Отчет о прохождении 3 этапа внешних курсов

Продвинутые темы

Овезов Мерген, НБИбд-02-23

Содержание

[1 Цель работы 1](#_Toc166944087)

[2 Задание 1](#_Toc166944088)

[3 Теоретическое введение 1](#_Toc166944089)

[4 Выполнение лабораторной работы 1](#_Toc166944090)

[5 Сертификат 23](#_Toc166944091)

[6 Выводы 24](#_Toc166944092)

[Список литературы 24](#_Toc166944093)

# 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. [1](#fig:001), [2](#fig:002), [3](#fig:003), [4](#fig:004), [5](#fig:005), [6](#fig:006), [7](#fig:007), [8](#fig:008), [9](#fig:009), [10](#fig:010), [11](#fig:011), [12](#fig:012), [13](#fig:013), [14](#fig:014), [15](#fig:015), [16](#fig:016), [17](#fig:017), [18](#fig:018), [19](#fig:019), [20](#fig:020), [21](#fig:021), [22](#fig:022), [23](#fig:023), [24](#fig:024), [25](#fig:025), [26](#fig:026), [27](#fig:027), [28](#fig:028), [29](#fig:029), [30](#fig:030), [31](#fig:031), [32](#fig:032), [33](#fig:033), [34](#fig:034), [35](#fig:035), [36](#fig:036), [37](#fig:037), [38](#fig:038), [39](#fig:039), [40](#fig:040), [41](#fig:041)).

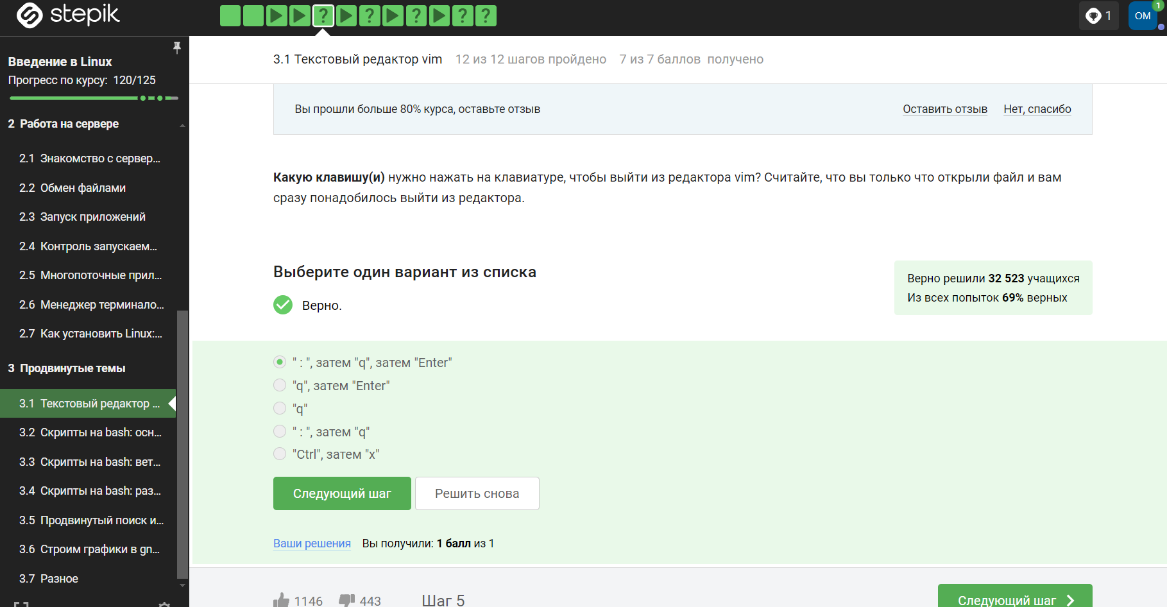


Figure 1: Задание 1

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.

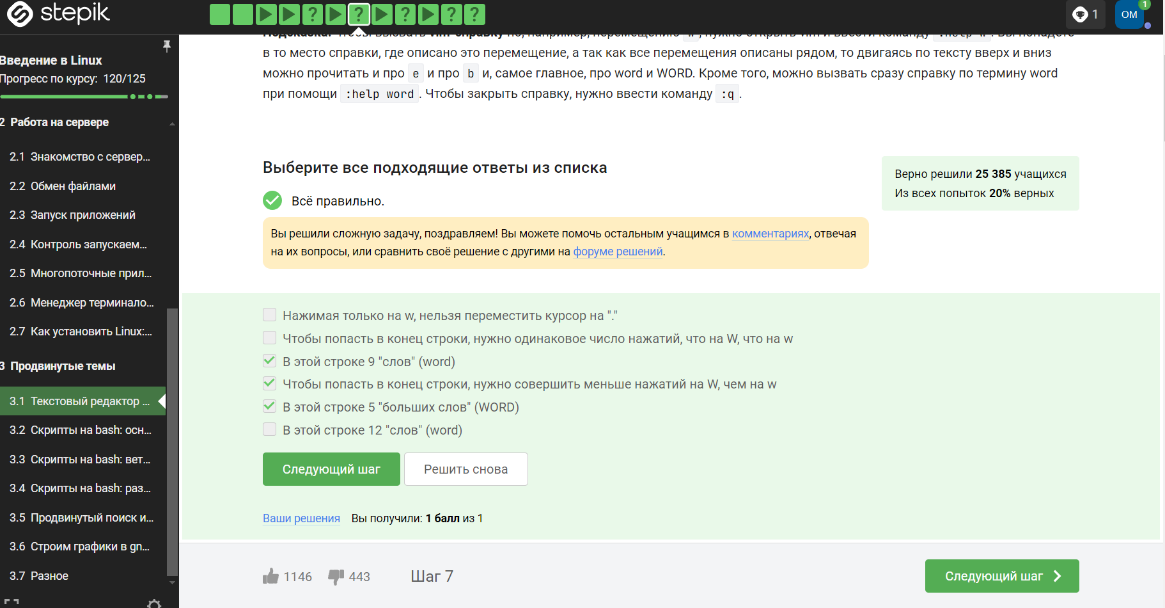


Figure 2: Задание 2

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,

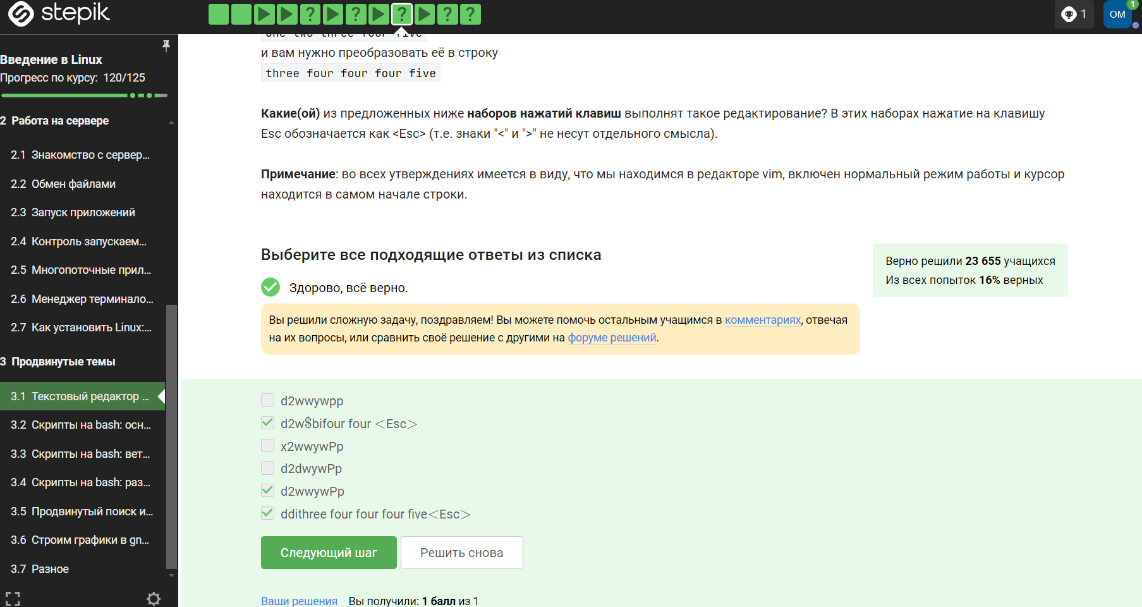


Figure 3: Задание 3

Ddithree four four four five<Esc>

d2wwywPp

d2w$bifour four <Esc>

* $ — 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;

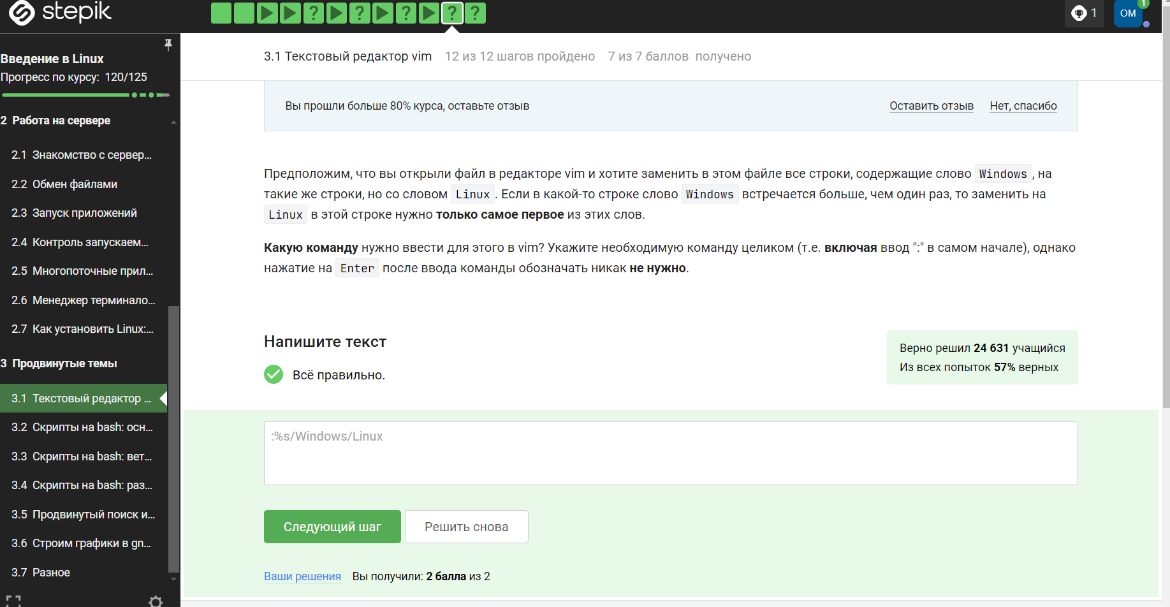


Figure 4: Задание 4

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 %.

