

Отчёт по внешнему курсу 3

Операционные системы

Луангсуваннавонг Сайпхачан

Содержание

1	Этап 3 внешнего курса (Продвинутые темы)	5
1.1	Текстовый редактор vim	5
1.2	Скрипты на bash: основы	8
1.3	Скрипты на bash: ветвления и циклы	10
1.4	Скрипты на bash: разное	15
1.5	Продвинутый поиск и редактирование	20
1.6	Строим графики в gnuplot	24
1.7	Разное	27

Список иллюстраций

1.1	Задание 3.1	5
1.2	Задание 3.1	6
1.3	Задание 3.1	7
1.4	Задание 3.1	7
1.5	Задание 3.1	8
1.6	Задание 3.2	8
1.7	Задание 3.2	9
1.8	Задание 3.2	9
1.9	Задание 3.2	10
1.10	Задание 3.3	11
1.11	Задание 3.3	12
1.12	Задание 3.3	13
1.13	Задание 3.3	14
1.14	Задание 3.3	15
1.15	Задание 3.4	16
1.16	Задание 3.4	16
1.17	Задание 3.4	17
1.18	Задание 3.4	18
1.19	Задание 3.4	19
1.20	Задание 3.4	20
1.21	Задание 3.5	21
1.22	Задание 3.5	21
1.23	Задание 3.5	22
1.24	Задание 3.5	22
1.25	Задание 3.5	23
1.26	Задание 3.5	23
1.27	Задание 3.5	24
1.28	Задание 3.6	25
1.29	Задание 3.6	25
1.30	Задание 3.6	26
1.31	Задание 3.6	27
1.32	Задание 3.7	28
1.33	Задание 3.7	29
1.34	Задание 3.7	29
1.35	Задание 3.7	30
1.36	Задание 3.7	30

Список таблиц

1 Этап 3 внешнего курса

(Продвинутые темы)

1.1 Текстовый редактор vim

Для выхода из vim после открытия файла нужно ввести :q, затем нажать Enter.
(рис. 1.1)

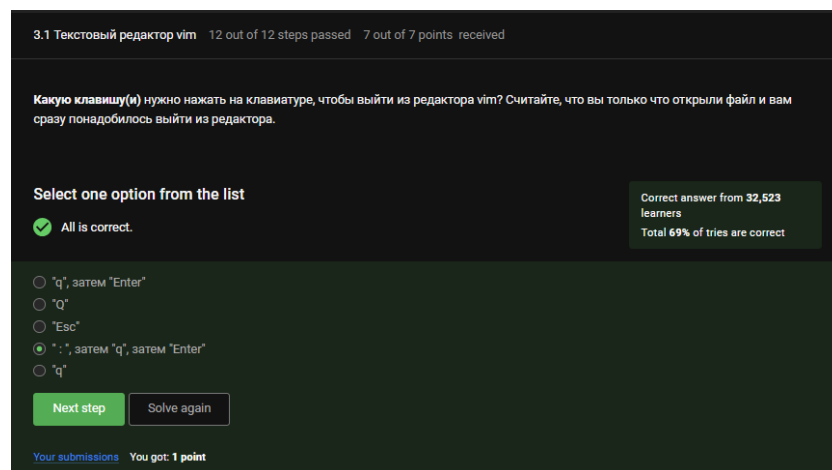


Рис. 1.1: Задание 3.1

На строке 9 слов, W перемещает по “большим словам”, потому нужно меньше нажатий, чем с w. (рис. 1.2)

3.1 Текстовый редактор vim 12 out of 12 steps passed 7 out of 7 points received

При перемещении в vim "по словам" есть небольшая разница в том, используем мы маленькую (w, e, b) или большую (W, E, B) букву. Первые перемещают нас по "словам" (word), а вторые по "большим словам" (WORD). Посмотрите справку по этим перемещениям и разберитесь в чем заключается разница между word и WORD.

А для того, чтобы убедиться, что вы разобрались, отметьте ниже **все верные** утверждения про следующую строку:
 Strange_ TEXT is_here. 2=2 YES!

Примечание: во всех утверждениях имеется ввиду, что мы находимся в редакторе vim, включен нормальный режим работы и курсор находится в самом начале строки.

Подсказка: чтобы вызвать **vim-справку** по, например, перемещению `w`, нужно открыть vim и ввести команду `:help w`. Вы попадете в то место справки, где описано это перемещение, а так как все перемещения описаны рядом, то двигаясь по тексту вверх и вниз можно прочитать и про `e` и про `b` и, самое главное, про word и WORD. Кроме того, можно вызвать сразу справку по термину word при помощи `:help word`. Чтобы закрыть справку, нужно ввести команду `:q`.

Select all correct options from the list

Great work!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 25,385 learners
Total 20% of tries are correct

- ☒ После 10 нажатий на W курсор окажется там же, где бы он был после 10 нажатий на w
- ☐ В этой строке 5 "слов" (word)
- ☐ Чтобы попасть в конец строки, нужно одинаковое число нажатий, что на W, что на w
- ☒ В этой строке 9 "слов" (word)
- ☒ Чтобы попасть в конец строки, нужно совершить меньше нажатий на W, чем на w
- ☒ Нажимая только на W, нельзя переместить курсор на "."

Next step Solve again

Your submissions You got: 1 point

Рис. 1.2: Задание 3.1

Корректный результат достигается с помощью d2wywPr и d2wwyurP, так как они заменяют нужное слово и вставляют его повторно. (рис. 1.3)

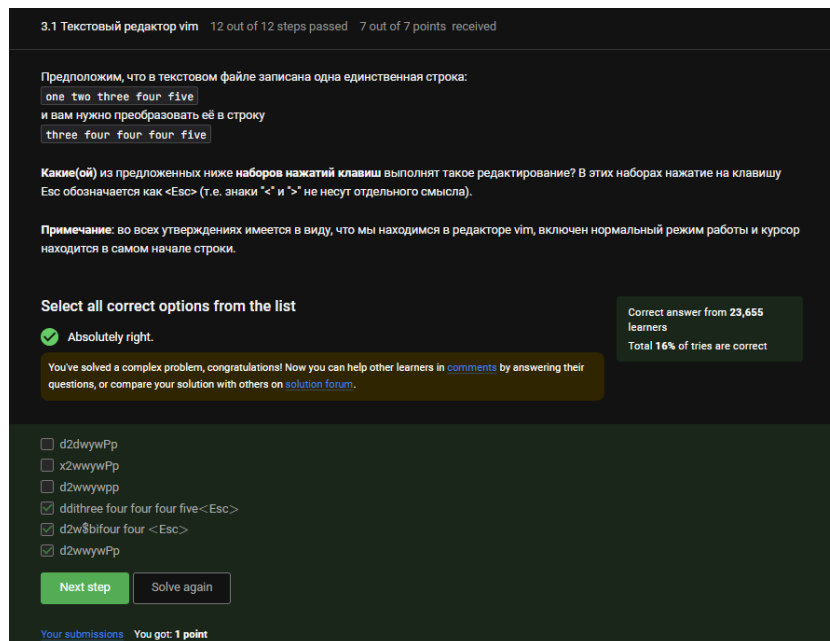


Рис. 1.3: Задание 3.1

Команда `:%s/Windows/Linux` заменяет слово Windows на Linux только один раз в каждой строке. (рис. 1.4)

