

Отчёт по прохождению 3 этапа внешних курсов на stepik

Продвинутые темы

Верниковская Екатерина Андреевна

Содержание

1	Цель работы	6
2	Задание	7
3	Выполнение 3 этапа внешних курсов на stepik	8
4	Выводы	48
5	Список литературы	49

Список иллюстраций

3.1	Задание №1	8
3.2	Задание №2	9
3.3	Задание №3	10
3.4	Задание №4	10
3.5	Задание №5	11
3.6	Задание №6	12
3.7	Задание №7	12
3.8	Задание №8	13
3.9	Создание файла для задания №9	13
3.10	Программа для задания №9	13
3.11	Работа программы для задания №9	13
3.12	Задание №9	14
3.13	Задание №10	15
3.14	Задание №11	16
3.15	Создание файла для задания №12	16
3.16	Программа для задания №12	16
3.17	Работа программы для задания №12	17
3.18	Задание №12	17
3.19	Задание №13	19
3.20	Создание файла для задания №14	19
3.21	Программа для задания №14	20
3.22	Работа программы для задания №14	21
3.23	Задание №14 (1)	21
3.24	Задание №14 (2)	22
3.25	Задание №15	24
3.26	Задание №16	24
3.27	Задание №17	25
3.28	Создание файла для задания №18	25
3.29	Программа для задания №18	25
3.30	Работа программы для задания №18	26
3.31	Задание №18	26
3.32	Создание файла для задания №19	27
3.33	Программа для задания №19	27
3.34	Работа программы для задания №19	28
3.35	Задание №19 (1)	28
3.36	Задание №19 (2)	29
3.37	Создание файла для задания №20	30

3.38	Программа для задания №20	31
3.39	Работа программы для задания №20	31
3.40	Задание №20 (1)	32
3.41	Задание №20 (2)	33
3.42	Задание №21	34
3.43	Задание №22	35
3.44	Задание №23	35
3.45	Задание №24	36
3.46	Создание файла	36
3.47	Файл	36
3.48	Вывод ответа	36
3.49	Задание №25	37
3.50	Справка по команде sed	37
3.51	Задание №26	37
3.52	Создание файла для задания №27	38
3.53	Программа для задания №27	38
3.54	Файл input.txt	38
3.55	Проверка работы программы для задания №27	38
3.56	Открытие созданного файла	38
3.57	Файл edited.txt	38
3.58	Задание №27	39
3.59	Задание №28	40
3.60	Задание №29	40
3.61	Задание №30	41
3.62	Скачивание нужных файлов	41
3.63	Создание файла animated.gnu	41
3.64	Программа для animated.gnu	42
3.65	Создание файла move.rot	42
3.66	Программа для move.rot	42
3.67	Работа программы для задания №31	43
3.68	Задание №31	43
3.69	Задание №32	44
3.70	Задание №33	45
3.71	Задание №34	46
3.72	Команда du -h -s	46
3.73	Задание №35	46
3.74	Задание №36	47

Список таблиц

1 Цель работы

Ознакомиться с функционалом операционной системы Linux.

2 Задание

Посмотреть много видео и на основе полученной информации пройти тестовые задания.

3 Выполнение 3 этапа внешних курсов на stepik

Задание №1: так как я работала с редактором vim, я помню что надо сделать для того чтобы выйти из него (рис. 3.1)

Какую клавишу(и) нужно нажать на клавиатуре, чтобы выйти из редактора vim? Считайте, что вы только что открыли файл и вам сразу понадобилось выйти из редактора.

Выберите один вариант из списка

☒ Здорово, всё верно.

Верно решили 32 523 учащихся
Из всех попыток 69% верных

☒ ":", затем "q", затем "Enter"
☐ "Esc"
☐ "Q"
☐ ":", затем "q"
☐ "Ctrl", затем "x"

Следующий шаг Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.1: Задание №1

Задание №2:

- Точка считается “маленьким словом”, так что всего их 9: Strange_, is_here, ., 2, =, 2, ! и два лишних пробела
- Клавиша W перемещает курсор на один символ вправо, а точка (“.”) находится на конце строки. Чтобы переместить курсор на точку, необходимо нажать W 25 раз, так как строка содержит 25 символов (включая пробелы).

Однако если вы нажмете W 25 раз, курсор переместится за пределы строки. Клавиша w, с другой стороны, перемещает курсор на одно слово влево или вправо. Поскольку точка отделена от слова “YES!” пробелом, можно переместить курсор на нее, нажав w один раз.