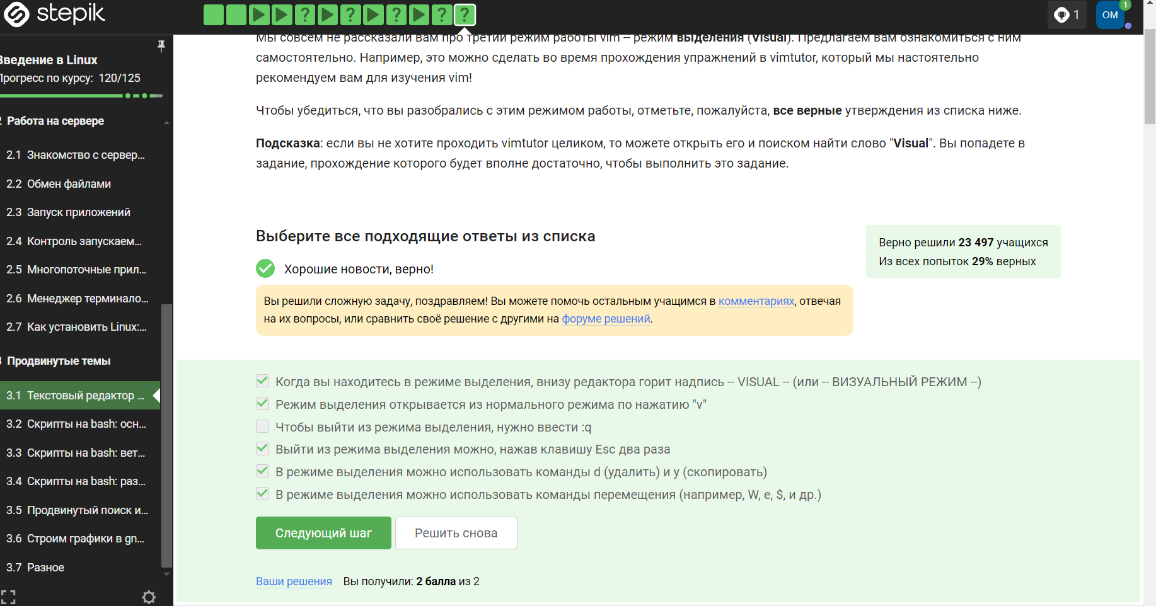


Figure 5: Задание 5

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

Нажать Esc достаточно один раз, но да ладно.

Надпись visual - горит.

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;

