, fig. 2, fig. 3, fig. 4, fig. 5, fig. 6, fig. 7, fig. **¿fig:008?**, fig. 8, fig. 9, fig. 10, fig. 11, fig. 12, fig. 13, fig. 14, fig. 15, fig. 16, fig. **¿fig:018?**, fig. 17, fig. 18, fig. 19, fig. 20, fig. 21, fig. 22, fig. 23, fig. 24, fig. 25, fig. 26, fig. 27, fig. 28, fig. 29, fig. 30, fig. 31, fig. 32, fig. 33, fig. 34, fig. **¿fig:037?**, fig. 35, fig. 36, fig. 37, fig. 38).



Рис. 1: Задание 1

Стоит упомянуть, что у редактора vim есть туториал, который позволяет разобраться с командами, необходимыми для стандартной работы. За выход из редактора отвечают следующие команды:

* ZQ - выйти без сохранения
* :q! - выйти без сохранения
* ZZ - записать файл и выйти (если файл не изменяли, то записываться он не будет)
* :wq - записать файл и выйти
* :x - записать файл и выйти
* :w - записать файл
* :sav filename - “сохранить как”
* :w filename - “сохранить как”
* :w! - записать файл

Как мы видим, вариантов много, при этом каждый сможет найти тот, который подойдёт под конкретную ситуацию.

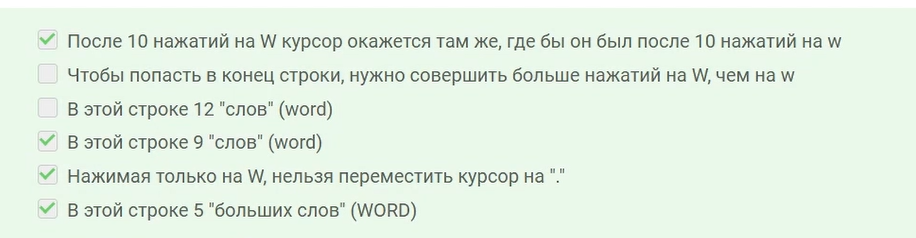


Рис. 2: Задание 2

Strange\_ TEXT is\_here. 2=2 YES!

Точка считается “маленьким словом”, так что всего их 9: Strange\_, is\_here, ., 2, =, 2, ! и два лишних пробела.

И если посчитать нажатия на w и на W, то действительно после 10 штук попадем в одно место. 10 нажатий на W, это то же самое, что и 10 нажатий на w,

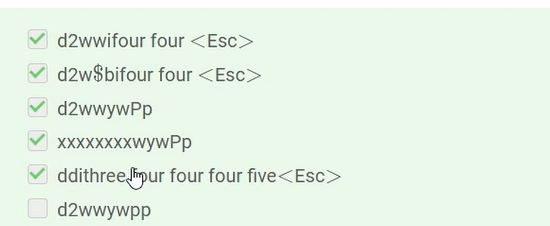


Рис. 3: Задание 3

* $ — в конец текущей строки;
* w — на слово вправо;
* b — на слово влево;
* i — начать ввод перед курсором;
* p — вставка содержимого неименнованного буфера под курсором;
* P — вставка содержимого неименованного буфера перед курсором;
* yy (также Y) — копирование текущей строки в неименованный буфер;
* yy — копирование числа строк начиная с текущей в неименованный буфер;



Рис. 4: Задание 4

Поиск и замена в редакторе работают по следующей схеме:

:{пределы}s/{что заменяем}/{на что заменяем}/{опции}

Для замены во всем файле можно использовать символ %.