(рис. 3.2)

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.

Выберите все подходящие ответы из списка

✓ Хорошие новости, верно!

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

Верно решили **25 385** учащихся
Из всех попыток **20%** верных

☒ В этой строке 9 "слов" (word)
☐ В этой строке 5 "слов" (word)
☐ В этой строке 12 "слов" (word)
☐ Нажимая только на w, нельзя переместить курсор на "."
☒ Нажимая только на W, нельзя переместить курсор на "."
☐ Чтобы попасть в конец строки, нужно одинаковое число нажатий, что на W, что на w

[Следующий шаг](#) [Решить снова](#)

Ваши решения Вы получили: **1 балл** из 1

Рис. 3.2: Задание №2

Задание №3:

- \$ — в конец текущей строки;
- w — на слово вправо;
- b — на слово влево;
- i — начать ввод перед курсором;
- р — вставка содержимого неименованного буфера под курсором;
- Р — вставка содержимого неименованного буфера перед курсором;
- уу (также Y) — копирование текущей строки в неименованный буфер;

- уу — копирование числа строк начиная с текущей в неименованный буфер;

Ответ: d2wwywPp; d2wwifour four ; ddithree four four four five

(рис. 3.3)

Предположим, что в текстовом файле записана одна единственная строка:

```
one two three four five
```

и вам нужно преобразовать её в строку

```
three four four four five
```

Какие(ой) из предложенных ниже **наборов нажатий клавиш** выполнят такое редактирование? В этих наборах нажатие на клавишу Esc обозначается как <Esc> (т.е. знаки "<" и ">" не несут отдельного смысла).

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

Выберите все подходящие ответы из списка

Верно. Верно решили 23 655 учащихся
Из всех попыток 16% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

☐ x2wwywPp
☒ d2wwywPp
☒ d2wwifour four <Esc>
☐ d2wwywpp
☒ ddithree four four four five<Esc>
☐ d2dwywPp

Следующий шаг Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.3: Задание №3

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. 3.4)

Предположим, что вы открыли файл в редакторе vim и хотите заменить в этом файле все строки, содержащие слово `Windows`, на такие же строки, но со словом `Linux`. Если в какой-то строке слово `Windows` встречается больше, чем один раз, то заменить на `Linux` в этой строке нужно **только самое первое** из этих слов.

Какую команду нужно ввести для этого в vim? Укажите необходимую команду целиком (т.е. включая ввод ":" в самом начале), однако нажатие на `Enter` после ввода команды обозначать никак **не нужно**.

Напишите текст

Прекрасный ответ. Верно решил 24 631 учащихся
Из всех попыток 57% верных

:%s/Windows/Linux

Следующий шаг Решить снова

[Ваши решения](#) Вы получили: 2 балла из 2

Рис. 3.4: Задание №4

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;

(рис. 3.5)

Мы совсем не рассказывали вам про третий режим работы vim – режим **выделения (Visual)**. Предлагаем вам ознакомиться с ним самостоятельно. Например, это можно сделать во время прохождения упражнений в vimtutor, который мы настоятельно рекомендуем вам для изучения vim!

Чтобы убедиться, что вы разобрались с этим режимом работы, отметьте, пожалуйста, **все верные** утверждения из списка ниже.

Подсказка: если вы не хотите проходить vimtutor целиком, то можете открыть его и поиском найти слово **"Visual"**. Вы попадете в задание, прохождение которого будет вполне достаточно, чтобы выполнить это задание.

Выберите все подходящие ответы из списка

✓ Отличное решение!

Верно решили **23 497** учащихся
Из всех попыток **29%** верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ Режим выделения открывается из любого другого режима по нажатию "v"
- ☒ Выйти из режима выделения можно, нажав клавишу Esc два раза
- ☐ Чтобы выйти из режима выделения, нужно ввести :q
- ☒ Когда вы находитесь в режиме выделения, внизу редактора горит надпись – VISUAL – (или – ВИЗУАЛЬНЫЙ РЕЖИМ –)
- ☒ В режиме выделения можно использовать команды d (удалить) и y (скопировать)
- ☒ Режим выделения открывается из нормального режима по нажатию "v"

[Следующий шаг](#) [Решить снова](#)

[Ваши решения](#) Вы получили: **2 балла** из 2

Рис. 3.5: Задание №5

Задание №6: только из набора C потому что у каждой оболочки свой буфер, который при выходе из нее будет записываться в файл истории (рис. 3.6)

Надеемся, что вы разобрались, что одну оболочку (например, `sh`) можно запустить из другой оболочки (например, из `bash`).