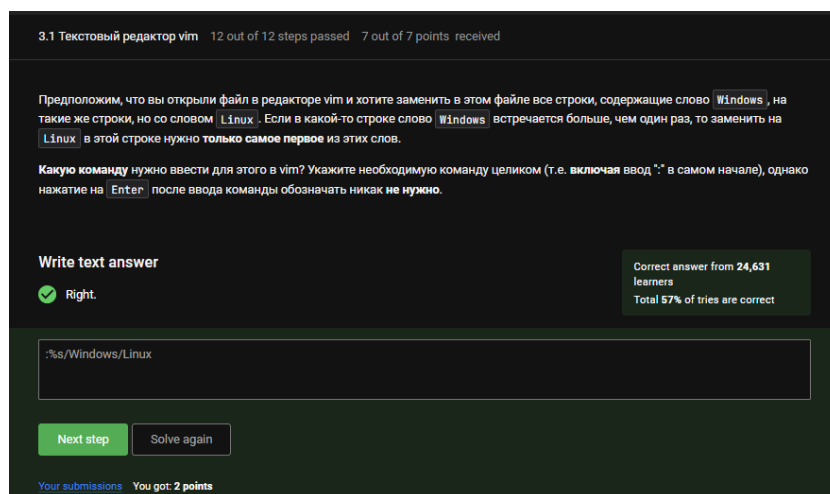


Рис. 1.4: Задание 3.1

Режим Visual активируется клавишей `v`, поддерживает перемещения и команды `d`, `y`. Завершение — `:q` или дважды `Esc`. (рис. 1.5)

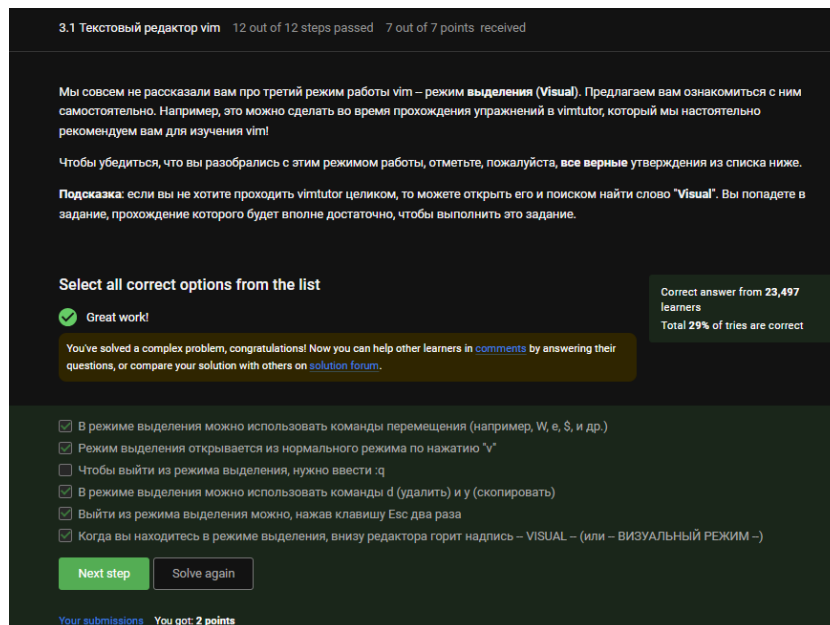


Рис. 1.5: Задание 3.1

1.2 Скрипты на bash: основы

История команд сохраняется только внутри текущей оболочки, поэтому отображаются только команды из последней оболочки C. (рис. 1.6)

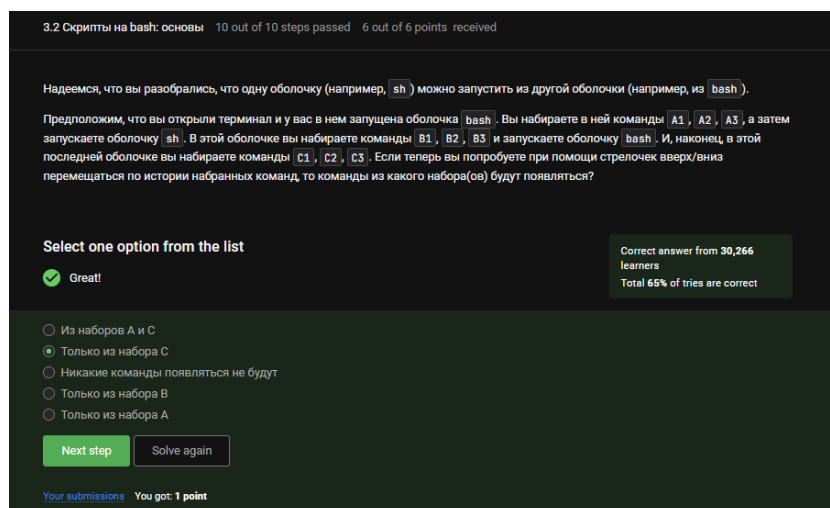


Рис. 1.6: Задание 3.2

Файл будет создан в директории /home/bi/, несмотря на начальный запуск из

/home/bi/Documents. (рис. 1.7)

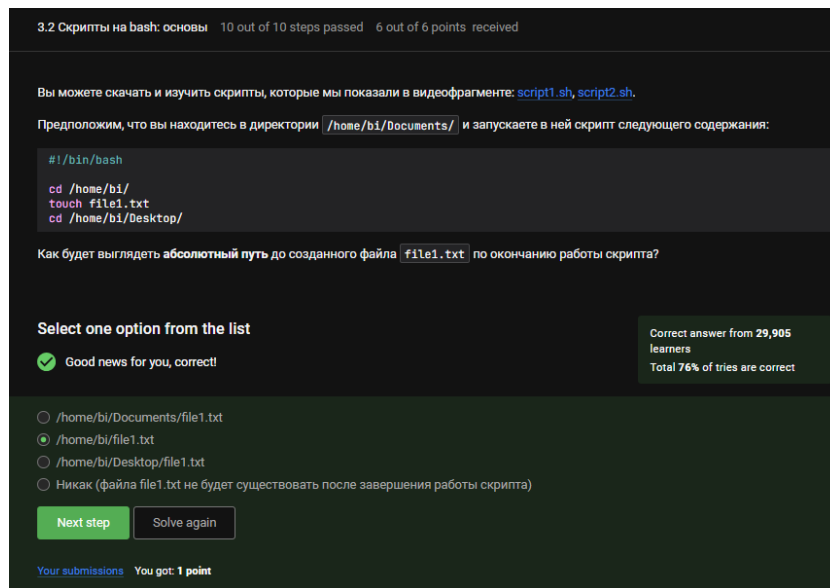


Рис. 1.7: Задание 3.2

В Bash корректные переменные: `variable123`, `__variable`, `variable`. (рис. 1.8)