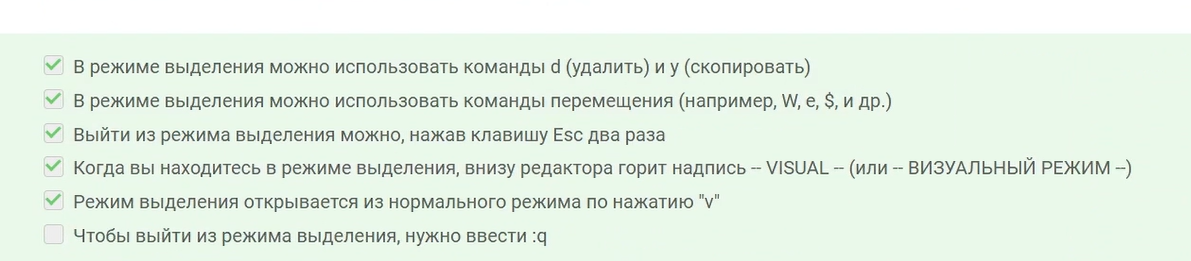


Рис. 5: Задание 5

Команда $ — в конец текущей строки, W - до пробела вправо - то есть, перемещение.

Нажать Esc достаточно один раз, но да ладно.

d — используется совместно с командами перемещения. Удаляет символы с текущего положения курсора до положения после ввода команды перемещения.

yy (также Y) — копирование текущей строки в буфер;

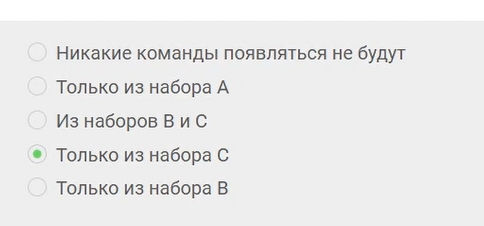


Рис. 6: Задание 6

Только из набора С потому что у каждой оболочки свой буфер, который при выходе из нее буде записываться в файл истории.

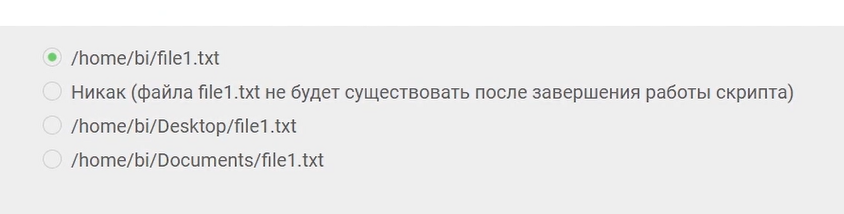


Рис. 7: Задание 7

/home/bi/file1.txt - потому что именно в этой директории мы создаем новый файл, а уже после его создания мы переходим в другую папку. 

Имя не может начинаться с цифры, содержать специальные символы или пробелы.



Рис. 8: Задание 9

$ echo опции строка Эта команда печатает строки, которые передаются в качестве аргументов в стандартный вывод и обычно используется в сценариях оболочки для отображения сообщения или вывода результатов других команд.

var1=$1 - обозначение переменных

var2=$2

echo “Arguments are: $1=var2” - строка печати.

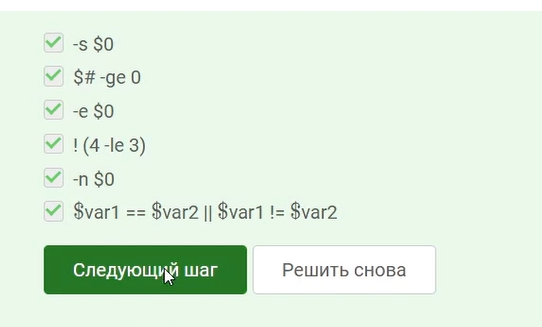


Рис. 9: Задание 10

* $0 - имя скрипта
* $# - вернет количество аргументов
* -ge - больше или равно
* -n - не пустая строка.

$# Это число аргументов без учета имени скрипта, который всегда $0. И число аргументов всегда будет или равно нулю, или больше него, тк просто не может скатиться в отрицательную сторону.



Рис. 10: Задание 11

* -lt, (<) - меньше
* -gt - больше
* -eq – равно

Оба раза выведет four.