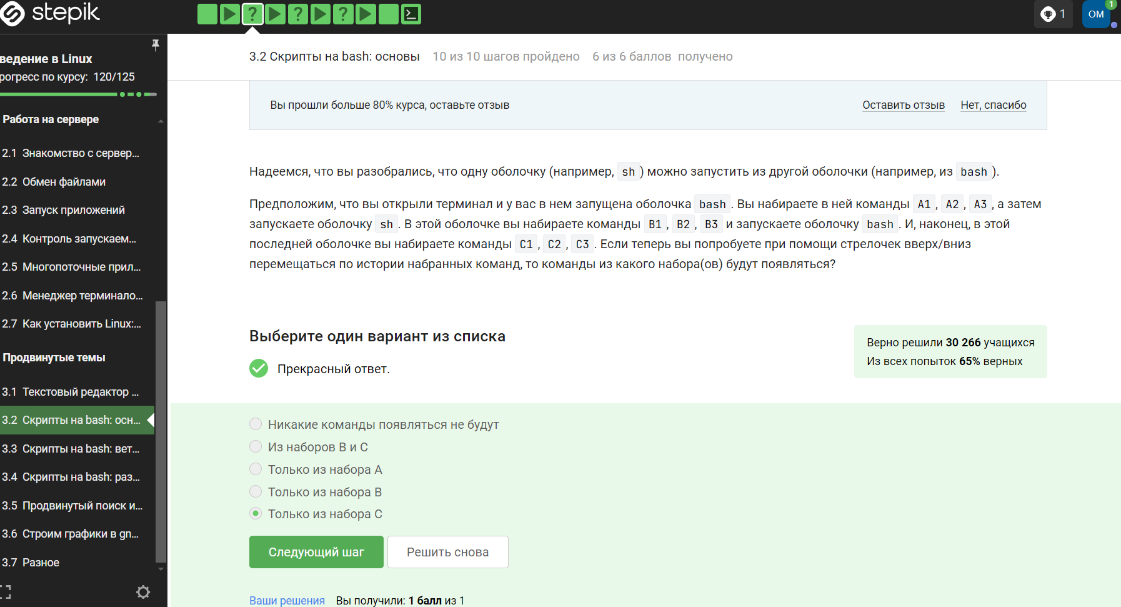


Figure 6: Задание 6

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

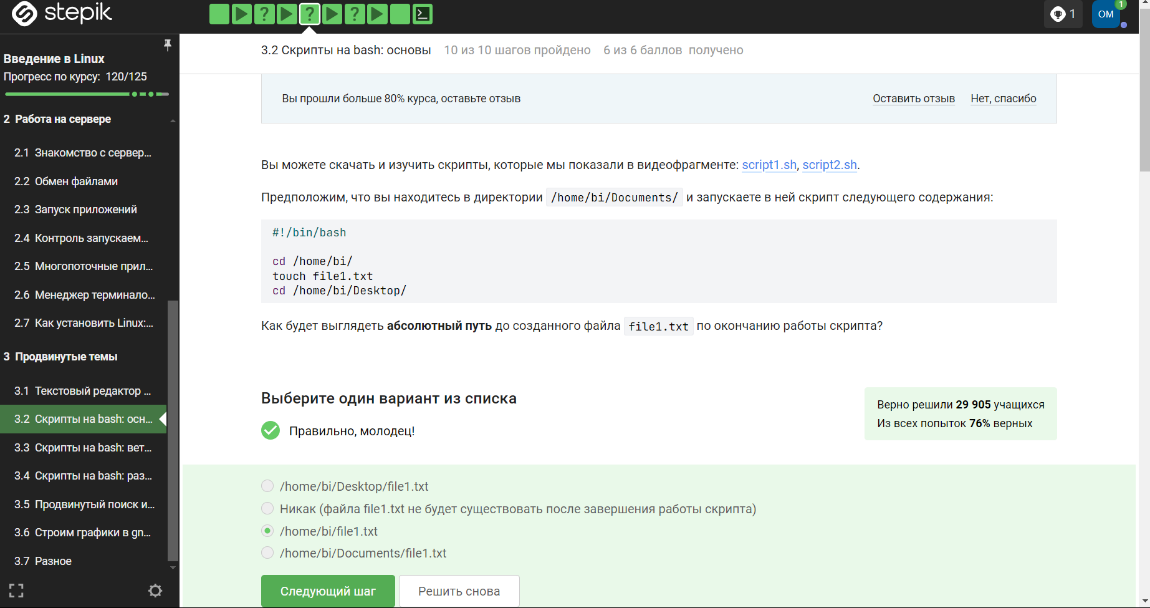


Figure 7: Задание 7

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

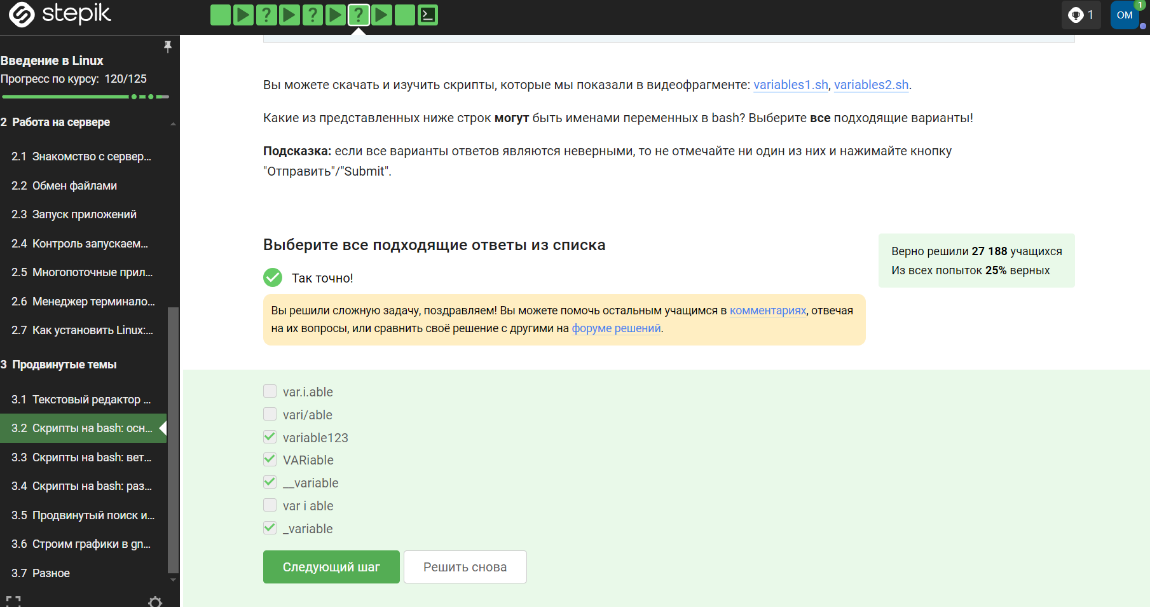


Figure 8: Задание 8

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

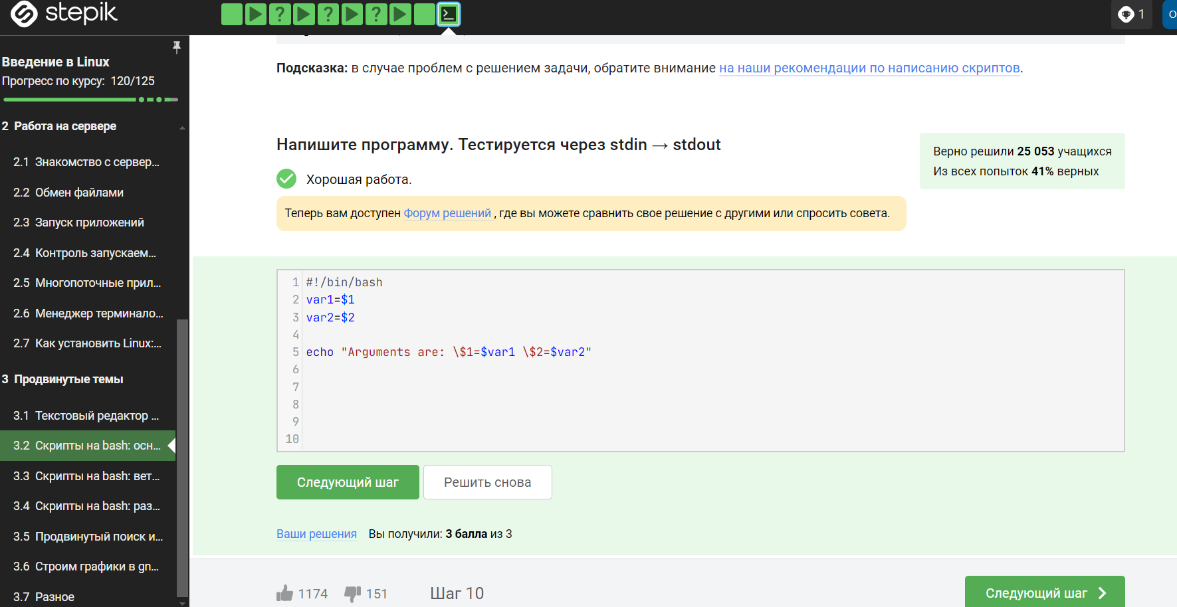


Figure 9: Задание 9

$ 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=$var1 \$2=$var2" - строка печати.

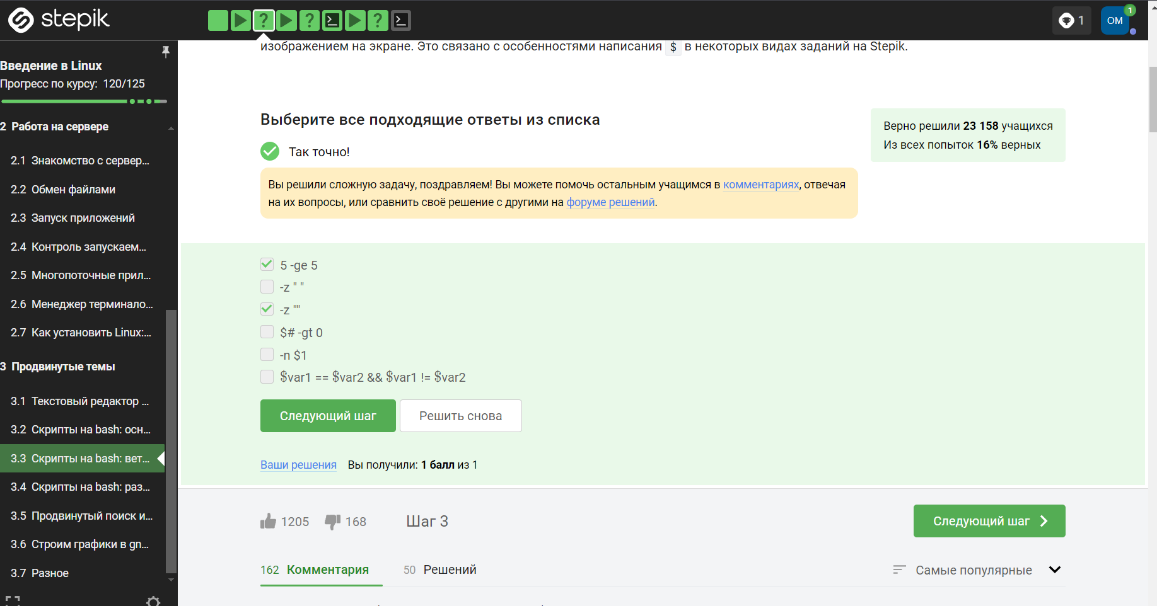


Figure 10: Задание 10

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

Имя скрипта - это 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.

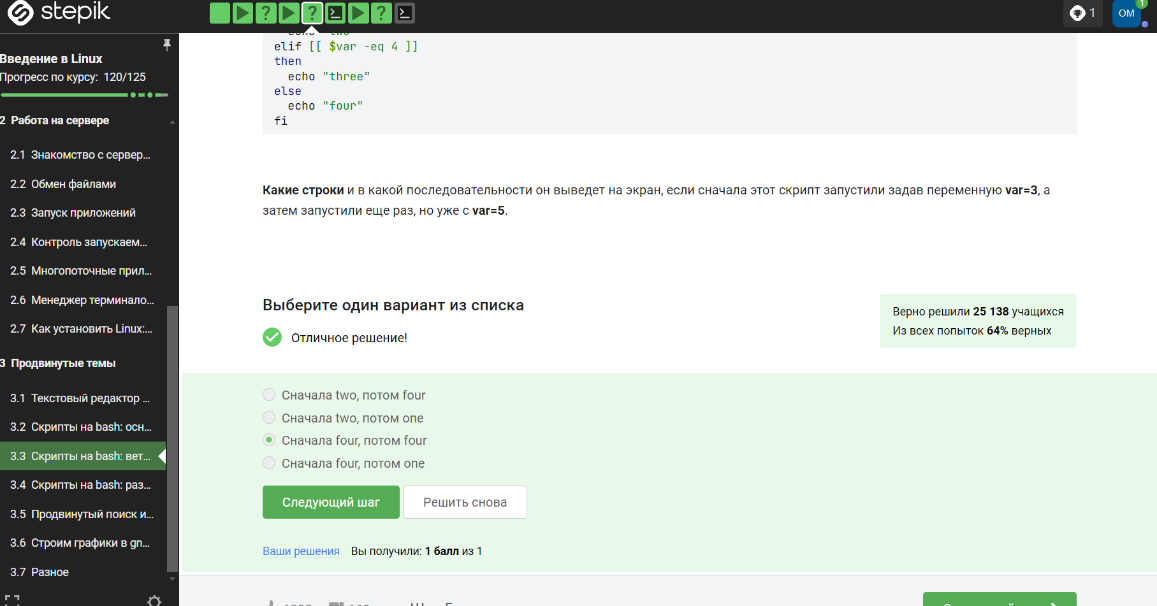


Figure 11: Задание 11

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

3 не больше 5, 3 не меньше 3, 3 не равно 4.

5 не больше 5, 5 не меньше 3, 5 не равно 4.

Оба раза выведет four.