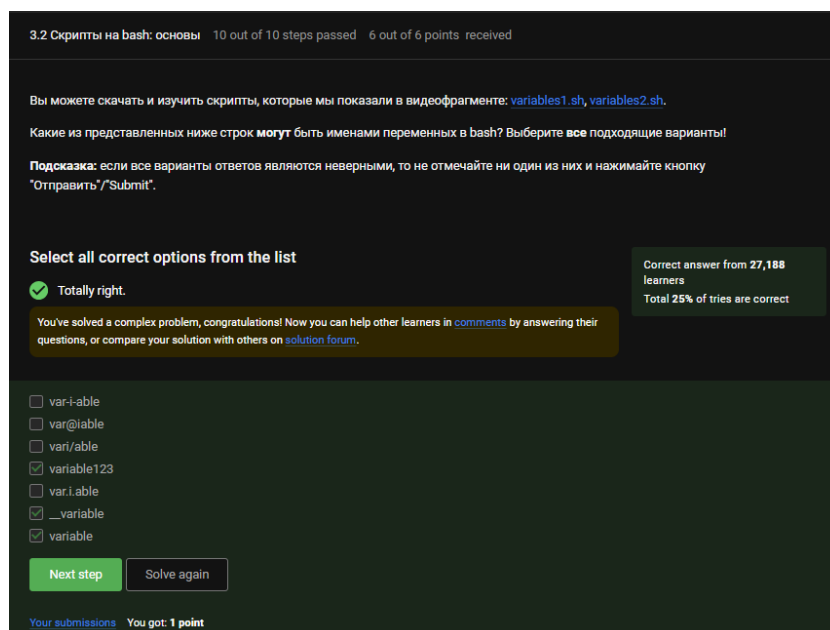


Рис. 1.8: Задание 3.2

Скрипт примет два аргумента и выведет их в требуемом формате, используя символ `"$"`. (рис. 1.9)

3.2 Скрипты на bash: основы 10 out of 10 steps passed 6 out of 6 points received

Вы можете скачать и изучить скрипт, который мы показали в видеофрагменте: [arguments.sh](#).

Напишите скрипт на bash, который принимает на вход два аргумента и выводит на экран строку следующего вида:

```
Arguments are: $1=первый_аргумент $2=второй_аргумент
```

Например, если ваш скрипт называется `./script.sh`, то при запуске его `./script.sh one two` на экране должно появиться:

```
Arguments are: $1=one $2=two
```

а при запуске `./script.sh three four` будет:

```
Arguments are: $1=three $2=four
```

Подсказка: в случае проблем с решением задачи, обратите внимание на [наши рекомендации по написанию скриптов](#).

Write a program, test using stdin → stdout

✓ Good news for you, correct!

Now you have access to the [Forum of Solutions](#) where you can discuss your solution with others.

Correct answer from 25,053 learners
Total 41% of tries are correct

```
1 var1=$1
2 var2=$2
3
4 echo "Arguments are: \${1}=${var1} \${2}=${var2}"
5
6
7
8
9
```

Next step Solve again

Your submissions You got 3 points

Рис. 1.9: Задание 3.2

1.3 Скрипты на bash: ветвления и циклы

Выбранные выражения корректно возвращают True, поскольку соответствуют синтаксису условий `[[...]]`. (рис. 1.10)

3.3 Скрипты на bash: ветвления и циклы 9 out of 9 steps passed 10 out of 10 points received

Вы можете скачать и изучить скрипт, который мы показали в видеофрагменте: [branching1.sh](#).

Предположим, вы пишете скрипт на bash и хотите использовать в нем конструкцию `if` в следующем фрагменте:

```
if [[ ... ]]
then
  echo "True"
fi
```

Вы можете написать вместо "..." (внутри `[[...]]` и не забудьте про пробелы после `[[` и перед `]]`) любое из перечисленных ниже условий. Однако мы просим вас выбрать только те из них, при которых `echo` напечатает на экран `True` вне зависимости от того, с какими параметрами был запущен ваш скрипт и какие в нем есть переменные.

Например, условие `0 -eq 0` подходит, т.к. ноль всегда равен нулю вне зависимости от аргументов и переменных внутри скрипта и на экран будет напечатано `True`. В то же время условие `$var1 -eq 0` не подходит, так как в переменной `var1` как может быть записан ноль (тогда будет напечатано `True`), так его может и не быть (тогда ничего напечатано не будет).

Примечание: если вы планируете проверять варианты ответов у себя в терминале, обратите внимание на то, что содержащие символ `$` тексты могут изменяться при копировании — не забудьте отредактировать их в соответствии с изображением на экране. Это связано с особенностями написания `$` в некоторых видах заданий на Stepik.

Select all correct options from the list

✓ You are right, well done!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 23,158 learners
Total 16% of tries are correct

- ☒ -s \$0
- ☒ \$var1 == \$var2 || \$var1 != \$var2
- ☒ -e \$0
- ☒ 5 -ge 5
- ☒ -n \$0
- ☐ \$var1 == \$var2 && \$var1 != \$var2

Next step Solve again

Рис. 1.10: Задание 3.3

При `var=3` срабатывает ветка `< 3 → four`, а при `var=5` срабатывает первая ветка `> 5` — тоже `four`. (рис. 1.11)

3.3 Скрипты на bash: ветвления и циклы 9 out of 9 steps passed 10 out of 10 points received

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [branching2.sh](#), [branching3.sh](#).

Посмотрите на фрагмент bash-скрипта:

```
if [[ $var -gt 5 ]]
then
  echo "one"
elif [[ $var -lt 3 ]]
then
  echo "two"
elif [[ $var -eq 4 ]]
then
  echo "three"
else
  echo "four"
fi
```

Какие строки и в какой последовательности он выведет на экран, если сначала этот скрипт запустили задав переменную `var=3`, а затем запустили еще раз, но уже с `var=5`.

Select one option from the list

☒ Good job.

Correct answer from 25,138 learners
Total 64% of tries are correct

☐ Сначала two, потом four
☒ Сначала four, потом four
☐ Сначала one, потом two
☐ Сначала four, потом one

Next step Solve again

Your submissions You got: 1 point

Рис. 1.11: Задание 3.3

Скрипт выводит сообщения в зависимости от количества учащихся, используя инструкции if-else (рис. 1.12)

3.3 Скрипты на bash: ветвления и циклы 9 out of 9 steps passed 10 out of 10 points received

Напишите скрипт на bash, который принимает на вход один аргумент (целое число от 0 до бесконечности), который будет обозначать число студентов в аудитории. В зависимости от значения числа нужно вывести разные сообщения.

Соответствие входа и выхода должно быть таким:

```
0 --> No students
1 --> 1 student
2 --> 2 students
3 --> 3 students
4 --> 4 students
5 и больше --> A lot of students
```

Примечание а): выводить нужно только строку справа, т.е. “-” выводить не нужно.
Примечание б): в последней строке слово “lot” с маленькой буквы!

Примечание 2: в этой и всех последующих задачах на написание скриптов, если не указано явно, что нужно проверять вход (например, что он будет именно числом и именно от 0 до бесконечности), то этого делать не нужно!

Пример №1: если ваш скрипт называется `./script.sh`, то при запуске его как `./script.sh 1` на экране должно появиться:

```
1 student
```

Пример №2: если ваш скрипт называется `./script.sh`, то при запуске его как `./script.sh 5` на экране должно появиться:

```
A lot of students
```

Подсказка: в случае проблем с решением задачи, обратите внимание на наши рекомендации по написанию скриптов.

Write a program, test using stdin → stdout

Well done!

Now you have access to the [Forum of Solutions](#) where you can discuss your solution with others.

Correct answer from 23,310 learners
Total 38% of tries are correct

```
1 if [[ $1 -eq 1 ]]; then
2     echo "1 student"
3 elif [[ $1 -gt 1 && $1 -le 4 ]]; then
4     echo "$1 students"
5 elif [[ $1 -ge 5 ]]; then
6     echo "A lot of students"
7 else
8     echo "No students"
9 fi
10
11
12
```

Next step Solve again

[Your submissions](#) You got: 3 points

Рис. 1.12: Задание 3.3

Слово finish не выводится при `str > "с"`, потому итог: 5 раз “start” и 4 раза “finish”.
(рис. 1.13)

3.3 Скрипты на bash: ветвления и циклы 9 out of 9 steps passed 10 out of 10 points received

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [loops1.sh](#), [loops2.sh](#).