Предположим, что вы открыли терминал и у вас в нем запущена оболочка `bash`. Вы набираете в ней команды `A1`, `A2`, `A3`, а затем запускаете оболочку `sh`. В этой оболочке вы набираете команды `B1`, `B2`, `B3` и запускаете оболочку `bash`. И, наконец, в этой последней оболочке вы набираете команды `C1`, `C2`, `C3`. Если теперь вы попробуете при помощи стрелочек вверх/вниз перемещаться по истории набранных команд, то команды из какого набора(ов) будут появляться?

Выберите один вариант из списка

✓ Правильно.

Верно решили 30 266 учащихся
Из всех попыток 65% верных

- ☐ Только из набора B
- ☐ Из наборов A и C
- ☐ Только из набора A
- ☐ Никакие команды появляться не будут
- ☒ Только из набора C

Следующий шаг

Решить снова

Ваши решения Вы получили: 1 балл из 1

Рис. 3.6: Задание №6

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 (рис. 3.7)

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [script1.sh](#), [script2.sh](#).

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

```
#!/bin/bash
cd /home/bi/
touch file1.txt
cd /home/bi/Desktop/
```

Как будет выглядеть абсолютный путь до созданного файла `file1.txt` по окончании работы скрипта?

Выберите один вариант из списка

✓ Отлично!

Верно решили 29 905 учащихся
Из всех попыток 76% верных

- ☐ `/home/bi/Desktop/file1.txt`
- ☒ `/home/bi/file1.txt`
- ☐ `/home/bi/Documents/file1.txt`
- ☐ Ничак (файла `file1.txt` не будет существовать после завершения работы скрипта)

Следующий шаг

Решить снова

Ваши решения Вы получили: 1 балл из 1

Рис. 3.7: Задание №7

Задание №8: в имени только буквы, цифры и подчеркивание. Больше никаких символов! (рис. 3.8)

Вы можете скачать и изучить скрипты, которые мы показали в видеофрагменте: [variables1.sh](#), [variables2.sh](#).

Какие из представленных ниже строк **могут** быть именами переменных в bash? Выберите **все** подходящие варианты!

Подсказка: если все варианты ответов являются неверными, то не отмечайте ни один из них и нажимайте кнопку "Отправить"/"Submit".

Выберите все подходящие ответы из списка

✓ Хорошая работа.

Верно решили 27 188 учащихся
Из всех попыток 25% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ var l able
- ☐ var-i-able
- ☐ 123variable
- ☐ var@iable
- ☒ _variable
- ☐ variab\$\$le
- ☐ var.l.able

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.8: Задание №8

Задание №9: создаю файл sh, пишу программу и проверяю работу командного файла (рис. 3.9), (рис. 3.10), (рис. 3.11), (рис. 3.12)

```
eavernikovskaya@ubuntu-katerok:~$ touch task1.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task1.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.9: Создание файла для задания №9

```
task1.sh
~/
1 #!/bin/bash
2 var1=$1
3 var2=$2
4
5 echo "Arguments are: \${1}=${var1} \${2}=${var2}"
```

Рис. 3.10: Программа для задания №9

```
eavernikovskaya@ubuntu-katerok:~$ bash task1.sh 1 3
Arguments are: $1=1 $2=3
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.11: Работа программы для задания №9

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

Напишите программу. Тестируется через stdin → stdout

✓ Так точно!

Верно решили 25 053 учащихся
Из всех попыток 41% верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 #!/bin/bash
2 var1=$1
3 var2=$2
4
5 echo "Arguments are: \${1=$var1} \${2=$var2}"
6
7
8
9
10
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 3 балла из 3

Рис. 3.12: Задание №9

Программа для задания №9:

```
#!/bin/bash
```

```
var1=$1
```

```
var2=$2
```

```
echo "Arguments are: \${1=$var1} \${2=$var2}"
```

Задание №10: на скрине всё видно (рис. 3.13)

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 вписать 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.

Выберите все подходящие ответы из списка

☒ Верно. Так держать!

Верно решили **23 158** учащихся
Из всех попыток **16%** верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ `-n $1`
- ☒ `5 -ge 5`
- ☒ `-n $0`
- ☒ `-s $0`
- ☒ `! (4 -le 3)`
- ☒ `-z ""`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **1 балл** из 1

Рис. 3.13: Задание №10

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
(рис. 3.14)

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**.

Выберите один вариант из списка

☒ Хорошие новости, верно!