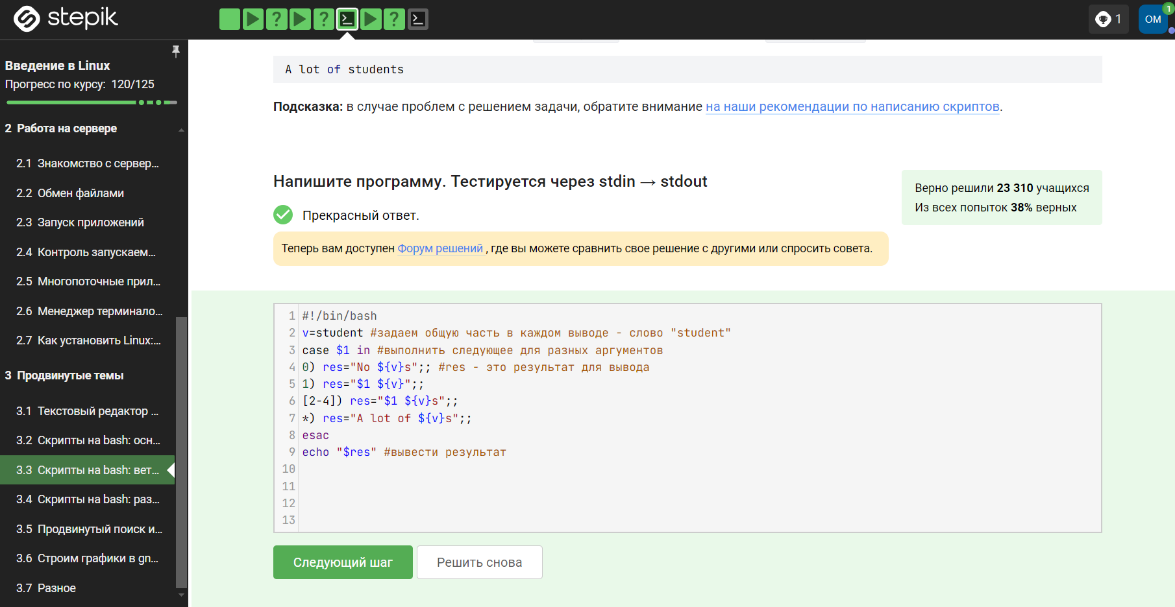


Figure 12: Задание 12

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

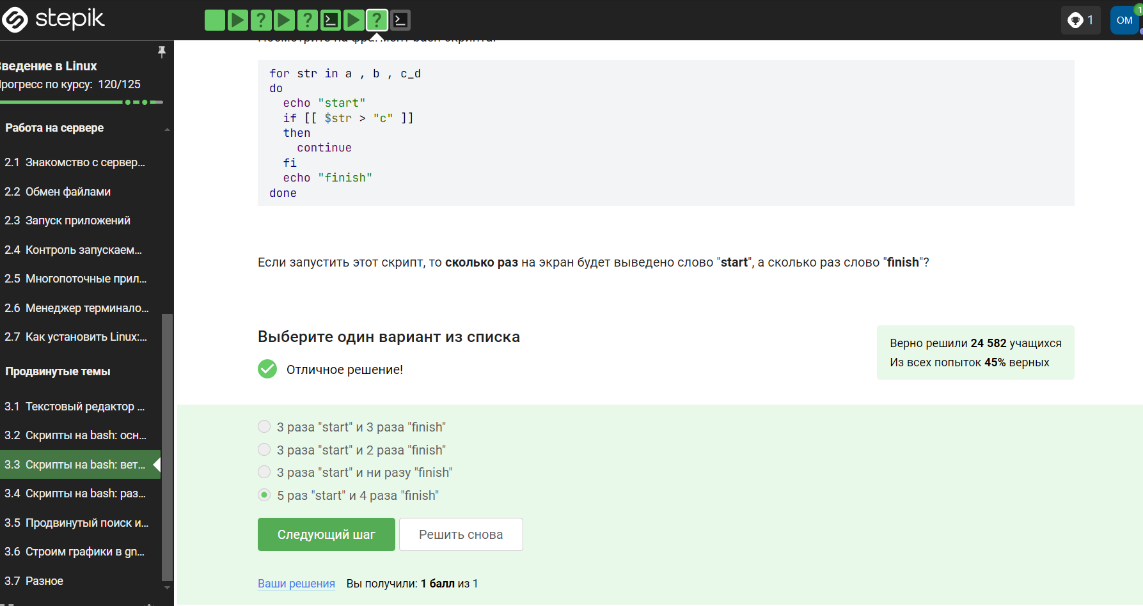


Figure 13: Задание 13

* (Start)
* a > c нет (Finish)
* (Start)
* , > c нет (Finish)
* (Start)
* b > c нет (Finish)
* (Start)
* , > c нет (Finish)
* (Start)
* c\_d > c да

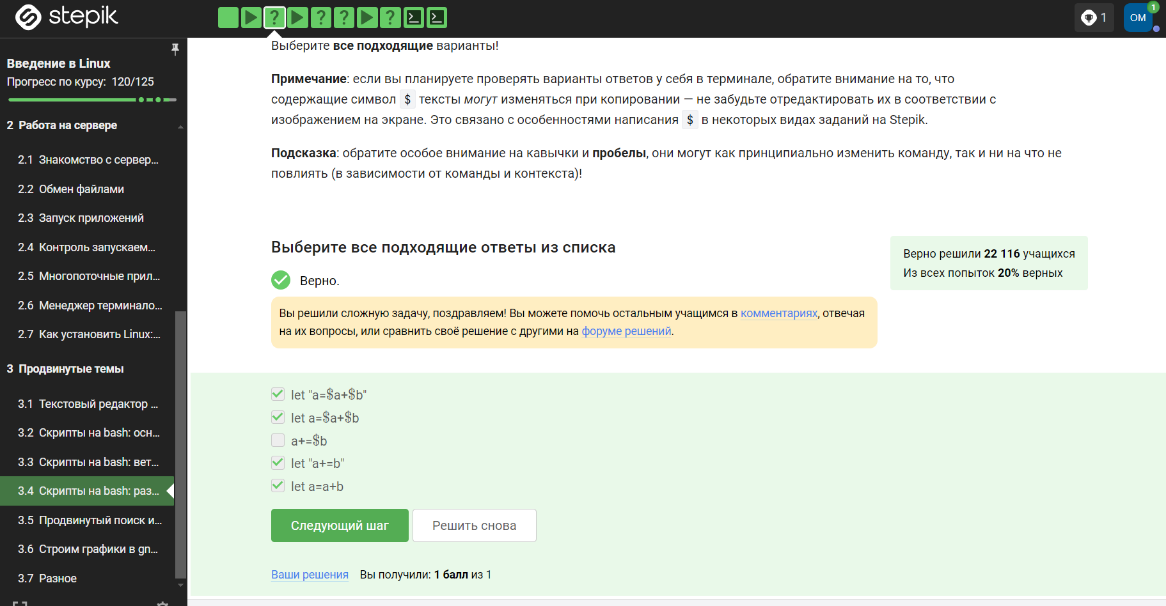


Figure 16: Задание 15

1. a = $a
2. a += b это то же самое, что и a = a + b, но с символами “+=” != “=+”
3. если выражение не в скобках, но с пробелами - работать не будет. (let a=a+b - сработает; let a = a + b - нет)



Figure 17: Задание 16

Выведет путь до директории, в которую мы перешли, так как “pwd” - это команда

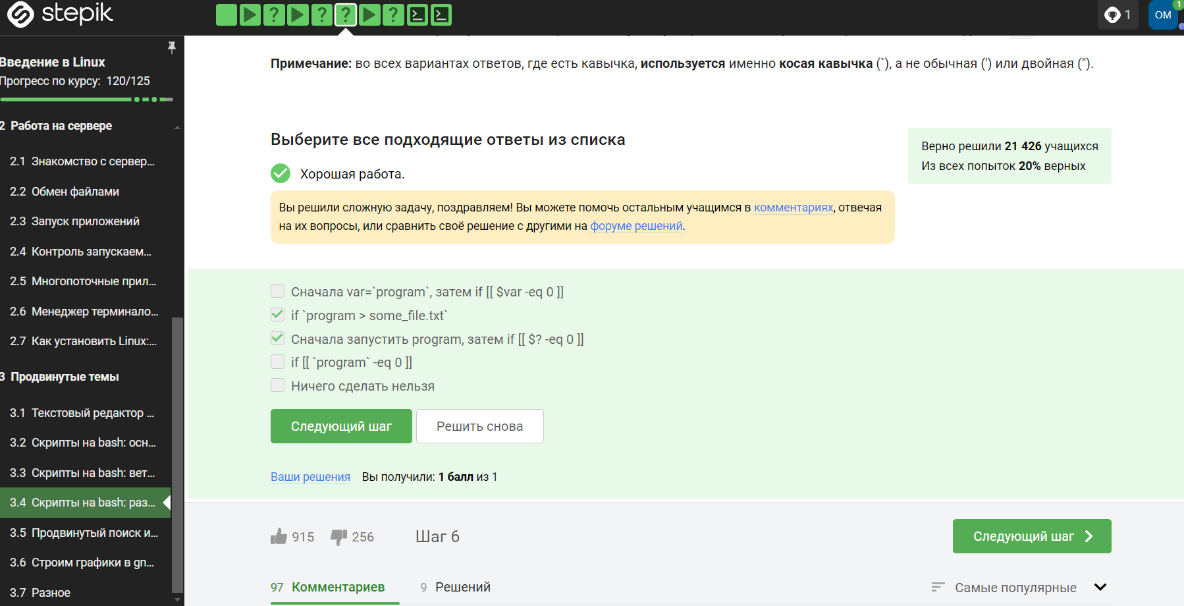


Figure 18: Задание 16\_2

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

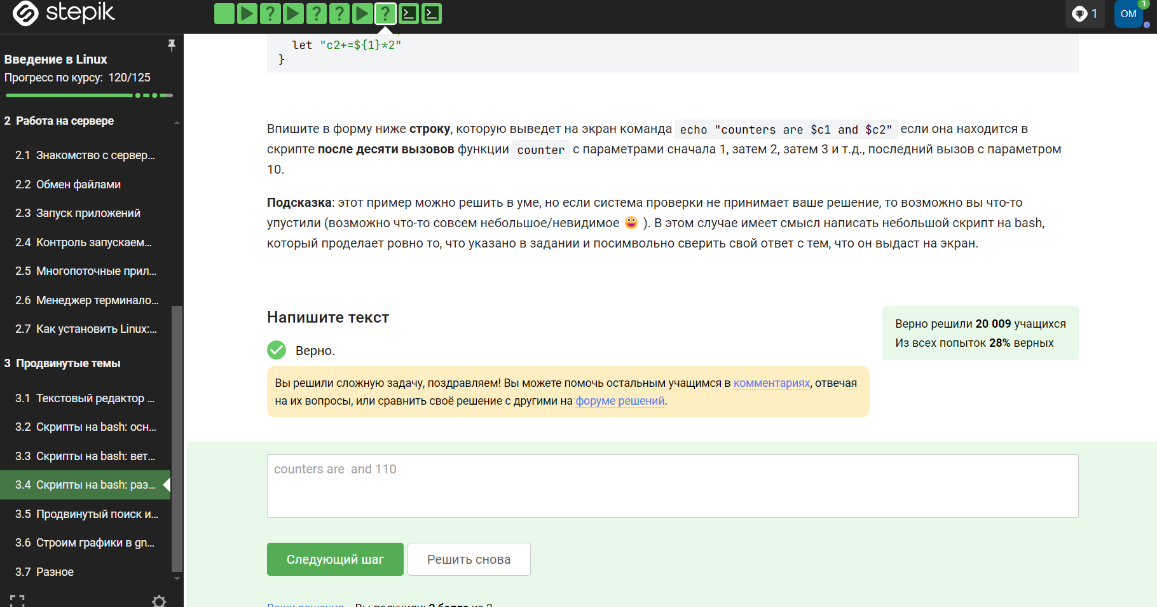


Figure 19: Задание 17

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.