Посмотрите на фрагмент bash-скрипта:

```
for str in a , b , c_d
do
  echo "start"
  if [[ $str > "c" ]]
  then
    continue
  fi
  echo "finish"
done
```

Если запустить этот скрипт, то сколько раз на экран будет выведено слово "start", а сколько раз слово "finish"?

Select one option from the list

☒ Great!

Correct answer from 24,582 learners
Total 45% of tries are correct

☐ 5 раз "start" и 5 раз "finish"

☒ 5 раз "start" и 4 раза "finish"

☐ 3 раза "start" и 3 раза "finish"

☐ 3 раза "start" и ни разу "finish"

Next step Solve again

[Your submissions](#) You got: 1 point

Рис. 1.13: Задание 3.3

Скрипт обрабатывает ввод имени и возраста, определяя группу (child, youth, adult). (рис. 1.14)

```

enter your name:
Elena Petrovna
enter your age:
25
Elena Petrovna, your group is youth
enter your name:
bye

```

Подсказка: в случае проблем с решением задачи, обратите внимание [на наши рекомендации по написанию скриптов](#).

Write a program, test using stdin → stdout

✓ All is correct.

Now you have access to the [Forum of Solutions](#) where you can discuss your solution with others.

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 21,670 learners
Total 23% of tries are correct

```

1 while true
2 do
3     echo "enter your name:"
4     read name
5     if [[ -z $name ]]
6     then echo "bye"
7     break
8 fi
9     echo "enter your age:"
10    read age
11    if [[ $age -eq 0 ]]
12    then echo "bye"
13    break
14 elif [[ $age -ge 0 ]] && [[ $age -le 16 ]]
15 then
16     echo "$name, your group is child"
17 elif [[ $age -ge 26 ]]
18 then
19     echo "$name, your group is adult"
20 elif [[ $age -ge 17 ]] && [[ $age -le 26 ]]
21 then
22     echo "$name, your group is youth"
23 fi
24 continue
25 done

```

Next step Solve again

[Your submissions](#) You got: 4 points

Рис. 1.14: Задание 3.3

1.4 Скрипты на bash: разное

Команды $a=a+b$, $a+=b$ и `let "a = a + b"` корректно увеличивают a на значение b .
(рис. 1.15)

3.4 Скрипты на bash: разное 10 out of 10 steps passed 14 out of 14 points received

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [math1.sh](#), [math2.sh](#).

Какие(ая) из предложенных ниже инструкций увеличат значение переменной `a` на значение переменной `b`? Например, если в `a` было записано 10, а в `b` было 5, то в `a` должно записаться 15. Выберите все подходящие варианты!

Примечание: если вы планируете проверять варианты ответов у себя в терминале, обратите внимание на то, что содержащие символ `$` тексты могут изменяться при копировании — не забудьте отредактировать их в соответствии с изображением на экране. Это связано с особенностями написания `$` в некоторых видах заданий на Stepik.

Подсказка: обратите особое внимание на кавычки и пробелы, они могут как принципиально изменить команду, так и ни на что не повлиять (в зависимости от команды и контекста)!

Select all correct options from the list

Well done!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 22,116 learners
Total 20% of tries are correct

☐ let "a+=b"

☐ let a = a + b

☐ a+=b

☐ a=\$a+b

☒ let "a = a + b"

Next step Solve again

[Your submissions](#) You got 1 point

Рис. 1.15: Задание 3.4

echo “pwd” выполняет команду pwd и выводит текущий путь — /home/bi. (рис. 1.16)

3.4 Скрипты на bash: разное 10 out of 10 steps passed 14 out of 14 points received

Вы можете скачать и изучить скрипт, который мы показали в видеофрагменте: [programs.sh](#).

Пусть вы находитесь в директории `/home/bi/Documents/` и запускаете в ней скрипт следующего содержания:

```
#!/bin/bash
cd /home/bi/
echo "pwd"
```

Что в этом случае выведет команда `echo` на экран?

Select one option from the list

Absolutely right.

Correct answer from 23,677 learners
Total 51% of tries are correct

☐ /home/bi/Documents

☒ /home/bi

☐ pwd

☐ Код возврата команды pwd (0 в случае успешного выполнения и не 0 в случае ошибок)

☐ "pwd"

Next step Solve again

[Your submissions](#) You got 1 point

Рис. 1.16: Задание 3.4

Команды `if /program`, `if [$? -eq 0]` и через промежуточную переменную

корректно проверяют код возврата. (рис. 1.17)

3.4 Скрипты на bash: разное 10 out of 10 steps passed 14 out of 14 points received

Мы рассказали, что можно проверить код возврата внешней программы прямо в конструкции `if` при помощи `if "program" options arguments` (действия внутри `if` выполняются, если программа закончилась с кодом 0). Однако это не всегда правда! Если запуск внешней программы выводит что-то в `stdout`, то в проверку `if` поступит именно этот вывод, а не код возврата! Вы можете убедиться в этом, написав простой bash-скрипт с использованием, например, `if `pwd``.

Однако как быть, если хочется всё-таки запустить программу `program`, которая пишет что-то в `stdout` и потом выполнить какие-то действия если её код возврата равен 0? Выберите все верные утверждения или правильно работающие конструкции `if`.

Примечание: во всех вариантах ответов, где есть кавычка, используется именно косая кавычка (`'`), а не обычная (`"`) или двойная (`"`).

Select all correct options from the list

✓ Good news for you, correct!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 21,426 learners
Total 20% of tries are correct

- ☐ `if [["program" -eq 0]]`
- ☐ Ничего сделать нельзя
- ☐ Сначала `var="program"`, затем `if [[$var -eq 0]]`
- ☐ Сначала запустить `program`, затем `if [[$? -eq 0]]`
- ☒ `if "program" > some_file.txt`

Next step Solve again

[Your submissions](#) You got: 1 point

Рис. 1.17: Задание 3.4

После 10 вызовов функции `counter`, сумма параметров 1–10 даёт 55, и 2+1, 2+2, ... 2+10 даёт 110. (рис. 1.18)

3.4 Скрипты на bash: разное 10 out of 10 steps passed 14 out of 14 points received

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [functions1.sh](#), [functions2.sh](#).

Посмотрите на функцию из bash-скрипта:

```
counter () # takes one argument
{
    local let "c1+=$1"
    let "c2+=$((c1)+2"
}
```

Впишите в форму ниже строку, которую выведет на экран команда `echo "counters are $c1 and $c2"` если она находится в скрипте после десяти вызовов функции `counter` с параметрами сначала 1, затем 2, затем 3 и т.д., последний вызов с параметром 10.

Подсказка: этот пример можно решить в уме, но если система проверки не принимает ваше решение, то возможно вы что-то упустили (возможно что-то совсем небольшое/невидимое 🤔). В этом случае имеет смысл написать небольшой скрипт на bash, который проделает ровно то, что указано в задании и посимвольно сверить свой ответ с тем, что он выдаст на экран.

Write text answer

✓ Right.

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 20,009 learners
Total 28% of tries are correct

counters are and 110

[Next step](#) [Solve again](#)

[Your submissions](#) You got 2 points

Рис. 1.18: Задание 3.4

Для этой задачи я пишу функцию, которая находит наибольший общий делитель (GCD), а также проверяет наличие пустых входных данных, после чего выдает ответ (рис. 1.19)

```
GCD is 1
bye
```

Примечание: в вызове функции из себя самой нет ничего страшного или неправильного, т.ч. смело вызывайте `gcd` прямо внутри `gcd`!