Верно решили 25 138 учащихся
Из всех попыток 64% верных

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

Следующий шаг

Решить снова

Ваши решения Вы получили: 1 балл из 1

Рис. 3.14: Задание №11

Задание №12: создаю файл sh, пишу программу и проверяю работу командного файла (рис. 3.15), (рис. 3.16), (рис. 3.17), (рис. 3.18)

```
eavernikovskaya@ubuntu-katerok:~$ touch task2.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task2.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.15: Создание файла для задания №12

```
task2.sh
~/
1#!/bin/bash
2
3if [[ $1 -eq 1 ]]; then
4  echo "$1 student"
5elif [[ $1 -gt 1 && $1 -le 4 ]]; then
6  echo "$1 students"
7elif [[ $1 -ge 5 ]]; then
8  echo "A lot of students"
9else
10  echo "No students"
11fi
```

Рис. 3.16: Программа для задания №12


```
eavernikovskaya@ubuntu-katerok:~$ bash task2.sh
No students
eavernikovskaya@ubuntu-katerok:~$ bash task2.sh 3
3 students
eavernikovskaya@ubuntu-katerok:~$ bash task2.sh 8
A lot of students
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.17: Работа программы для задания №12

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](#).

Напишите программу. Тестируется через stdin → stdout

✓ Прекрасный ответ.

Верно решили **23 310** учащихся
Из всех попыток **38%** верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 #!/bin/bash
2
3 if [[ $1 -eq 1 ]]; then
4     echo "1 student"
5 elif [[ $1 -gt 1 && $1 -le 4 ]]; then
6     echo "$1 students"
7 elif [[ $1 -ge 5 ]]; then
8     echo "A lot of students"
9 else
10    echo "No students"
11 fi
12
13
14
15
16
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **3 балла** из 3

Рис. 3.18: Задание №12

Программа для задания №12:

```
#!/bin/bash
```

```
if [[ $1 -eq 1 ]]; then
    echo "$1 student"
elif [[ $1 -gt 1 && $1 -le 4 ]]; then
    echo "$1 students"
elif [[ $1 -ge 5 ]]; then
    echo "A lot of students"
else
    echo "No students"
fi
```

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

(рис. 3.19)

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"?

Выберите один вариант из списка

✓ Отлично!

Верно решили **24 582** учащихся
Из всех попыток **45%** верных

- ☐ 3 раза "start" и ни разу "finish"
- ☒ 5 раз "start" и 4 раза "finish"
- ☐ 5 раз "start" и 2 раза "finish"
- ☐ 3 раза "start" и 3 раза "finish"

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **1 балл** из 1

Рис. 3.19: Задание №13

Задание №14: создаю файл sh, пишу программу и проверяю работу командного файла (рис. 3.20), (рис. 3.21), (рис. 3.22), (рис. 3.23), (рис. 3.24)

```
eavernikovskaya@ubuntu-katerok:~$ touch task3.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task3.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.20: Создание файла для задания №14

```
task3.sh
~/
1#!/bin/bash
2
3child=16
4adult=25
5stdout=0
6
7while [[ $stdout != 1 ]]
8do
9    echo "enter your name: "
10   read name
11   if [[ (-z $name) || ($name = 0) ]] ;then
12       echo "bye"
13       stdout=1
14   elif [[ -n $name ]]; then
15       while [[ $stdout != 1 ]] ;do
16           echo "enter your age: "
17           read age
18           if [[ ($age -eq 0) || (-z $age) ]] ;then
19               echo "bye"
20               stdout=1
21           elif [[ $age -le $child ]] ;then
22               echo "$name, your group is child"
23           elif [[ $age -gt $adult ]] ; then
24               echo "$name, your group is adult" ;else
25               if [[ ($age -ge 17) && ($age -le 25) ]] ;then
26                   echo "$name, your group is youth" ;fi
27               fi ;break
28           done ;fi
29done
```

Рис. 3.21: Программа для задания №14

```
eavernikovskaya@ubuntu-katerok:~$ bash task3.sh
enter your name:
katya
enter your age:
18
katya, your group is youth
enter your name:
nastya
enter your age:
5
nastya, your group is child
enter your name:
masha
enter your age:
30
masha, your group is adult
enter your name:

bye
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.22: Работа программы для задания №14

Напишите скрипт на `bash`, который будет определять в какую возрастную группу попадают пользователи. При запуске скрипт должен вывести сообщение `"enter your name:"` и ждать от пользователя ввода имени (используйте `read`, чтобы прочитать его). Когда имя введено, то скрипт должен написать `"enter your age:"` и ждать ввода возраста (опять нужен `read`). Когда возраст введен, скрипт пишет на экран `"<Имя>, your group is <группа>"`, где `<группа>` определяется на основе возраста по следующим правилам:

- младше либо равно 16: `"child"`,
- от 17 до 25 (включительно): `"youth"`,
- старше 25: `"adult"`.