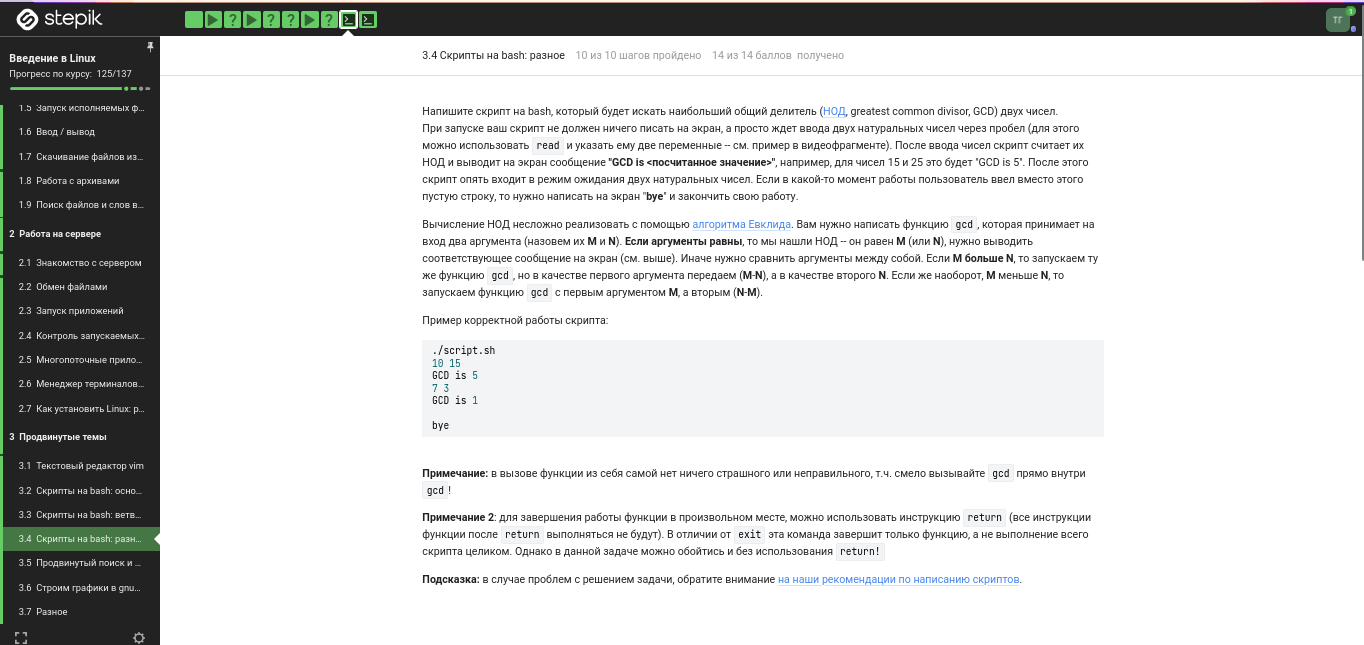


Figure 20: Задание 18

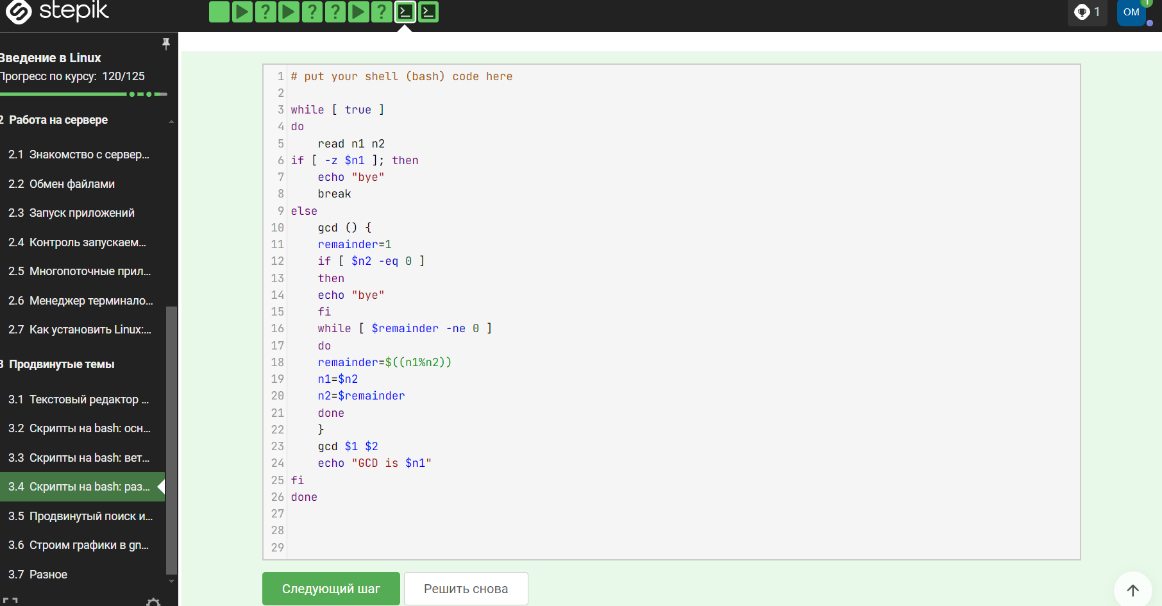


Figure 21: Задание 18

Алгоритм нахождения НОД делением

1. Большее число делим на меньшее.
2. Если делится без остатка, то меньшее число и есть НОД (следует выйти из цикла).
3. Если есть остаток, то большее число заменяем на остаток от деления.
4. Переходим к пункту 1.



Figure 22: Задание 19

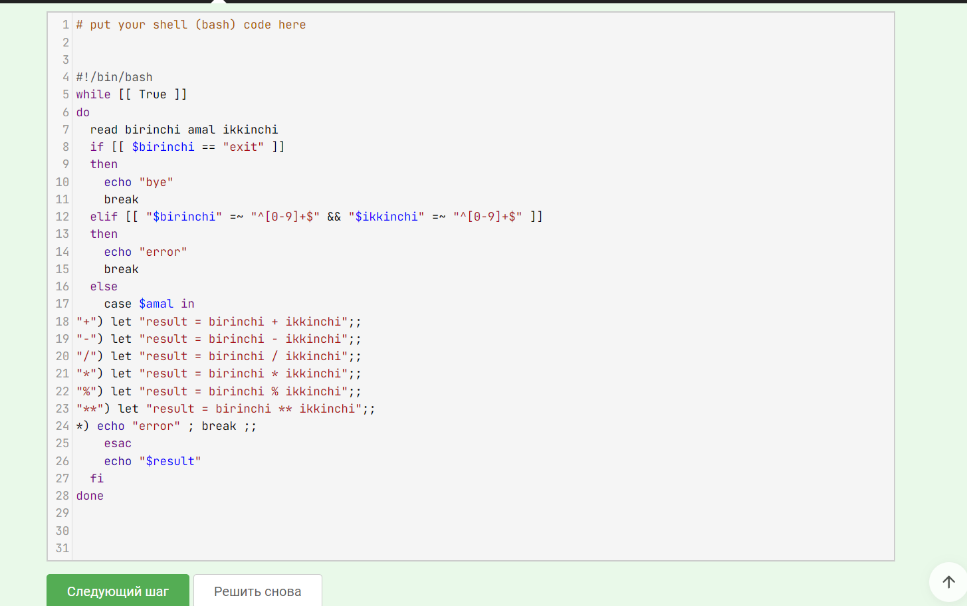


Figure 23: Задание 19

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

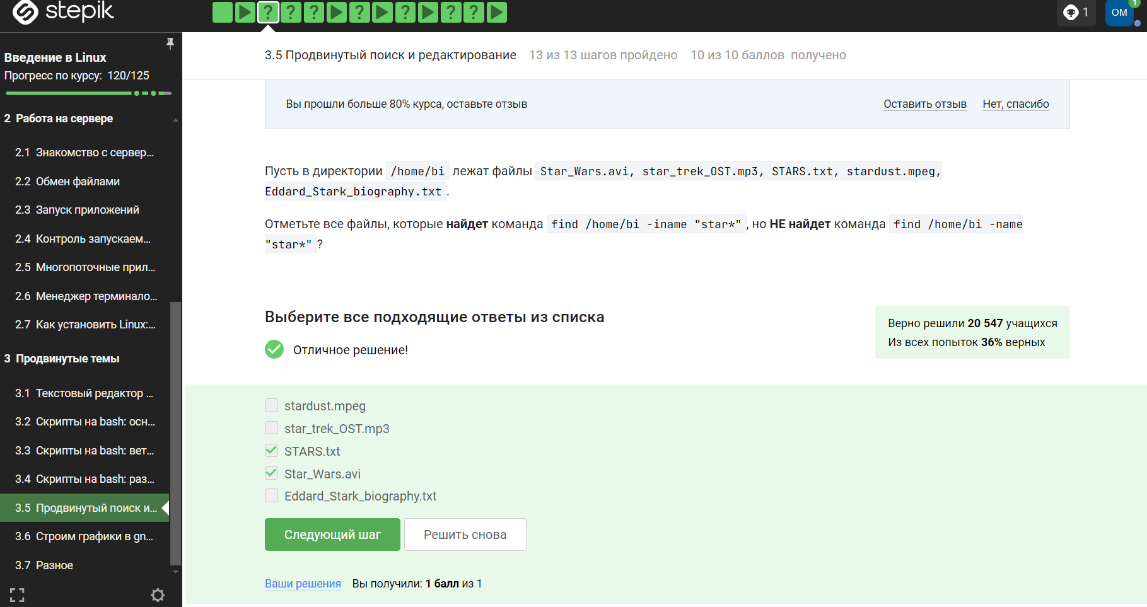


Figure 24: Задание 20

-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.