Примечание 2: для завершения работы функции в произвольном месте, можно использовать инструкцию `return` (все инструкции функции после `return` выполняться не будут). В отличие от `exit`, эта команда завершит только функцию, а не выполнение всего скрипта целиком. Однако в данной задаче можно обойтись и без использования `return`!

Подсказка: в случае проблем с решением задачи, обратите внимание на [наши рекомендации по написанию скриптов](#).

Write a program, test using stdin → stdout

✓ Correct.

Now you have access to the [Forum of Solutions](#) where you can discuss your solution with others.

Correct answer from 18,148 learners
Total 35% of tries are correct

```
1 # put your shell (bash) code here
2 gcd ()
3 {
4     if [[ -z $M && -z $N ]]; then
5         echo 'bye'
6         exit
7     fi
8
9     if [[ $M -eq $N ]]; then
10        echo "GCD is $M"
11    else
12        if [[ $M -gt $N ]]; then
13            let "M-= $N"
14            gcd $M $N
15        else
16            let "N-= $M"
17            gcd $M $N
18        fi
19    fi
20 }
21
22 while [[ true ]]; do
23     read M N
24     gcd $M $N
25 done
26
27
```

Next step Solve again

Рис. 1.19: Задание 3.4

Скрипт проверяет валидность ввода чисел и операций, обрабатывает `exit` и арифметику. (рис. 1.20)

то на экране будет:

```
#2
error
```

т.к. вторая команда была некорректной (в ней всего один аргумент, т.к. нет пробелов между числами и операцией, а единственная допустимая команда из одного аргумента это "exit").

Подсказка: в случае проблем с решением задачи, обратите внимание на [наши рекомендации по написанию скриптов](#).

Write a program, test using stdin → stdout

✓ Great!

Now you have access to the [Forum of Solutions](#) where you can discuss your solution with others.

Correct answer from 16,980 learners
Total 36% of tries are correct

```
1 # put your shell (bash) code here
2
3
4 while [[ True ]]
5 do
6   read a op b
7   if [[ $a == "exit" ]]
8   then
9     echo "bye"
10    break
11  elif [[ "$a" =~ ^[0-9]+$ && "b" =~ ^[0-9]+$ ]]
12  then
13    echo "error"
14    break
15  else
16    case $op in
17      "+") let "result = a + b";;
18      "-") let "result = a - b";;
19      "/" let "result = a / b";;
20      "*") let "result = a * b";;
21      "%") let "result = a % b";;
22      "**") let "result = a ** b";;
23      *) echo "error" ; break ;;
24    esac
25    echo "$result"
26  fi
27 done
28
29
```

Next step Solve again

Рис. 1.20: Задание 3.4

1.5 Продвинутый поиск и редактирование

Команда `find -iname "star*"` найдёт файлы с учетом регистра, включая `Star_Wars.avi`. (рис. 1.21)

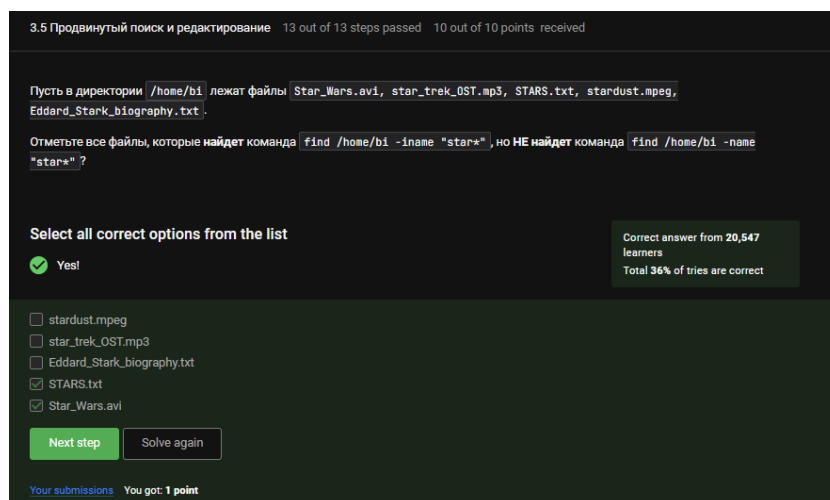


Рис. 1.21: Задание 3.5

-path может дать аналогичный результат как -name, но не всегда, особенно с учётом регистра. (рис. 1.22)

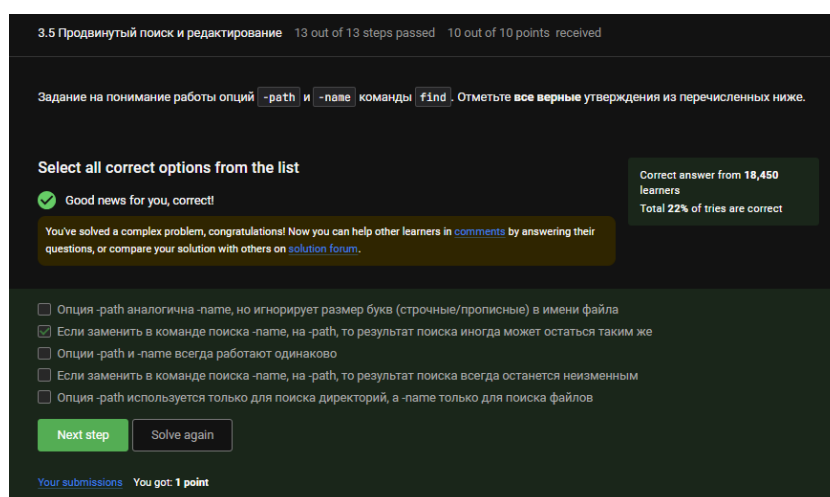


Рис. 1.22: Задание 3.5

Команда `-mindepth 2 -maxdepth 3` исключает файлы на первом уровне, потому `file3` не найден. (рис. 1.23)

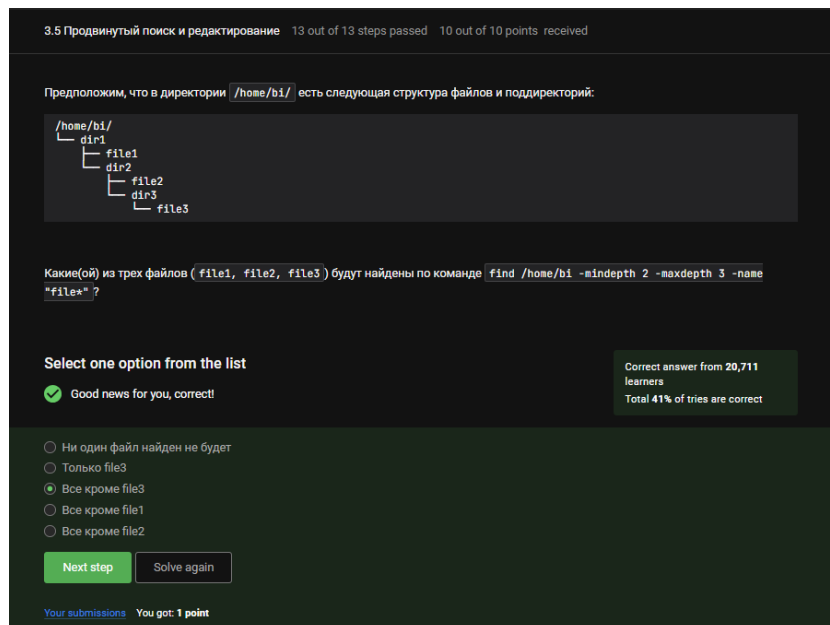


Рис. 1.23: Задание 3.5

Так как word есть в каждой строке, любой вариант -A, -B, -C даст одинаковый размер файла. (рис. 1.24)