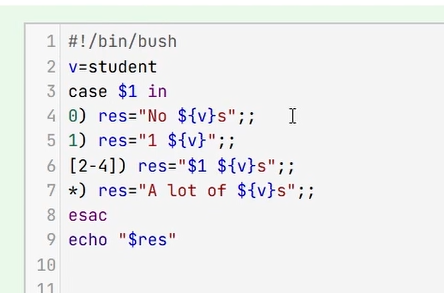


Рис. 11: Задание 12

1. Задаю общую часть в каждом выводе - слово “student”: v=student
2. Выполняем команды для разных аргументов.
3. res - это результат для вывода
4. echo “$res” - вывести результат

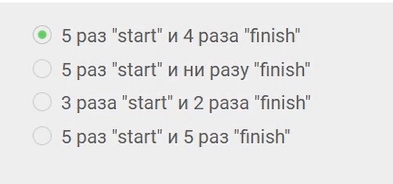


Рис. 12: Задание 13

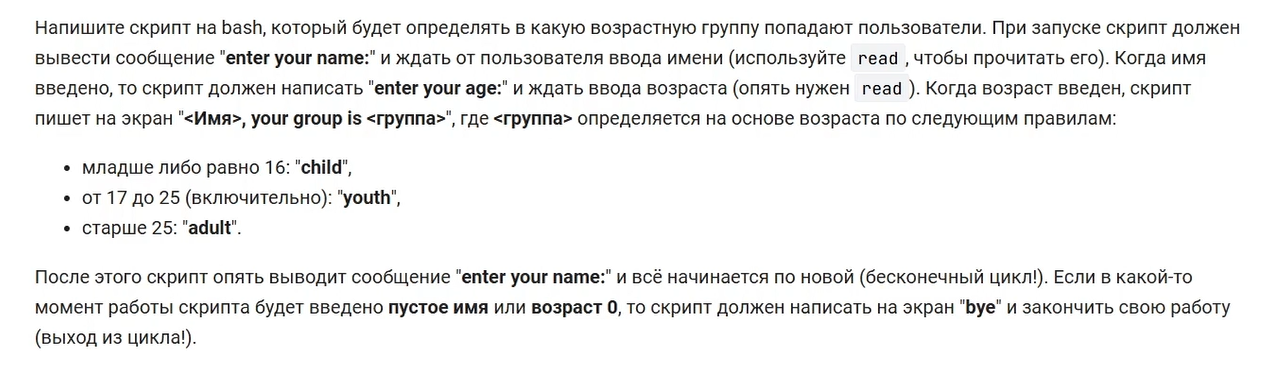


Рис. 13: Задание 14



Рис. 14: Задание 14

Пишем код исходя из условий задания



Рис. 15: Задание 15

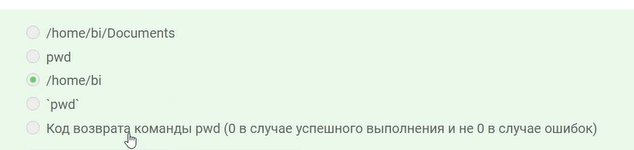
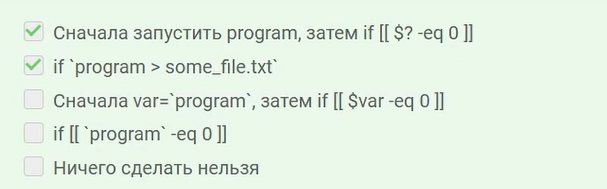


Рис. 16: Задание 16

Выведет путь до директории, в которую мы перешли, так как ”

pwd

” - это команда 

programm выполняет стандартный вывод в терминал (если это принцип работы программы). И нам нужно настроить вывод в файл.