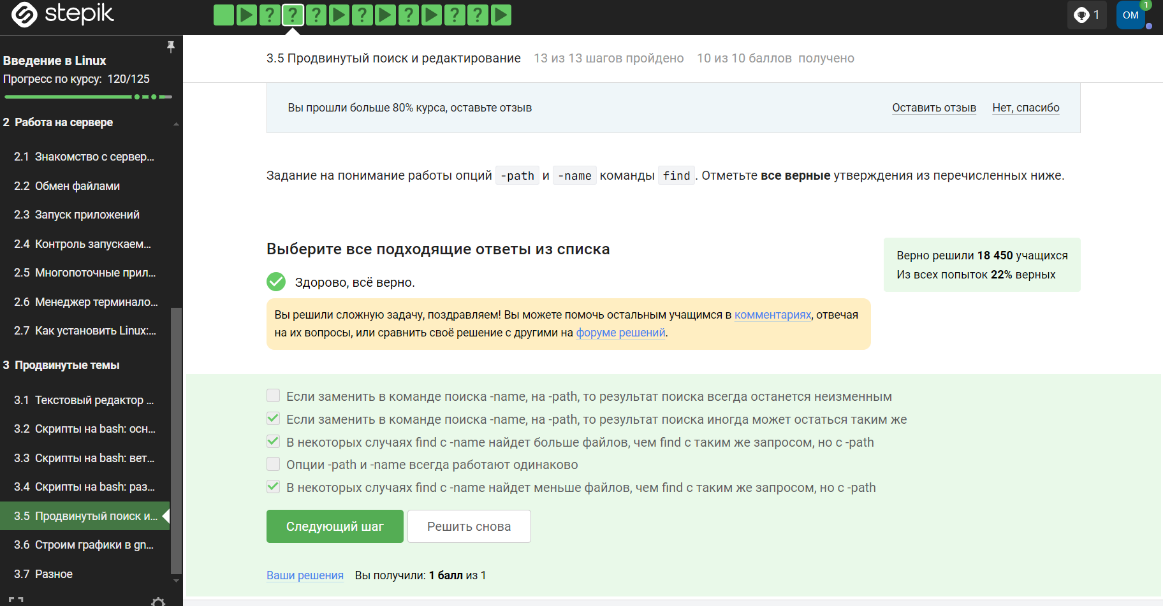


Figure 25: Задание 21

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

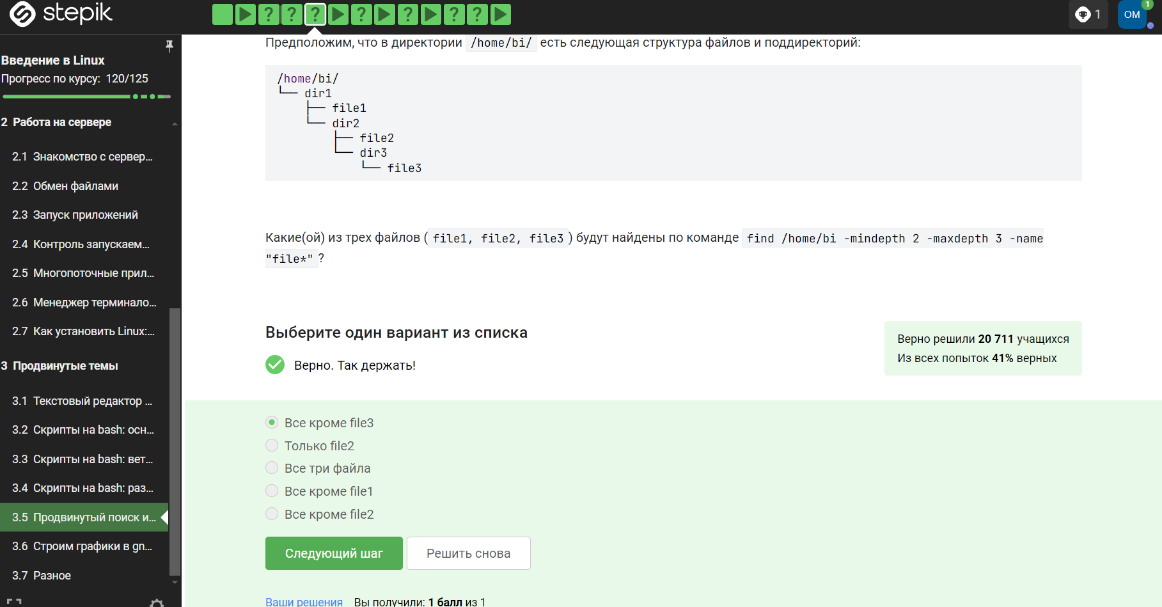


Figure 26: Задание 22

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

/home/bi -> depth=1

/home/bi/dir1 -> depth=2

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

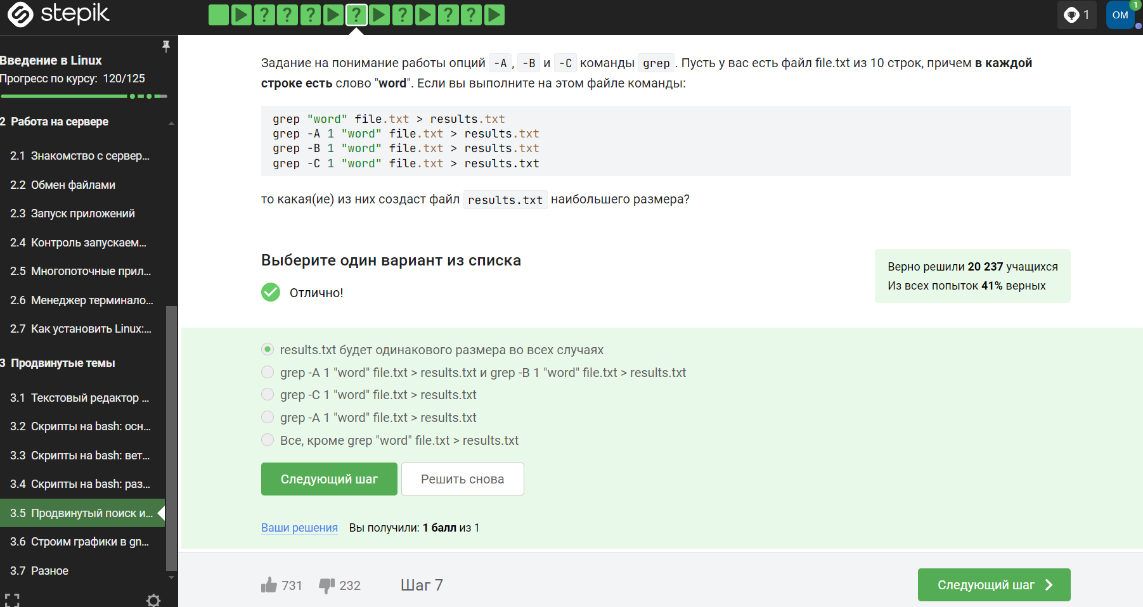


Figure 27: Задание 23

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

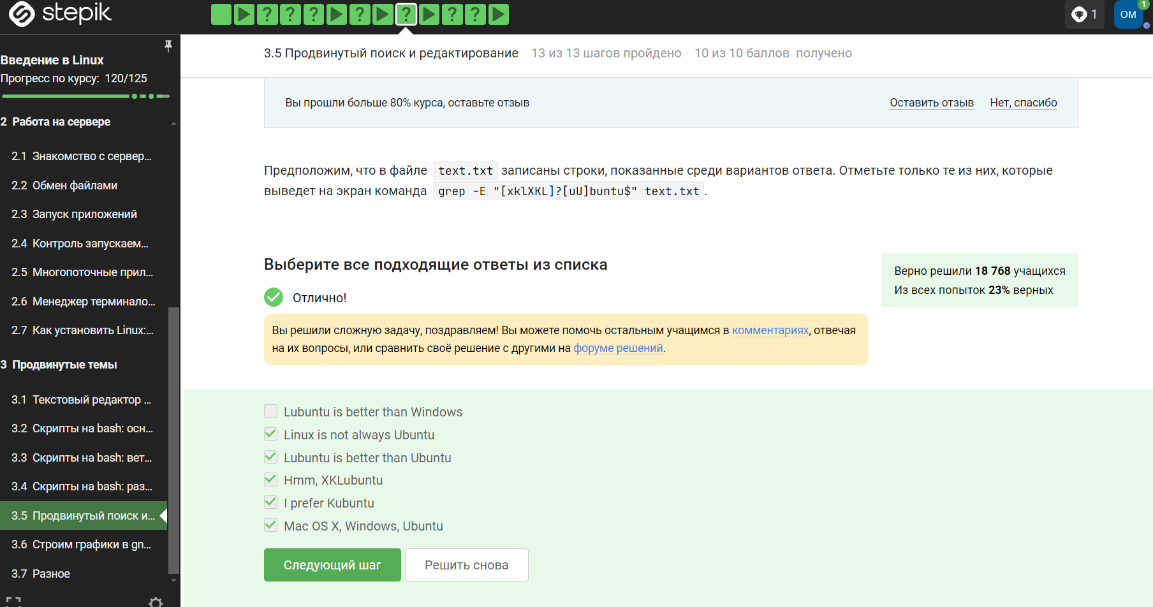


Figure 28: Задание 24

Выбираем все варианты ответа в которых в конце есть слово «ubuntu» и все.

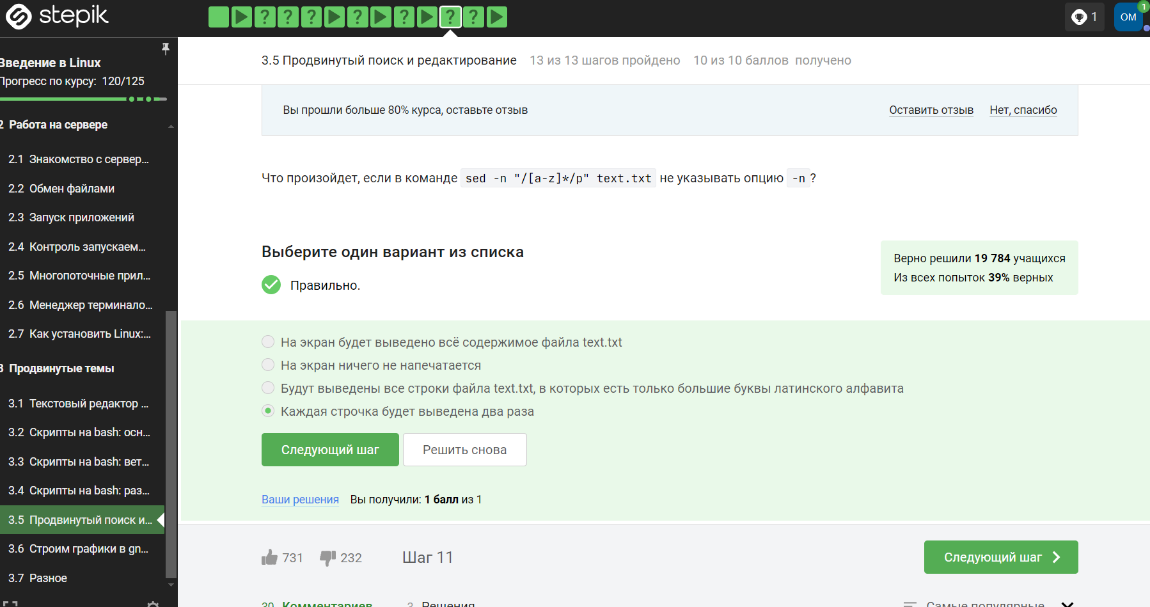


Figure 30: Задание 25

Параметр -n отключает автоматическую печать, что означает, что строки, которые вы специально не указываете для печати, не будут напечатаны, а строки, которые вы явно указываете для печати (например, с помощью p), будут напечатаны только один раз.