После этого скрипт опять выводит сообщение `"enter your name:"` и всё начинается по новой (бесконечный цикл). Если в какой-то момент работы скрипта будет введено `пустое имя` или `возраст 0`, то скрипт должен написать на экран `"bye"` и закончить свою работу (выход из цикла!).

Примеры корректной работы скрипта:

№1

```
./script.sh
enter your name:
Egor
enter your age:
16
Egor, your group is child
enter your name:
Elena
enter your age:
0
bye
```

№2:

```
./script.sh
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](#).

Рис. 3.23: Задание №14 (1)

Напишите программу. Тестируется через stdin → stdout

✓ Хорошая работа.

Верно решили 21 670 учащихся
Из всех попыток 23% верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить свое решение с другими на [форуме решений](#).

```
1 # put your shell (bash) code here
2 #!/bin/bash
3
4 child=16
5 adult=25
6 stdout=0
7
8 while [[ $stdout != 1 ]]
9 do
10     echo "enter your name: "
11     read name
12     if [[ (-z $name) || ($name = 0) ]];then
13         echo "bye"
14         stdout=1
15     elif [[ -n $name ]]; then
16         while [[ $stdout != 1 ]];do
17             echo "enter your age: "
18             read age
19             if [[ ($age -eq 0) || (-z $age) ]]; then
20                 echo "bye"
21                 stdout=1
22             elif [[ $age -le $child ]]; then
23                 echo "$name, your group is child"
24             elif [[ $age -gt $adult ]]; then
25                 echo "$name, your group is adult" ;else
26                 if [[ ($age -ge 17) && ($age -le 25) ]]; then
27                     echo "$name, your group is youth" ;fi
28                 fi ;break
29             done ;fi
30         done
31     done
32
33
34
```

Следующий шаг Решить снова

[Ваши решения](#) Вы получили: 4 балла из 4

Рис. 3.24: Задание №14 (2)

Программа для задания №14:

```
#!/bin/bash
```

```
child=16
```

```
adult=25
```

```
stdout=0
```

```
while [[ $stdout != 1 ]]
```

```
do
```

```
    echo "enter your name: "
```

```
    read name
```

```
    if [[ (-z $name) || ($name = 0) ]] ;then
```

```

    echo "bye"
    stdout=1
elif [[ -n $name ]]; then
    while [[ $stdout != 1 ]] ;do
        echo "enter your age: "
        read age
        if [[ ($age -eq 0) || (-z $age) ]] ;then
            echo "bye"
            stdout=1
        elif [[ $age -le $child ]] ;then
            echo "$name, your group is child"
        elif [[ $age -gt $adult ]] ; then
            echo "$name, your group is adult" ;else
            if [[ ($age -ge 17) && ($age -le 25) ]] ;then
                echo "$name, your group is youth" ;fi
            fi ;break
        done ;fi
done

```

Задание №15: на скрине всё видно (рис. 3.25)

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, 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)!

Выберите все подходящие ответы из списка

✔ Отличное решение!

Верно решили 22 116 учащихся

Из всех попыток 20% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ `let a = a + b`
- ☐ `let "a+=b"`
- ☐ `a+=b`
- ☒ `let "a=${a+$b}"`
- ☒ `let a=a+b`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.25: Задание №15

Задание №16: выведет путь до директории, в которую мы перешли, так как “`pwd`” — это команда (рис. 3.26)

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?

Выберите один вариант из списка

✔ Так точно!

Верно решили 23 677 учащихся

Из всех попыток 51% верных

- ☐ Код возврата команды `pwd` (0 в случае успешного выполнения и не 0 в случае ошибок)
- ☐ ``pwd``
- ☒ `/home/bi`
- ☐ `pwd`
- ☐ `/home/bi/Documents`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.26: Задание №16

Задание №17: на скрине всё видно (рис. 3.27)

Мы рассказали, что можно проверить код возврата внешней программы прямо в конструкции `if` при помощи `if `program options arguments`` (действия внутри `if` выполнятся, 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 все-таки запустить программу `program`, которая пишет что-то в `stdout` и потом выполнить какие-то действия если ее 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 (`"`).

Выберите все подходящие ответы из списка

☒ Верно. Так держать!

Верно решили 21 426 учащихся

Из всех попыток 20% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

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

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.27: Задание №17

Задание №18: создаю файл `sh`, пишу программу и проверяю работу командного файла (рис. 3.28), (рис. 3.29), (рис. 3.30), (рис. 3.31)