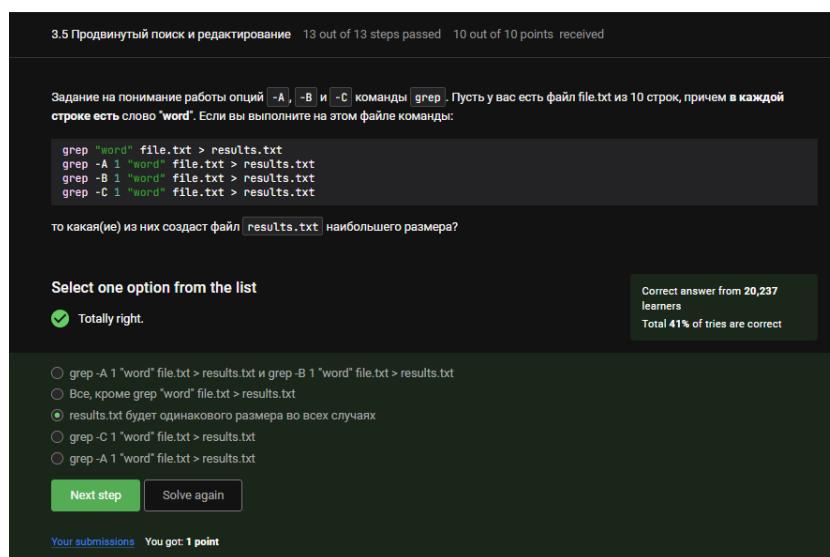


Рис. 1.24: Задание 3.5

Команда находит строки, заканчивающиеся на `ubuntu`, с учетом возможного префикса `X`, `x`, `K` и т.д. (рис. 1.25)

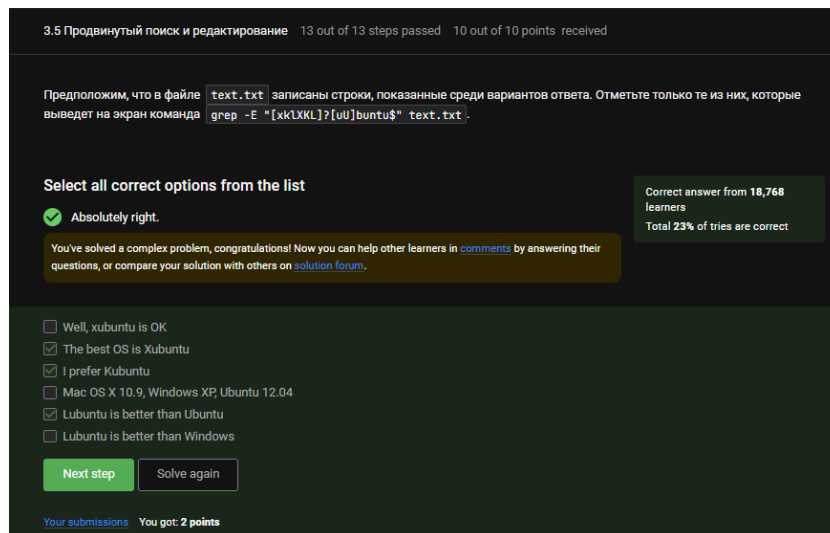


Рис. 1.25: Задание 3.5

Без `-n` `sed` выведет строку и результат команды, поэтому каждая строка выводится дважды. (рис. 1.26)

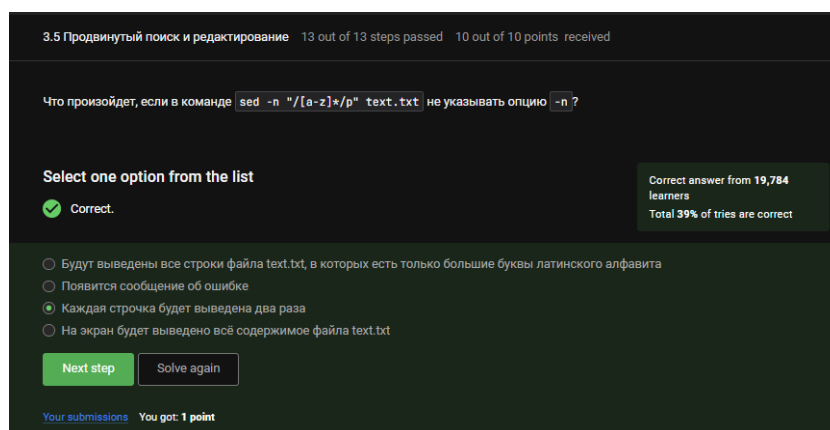


Рис. 1.26: Задание 3.5

Регулярное выражение заменяет аббревиатуры на “abbreviation”, сохраняя пробелы. (рис. 1.27)

результат в файл `edited.txt` (на экран при этом ничего выводить не нужно). Обратите внимание, что в инструкции должны быть указаны и сам `sed`, и оба файла!

Под "аббревиатурой" будем понимать слово, которое удовлетворяет следующим условиям:

- состоит только из больших букв латинского алфавита,
- состоит из хотя бы двух букв,
- окружено одним пробелом с каждой стороны.

При этом будем считать, что в тексте не может быть две "аббревиатуры" подряд. Например, текст `" YOU YOU and YOU !"` является некорректным (в нем есть две "аббревиатуры", но они идут подряд) и на таких примерах мы проверять вашу инструкцию не будем.

Пример: если у вас был текст `"Hi, I heard these songs by ABBA, TLA and DM !"`, то он должен быть преобразован в `"Hi, I heard these songs by ABBA, abbreviation and abbreviation !"`.

Примечание: после вашей замены "аббревиатуры" на слово "abbreviation" количество пробелов в тексте не должно меняться!

Внимание! Во время проверки мы не запускаем команду, которую вы ввели на реальном файле с "аббревиатурами" (это небезопасно, можно же ввести `rm -rf /*`!). Вместо этого мы сперва анализируем структуру вашей инструкции (например, что в ней использован именно `sed` и сделано это ровно один раз, что на вход подается `input.txt`, а результат будет записан в `edited.txt` и т.д.), а затем запускаем её смысловую часть (т.е. поиск по регулярному выражению и замена на "abbreviation") на тестовых примерах. К сожалению, наш запуск не идеально повторяет `sed`, но он очень близок к нему. Главная "несовместимость" заключается в том, что наша проверка не понимает идущие подряд символы, отвечающие за количество повторений (т.е. `*`, `+`, `?` и `{}`). Однако эту "несовместимость" легко исправить указав при помощи "`(`" и `)`" какой из символов к чему относится! Например, регулярное выражения `a+?` (ноль или один раз по одной или более букве "a") нужно записать как `(a)+?` (при этом запись `(a)+?`, конечно же, не поможет).