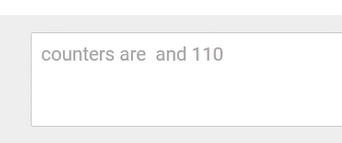


Рис. 17: Задание 17

Первая переменная локальная, и это просто пустая строка, вторая переменная - это сумма арифметической прогрессии от 1 до 10, равна 55, но при умножении на 2 даст 110.

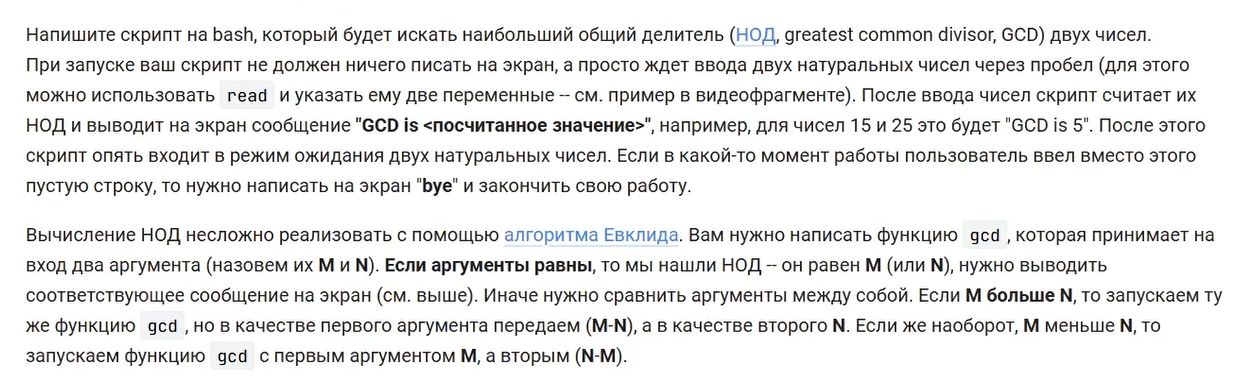


Рис. 18: Задание 18

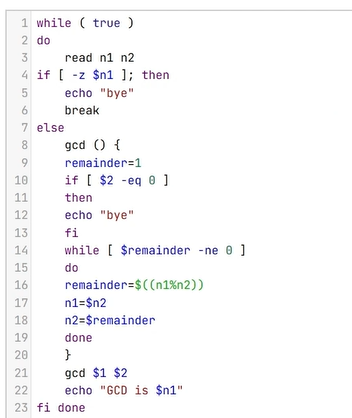


Рис. 19: Задание 18

Алгоритм нахождения НОД делением



Рис. 20: Задание 19



Рис. 21: Задание 19

Калькулятор выглядит обычно - мы вводим два числа, пишем, что с ними надо сделать, и потом, учитывая случаи ошибок, выводим результат.

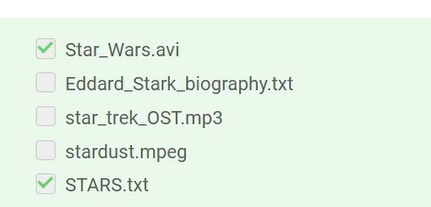


Рис. 22: Задание 20

-iname ищет без учета регистра, а -name в точности как в запросе. Звездочка стоит после слова - это значит после слова может быть сколько угодно символов.

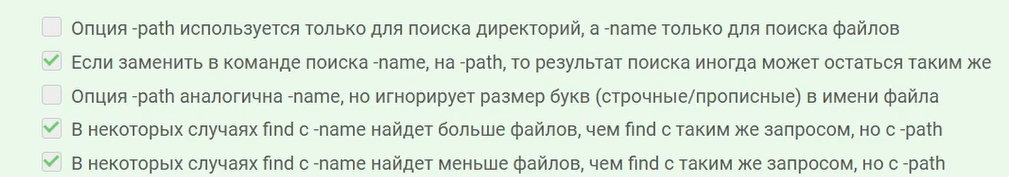


Рис. 23: Задание 21

find [path] [expression]

где: path - это путь к директории, в которой нужно выполнить поиск файлов (по умолчанию, поиск производится в текущей директории и всех ее поддиректориях);

expression - это выражение, которое определяет критерии поиска файлов.