Figure 31: Задание 26

аббревиатура ABBA отличается от двух других аббревиатур тем, что справа он неё стоит запятая без пробела: “ABBA,”.

При этом по условию аббревиатура должна выглядеть как [ XX ] или [ XXX ] (и ещё больше X). Следовательно, для этой проверки надо добавить пробел квадратными скобками [ ] слева и, соответственно, с права.

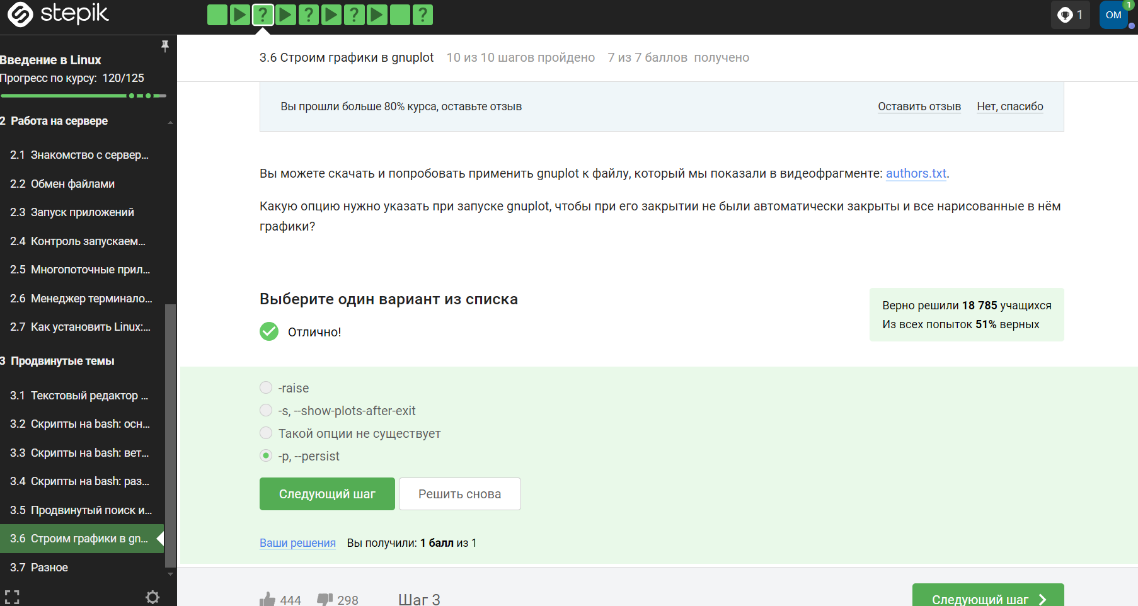


Figure 32: Задание 27

-persist позволяет сохранить окна отображения после завершения работы основной программы gnuplot.

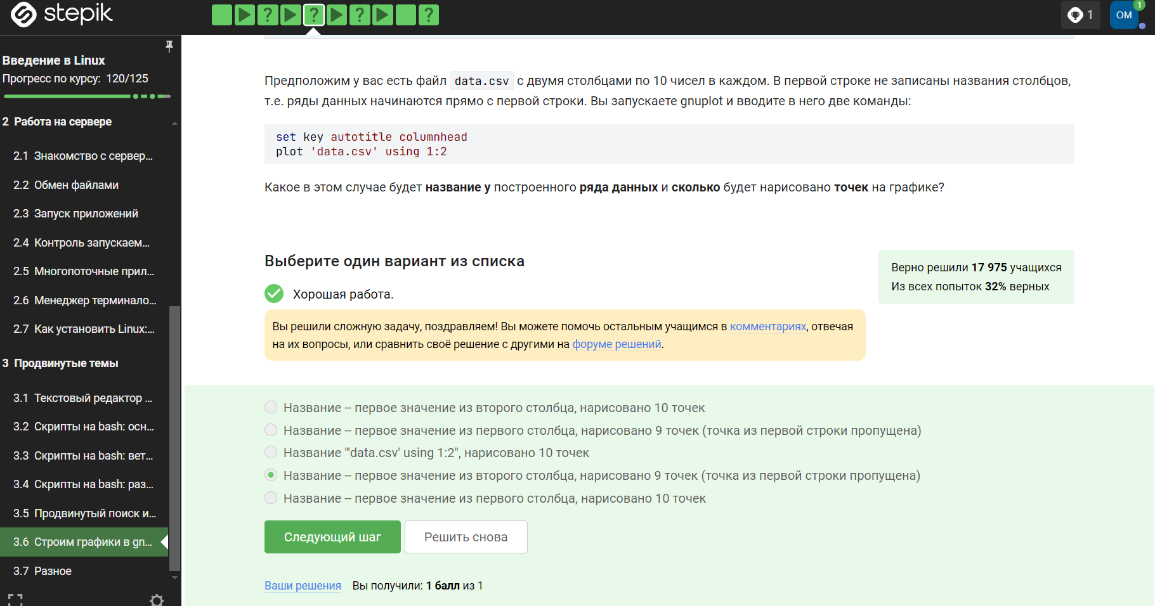


Figure 33: Задание 28

plot 'data.csv' using 1:2 даст ошибку:

warning: Skipping data file with no valid points ^ x range is invalid

Скорее всего причиной такого поведения является тот факт, что формат CSV содержит строки, где столбцы разделены запятой? Содержимое файла:

1,21  
2,22  
3,23  
4,24  
5,25  
6,26  
7,27  
8,28  
9,29  
10,30

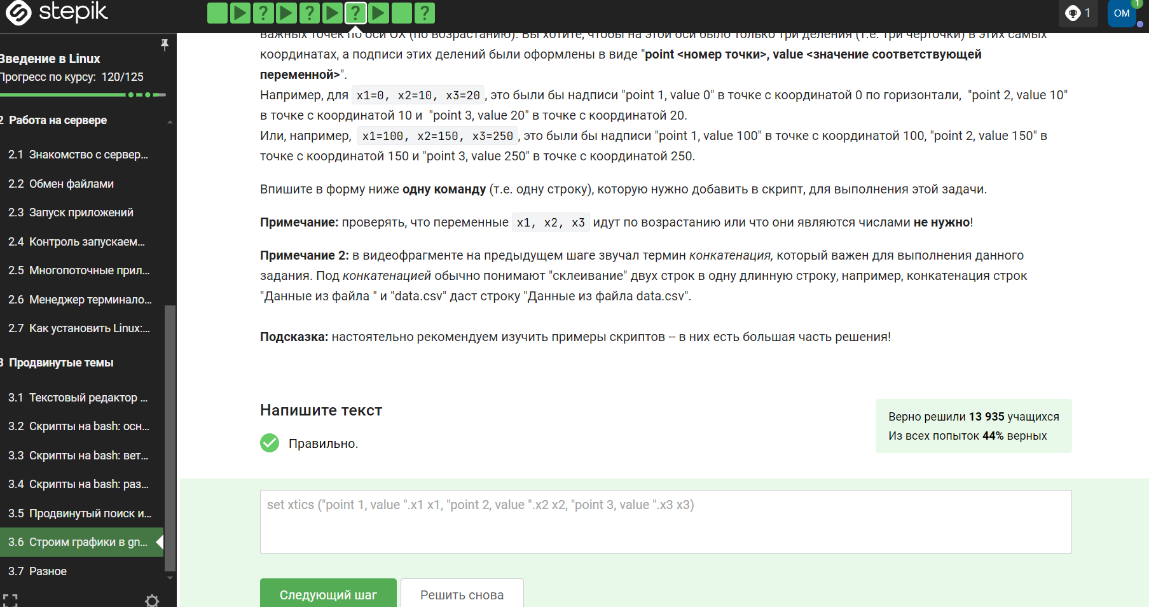


Figure 34: Задание 29

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.

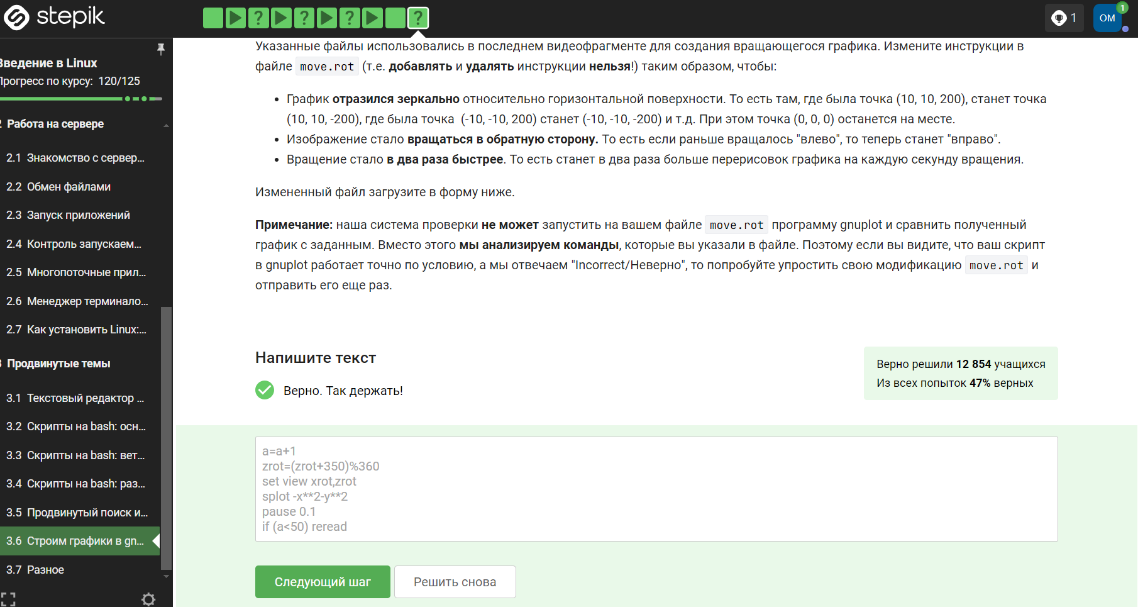


Figure 35: Задание 30

1. График строится строкой “splot x**2+y**2”.
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.