Write text answer

✔ Great!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

Correct answer from 16,632 learners
Total 34% of tries are correct

```
sed 's/[A-Z]\(2,\)/abbreviation/g' input.txt > edited.txt
```

Next step Solve again

Your submissions You got 3 points

Рис. 1.27: Задание 3.5

1.6 Строим графики в gnuplot

Опция `-p` (`-persist`) предотвращает автоматическое закрытие окна с графиком после построения. (рис. 1.28)

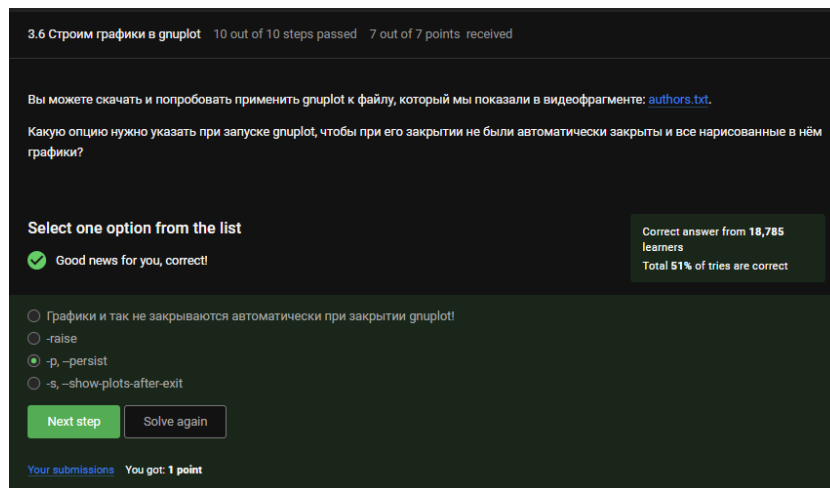


Рис. 1.28: Задание 3.6

Пропущена первая строка с заголовками, поэтому построено 9 точек и в заголовке используется второе значение. (рис. 1.29)

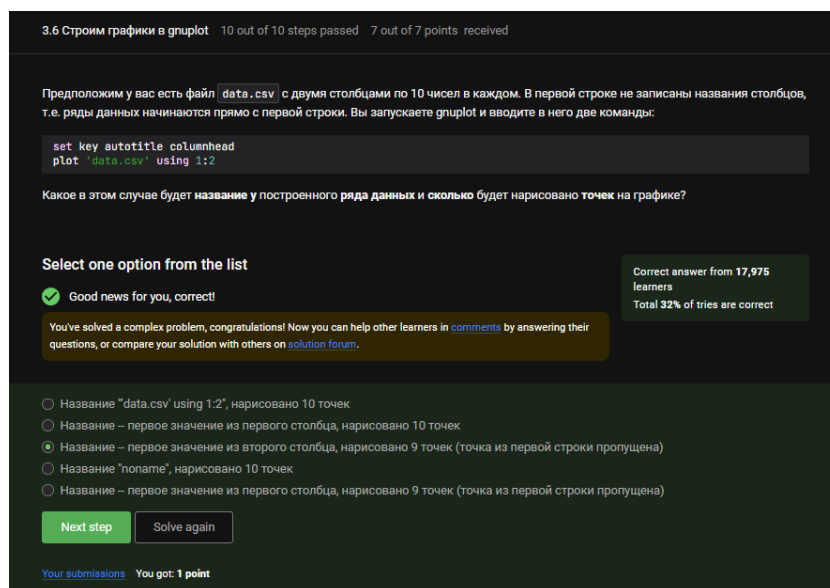


Рис. 1.29: Задание 3.6

Команда `set xtics` формирует подписи точек по значениям переменных `x1`, `x2`, `x3`. (рис. 1.30)

3.6 Строим графики в `gnuplot` 10 out of 10 steps passed 7 out of 7 points received

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [plot.gnu](#), [plot_advanced.gnu](#), [plot_advanced2.gnu](#). Все три скрипта основаны на [этой заметке](#), данные также взяты оттуда.

Предположим, что вы пишете `gnuplot`-скрипт и у вас в нем есть три переменные `x1`, `x2`, `x3`, в которых записаны координаты важных точек по оси OX (по возрастанию). Вы хотите, чтобы на этой оси было только три деления (т.е. три черточки) в этих самых координатах, а подписи этих делений были оформлены в виде "point «номер точки», value «значение соответствующей переменной»".

Например, для `x1=0`, `x2=10`, `x3=20`, это были бы надписи "point 1, value 0" в точке с координатой 0 по горизонтали, "point 2, value 10" в точке с координатой 10 и "point 3, value 20" в точке с координатой 20.

Или, например, `x1=100`, `x2=150`, `x3=250`, это были бы надписи "point 1, value 100" в точке с координатой 100, "point 2, value 150" в точке с координатой 150 и "point 3, value 250" в точке с координатой 250.

Впишите в форму ниже **одну команду** (т.е. одну строку), которую нужно добавить в скрипт, для выполнения этой задачи.

Примечание: проверять, что переменные `x1`, `x2`, `x3` идут по возрастанию или что они являются числами **не нужно**!

Примечание 2: в видеофрагменте на предыдущем шаге звучал термин *конкатенация*, который важен для выполнения данного задания. Под *конкатенацией* обычно понимают "склеивание" двух строк в одну длинную строку, например, конкатенация строк "Данные из файла " и "data.csv" даст строку "Данные из файла data.csv".

Подсказка: настоятельно рекомендуем изучить примеры скриптов – в них есть большая часть решения!

Write text answer

✓ Totally right.

Correct answer from 13,935 learners
Total 44% of tries are correct

```
set xtics ("point 1, value "x1 x1, "point 2, value "x2 x2, "point 3, value "x3 x3)
```

Next step Solve again

[Your submissions](#) You got: 2 points

Рис. 1.30: Задание 3.6

Используя скрипт, он отображает график, меняет направление вращения на противоположное и ускоряет его в 2 раза. (рис. 1.31)

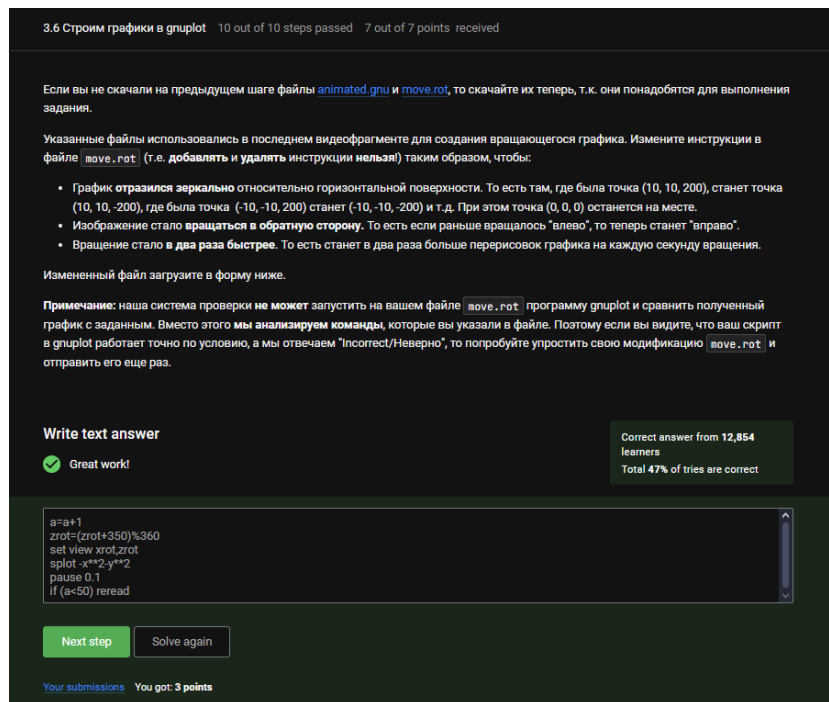


Рис. 1.31: Задание 3.6

1.7 Разное

Подходящие команды корректно изменяют права файла на `gwxrw-r--`. (рис. 1.32)

- Первый предоставляет разрешение на запись группе и пользователю, а также разрешение на выполнение пользователю
- Во-вторых, он предоставляет разрешение на запись и выполнение всем, затем удаляет разрешение на запись и выполнение у других и удаляет разрешение на выполнение из группы
- Последняя команда даёт пользователю и группе право на запись, а также пользователю — право на выполнение.