-name: поиск файлов по имени. Например: find /home/user -name myfile.txt

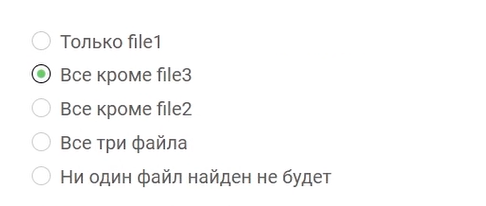


Рис. 24: Задание 22

Текущий каталог - это depth=1, а остальное считается просто:

/home/bi -> depth=1

/home/bi/dir1 -> depth=2

/home/bi/dir1/dir2 -> depth=3

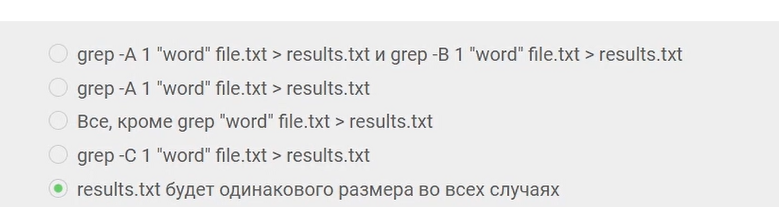


Рис. 25: Задание 23

Из описания man: Print NUM lines of trailing context after/before matching lines “matching lines” - множественное число, строки в которых нашлось совпадение

Т.е. если идут 2…10…100 строк подряд, в которых обнаружилось совпадение, контекст будет выведен до и после этой ГРУППЫ строк, а не до и после каждой строки в этой группе

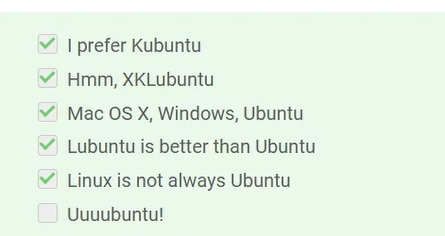


Рис. 26: Задание 24

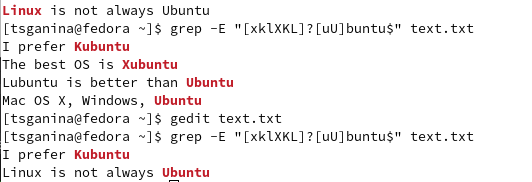


Рис. 27: Задание 24

Объяснение на втором скриншоте.