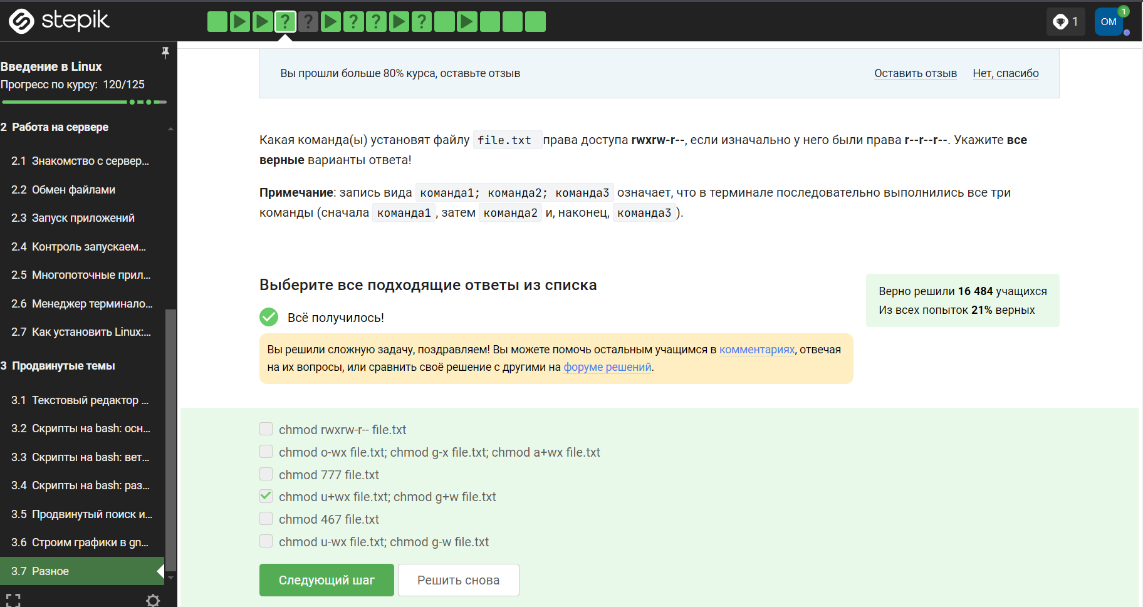


Figure 36: Задание 31

* 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.

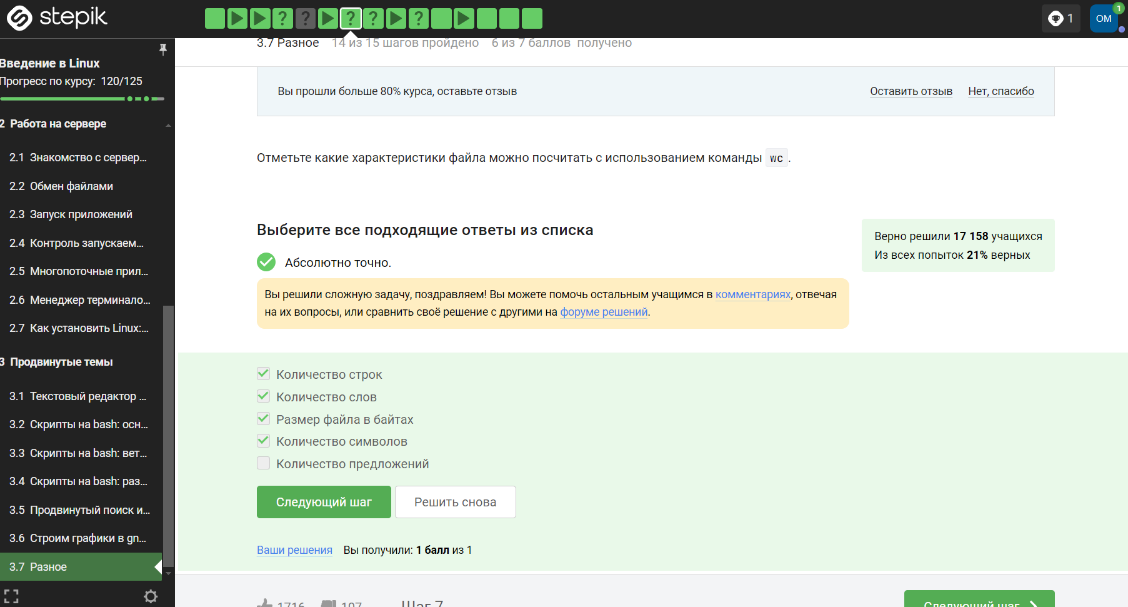


Figure 37: Задание 32

* 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

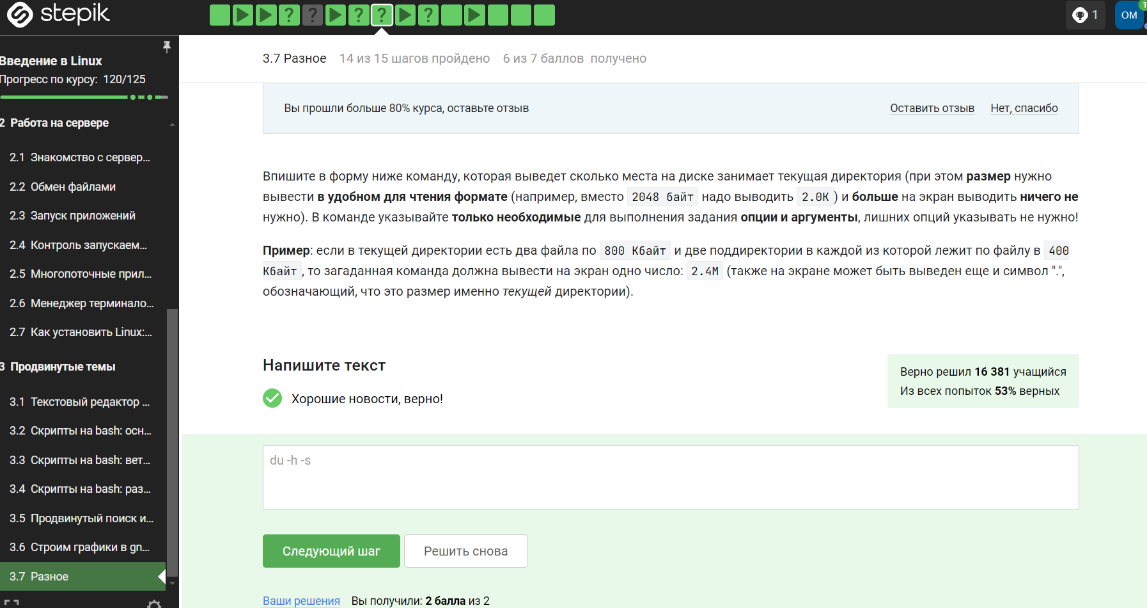


Figure 38: Задание 33

-h, – удобочитаемые размеры печати в удобочитаемом формате

-s, – суммировать и отображать только итоговое значение для каждого аргумента

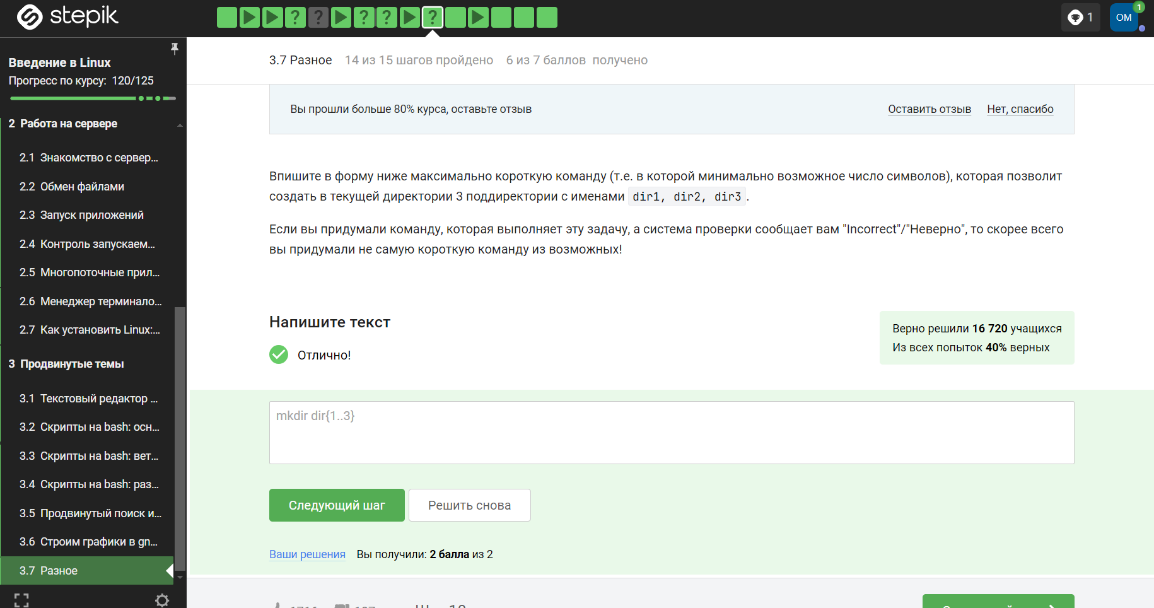


Figure 39: Задание 34

Команда создаст три директории от dir1 до dir3.

# 5 Сертификат

Figure 41: Сертификат

Figure 41: Сертификат

# 6 Выводы

Я просмотрел курс и освежил в памяти навыки работы с более сложными командами в Линукс.

# Список литературы

1. Введение в Linux