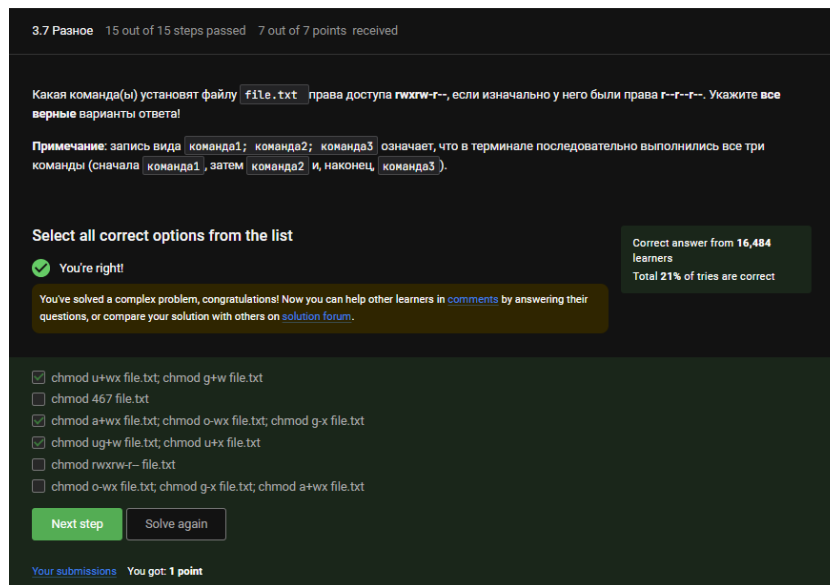


Рис. 1.32: Задание 3.7

Команды `sudo chown`, `sudo chmod` дают группе возможность записывать в директорию. (рис. 1.33)

доступа `rwxt-x` (владелец `root`, группа `root`). Таким образом никто кроме пользователя `root` не может ничего записывать в эту директорию, например, не может создавать файлы в ней.

После выполнения какой команды `user` из группы `group` всё-таки сможет создать файл внутри `dir`? Укажите **все верные** варианты ответов!

Примечание: считаем, что все команды выполняются от имени `user`, если явно не указано, что команда выполнена с `sudo`.

Примечание 2: мы выбрали пример с директорией, а не с файлом не случайно. Дело в том, что если создать при помощи `sudo` файл с правами `rw-r--r--` в директории, которая принадлежит пользователю, то возникнет любопытная ситуация. С одной стороны пользователь может удалить этот файл (т.к. ему разрешено удалять **все** файлы внутри его директории) и может прочитать его содержимое (т.к. право `"r"` у файла установлено для всех), с другой стороны он не может этот файл редактировать (т.к. право `"w"` у файла есть только для `root`). При этом некоторые "умные" редакторы, например, `vim` позволяют даже редактировать этот файл, но делают они это своеобразно: через удаление оригинала и создание копии уже с нужными правами (удалять мы можем, а раз можем читать, то и копию создать не сложно). Итого получается, что несмотря на права `rw-r--r--`, пользователь может сделать с этим файлом почти всё что угодно! В случае же, когда речь идет о директории созданной `root`, ситуация будет проще: пользователь сможет посмотреть её содержимое (у него есть право `"r"`), но удалять и создавать файлы в ней не сможет (права `"w"` у него нет). Важно отметить, что директории в `Linux` это в каком-то смысле **файлы**. Содержимое такого "файла" – это записи о файлах и поддиректориях этой директории (грубо говоря их **названия**). Таким образом, право `"r"` у директории дает возможность просматривать "записи", т.е. просматривать её состав. Право `"w"` у директории дает возможность удалять/добавлять новые "записи", т.е. удалять/создавать файлы/поддиректории в ней. На самом деле и это еще не всё. Существует так называемый **sticky bit** (атрибут файла или директории), выставление которого меняет описанное выше поведение. Файлы (или директории) с таким атрибутом сможет удалить только их владелец вне зависимости от прав, установленных у директории, в которой эти файлы (или директории) лежат!

Отдельное спасибо слушателю курса **Alexey Antipovsky** за помощь в оформлении **Примечания 2!**

Select all correct options from the list

☒ Yes!

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

☒ `sudo chown user:group dir`
☐ `sudo chmod g+w dir`
☒ `sudo chown user dir`
☐ `chmod o+w dir`
☐ `sudo chmod o+x dir`
☒ `sudo chmod o+w dir`

Correct answer from 14,683 learners
Total 15% of tries are correct

Рис. 1.33: Задание 3.7

`wc` позволяет подсчитать строки, слова и размер файла в байтах. (рис. 1.34)

3.7 Разное 15 out of 15 steps passed 7 out of 7 points received

Отметьте какие характеристики файла можно посчитать с использованием команды `wc`.

Select all correct options from the list

☒ Absolutely right.

You've solved a complex problem, congratulations! Now you can help other learners in [comments](#) by answering their questions, or compare your solution with others on [solution forum](#).

☐ Количество определенных букв (например, количество букв "A")
☐ Количество предложений
☒ Количество строк
☒ Размер файла в байтах
☒ Количество слов

Next step Solve again

Your submissions You got: 1 point

Correct answer from 17,158 learners
Total 21% of tries are correct

Рис. 1.34: Задание 3.7

Команда `du -s -h` показывает размер текущей директории в удобном формате.

(рис. 1.35)

3.7 Разное 15 out of 15 steps passed 7 out of 7 points received

Впишите в форму ниже команду, которая выведет сколько места на диске занимает текущая директория (при этом **размер** нужно вывести в **удобном для чтения формате** (например, вместо `2848 байт` надо вывести `2.8К`) и **больше** на экран выводить **ничего не нужно**). В команде указывайте **только необходимые** для выполнения задания **опции и аргументы**, лишних опций указывать не нужно!

Пример: если в текущей директории есть два файла по `888 Кбайт` и две поддиректории в каждой из которой лежит по файлу в `488 Кбайт`, то загаданная команда должна вывести на экран одно число: `2.4М` (также на экране может быть выведен еще и символ `.`, обозначающий, что это размер именно текущей директории).

Write text answer

✓ Totally right.

Correct answer from 16,381 learners
Total 53% of tries are correct

`du -s -h`

Next step Solve again

Your submissions You got 2 points

Рис. 1.35: Задание 3.7

Команда `mkdir dir{1,2,3}` — это самый короткий способ создания трёх директо-
рий одновременно. (рис. 1.36)

3.7 Разное 15 out of 15 steps passed 7 out of 7 points received

Впишите в форму ниже максимально короткую команду (т.е. в которой минимально возможное число символов), которая позволит создать в текущей директории 3 поддиректории с именами `dir1`, `dir2`, `dir3`.

Если вы придумали команду, которая выполняет эту задачу, а система проверки сообщает вам "Incorrect"/"Неверно", то скорее всего вы придумали не самую короткую команду из возможных!

Write text answer

✓ Absolutely right.

Correct answer from 16,720 learners
Total 40% of tries are correct

`mkdir dir{1,2,3}`

Next step Solve again

Your submissions You got 2 points

Рис. 1.36: Задание 3.7