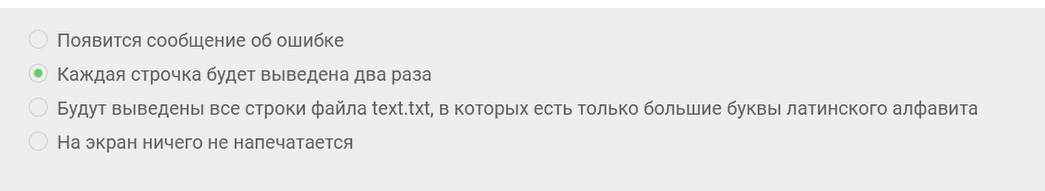


Рис. 28: Задание 25

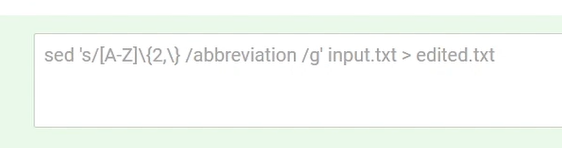


Рис. 29: Задание 26

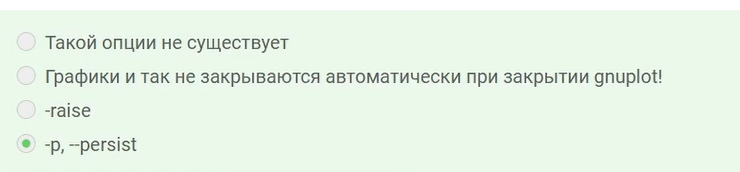


Рис. 30: Задание 27

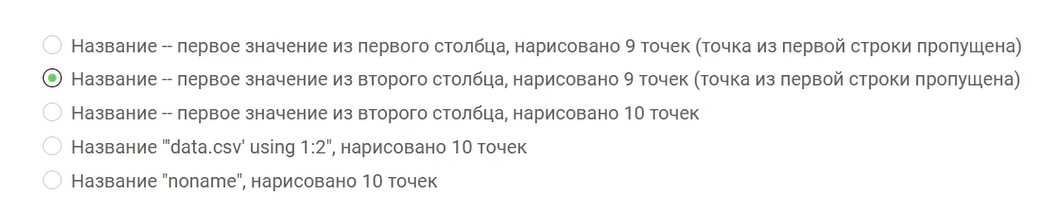


Рис. 31: Задание 28



Рис. 32: Задание 29

Cначала идет команда установки подписей, а потом в скобках:

подпись - пробел - переменная с координатой - запятая

Повторяется это количество раз соответствующее числу переменных, и без запятой (в случае с последней переменной)

А подпись в свою очередь получается конкатенацией текста из задания и переменной с координатой.

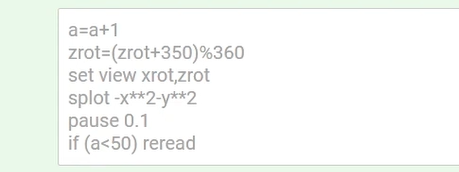


Рис. 33: Задание 30

1. График строится строкой “splot x2+y2”.
2. Вращение задается строкой “zrot=(zrot+10)%360”. Значит, смещение вперед (которое было изначально) можно также задать строкой “zrot=(zrot+360+10)%360” или иначе говоря “zrot=(zrot+370)%360”. А теперь посмотрим на наше требование - чтоб вращалось в другую сторону, значит, по аналогии, необходимо вместо перебора на 10 сделать недобор.

“zrot=(zrot+350)%360”

1. Строка “pause 0.2” ставит выполнение на паузу на определенный промежуток времени. В задании сказали перерисовывать чаще, значит пауза должна быть меньше.

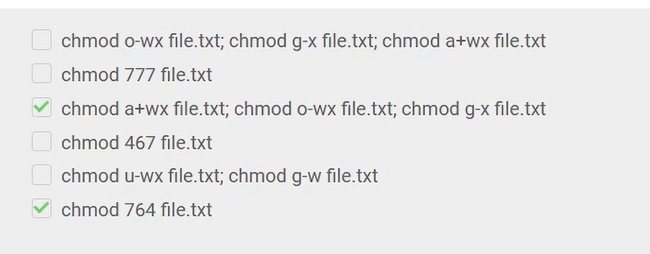
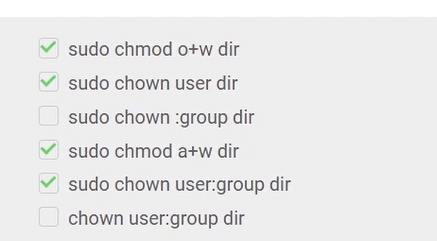


Рис. 34: Задание 31

* r - чтение;
* w - запись;
* x - выполнение;
* s - выполнение от имени суперпользователя (дополнительный);
* u - владелец файла;
* g - группа файла;
* o - все остальные пользователи;
* 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. 

Решений два типа:

* Сменить права гостей, добавив W
* Сделать владельцем нужную группу или пользователя, в зависимости от того, у кого из них уже есть права на W
* Помнить, что root - владелец и остальные для него - others.

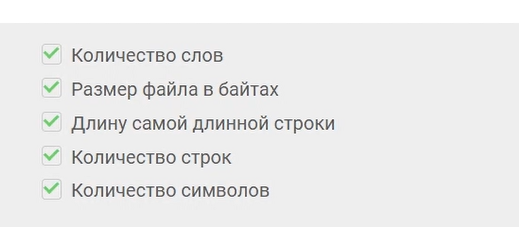


Рис. 35: Задание 33

* wc -l вывести количество строк
* wc -c вывести количество байт
* wc -m вывести количество символов
* wc -L вывести длину самой длинной строки
* wc -w вывести количество слов

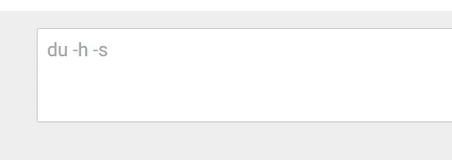


Рис. 36: Задание 34

-h, –human-readable print sizes in human readable format (e.g., 1K 234M 2G)

-s, –summarize display only a total for each argument



Рис. 37: Задание 35

Команда создаст три директории от dir1 до dir3.

# 5 Сертификат

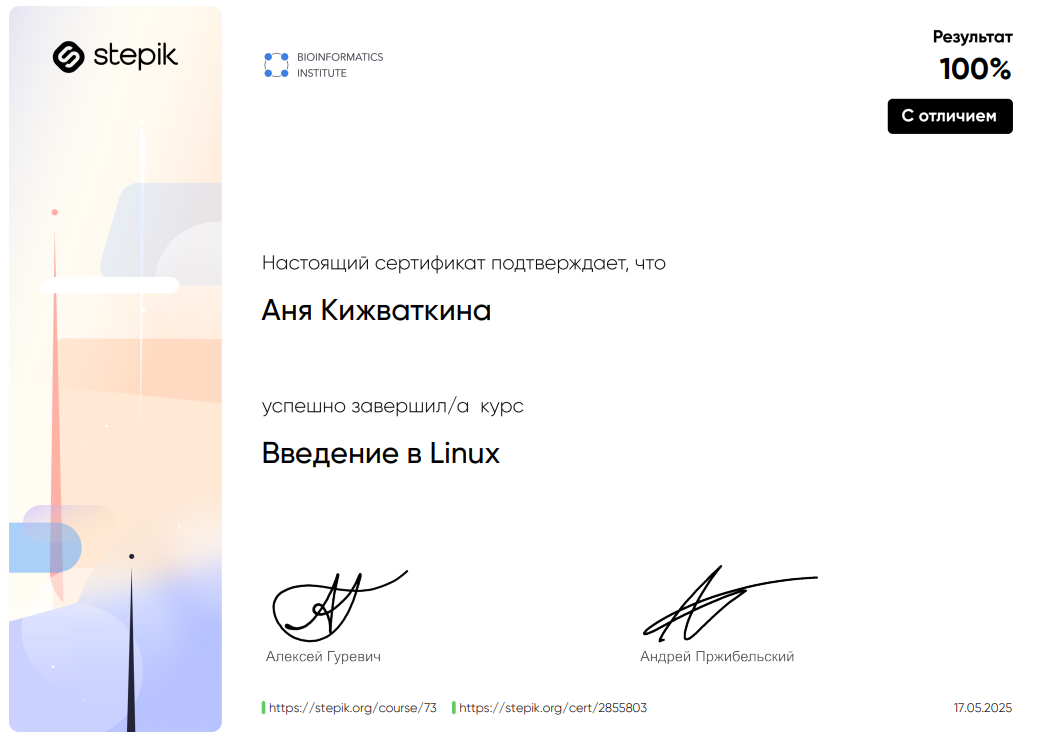


Рис. 38: Сертификат

# 6 Выводы

Я просмотрела курс и освежила в памяти навыки работы с более сложными командами в Линукс.