```
eavernikovskaya@ubuntu-katerok:~$ touch task4.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task4.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.28: Создание файла для задания №18

```
task4.sh
#!/bin/bash
2
3 counter () # takes one argument
4 {
5     local let "c1+=${1}"
6     let "c2+=${1}*2"
7 }
8
9 for i in {1..10}
10 do
11     counter $i
12 done
13
14 echo "counters are $c1 and $c2"
```

Рис. 3.29: Программа для задания №18

```
eavernikovskaya@ubuntu-katerok:~$ bash task4.sh
counters are  and 110
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.30: Работа программы для задания №18

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+=${1}"
    let "c2+=${1}*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.

Напишите текст

✓ Так точно!

Верно решили **20 009** учащихся
Из всех попыток **28%** верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

counters are and 110

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **2 балла** из 2

Рис. 3.31: Задание №18

Программа для задания №18:

```
#!/bin/bash
```

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

```
for i in {1..10}
```

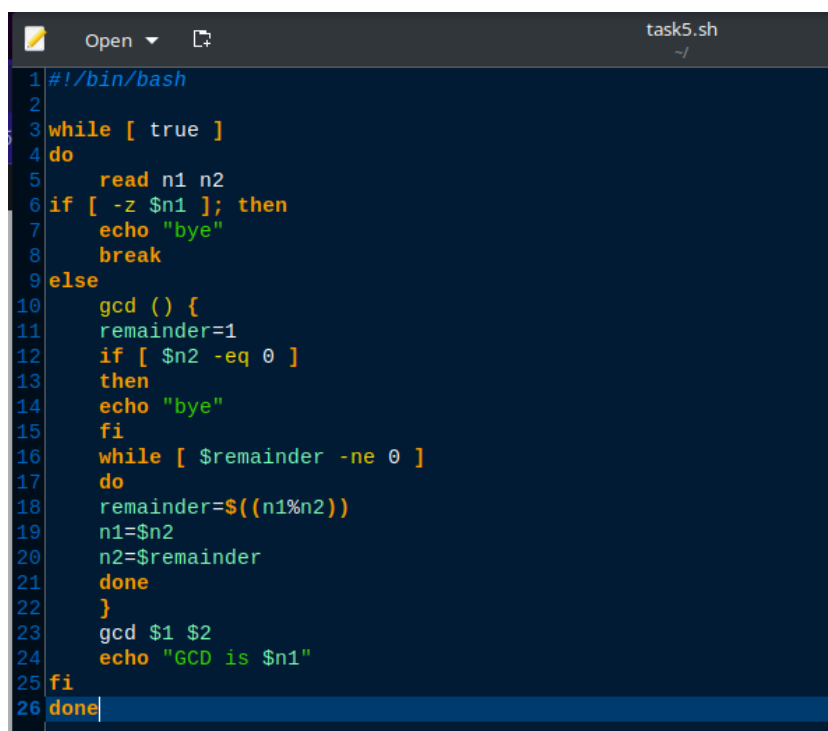
```
do
    counter $i
done

echo "counters are $c1 and $c2"
```

Задание №19: создаю файл sh, пишу программу и проверяю работу командного файла (рис. 3.32), (рис. 3.33), (рис. 3.34), (рис. 3.35), (рис. 3.36)

```
eavernikovskaya@ubuntu-katerok:~$ touch task5.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task5.sh
```

Рис. 3.32: Создание файла для задания №19



The screenshot shows a terminal window with a dark background and light-colored text. The window title is "task5.sh". The script content is as follows:

```
1#!/bin/bash
2
3while [ true ]
4do
5    read n1 n2
6    if [ -z $n1 ]; then
7        echo "bye"
8        break
9    else
10       gcd () {
11           remainder=1
12           if [ $n2 -eq 0 ]
13           then
14               echo "bye"
15           fi
16           while [ $remainder -ne 0 ]
17           do
18               remainder=$((n1%n2))
19               n1=$n2
20               n2=$remainder
21           done
22       }
23       gcd $1 $2
24       echo "GCD is $n1"
25   fi
26done
```

Рис. 3.33: Программа для задания №19

```
eavernikovskaya@ubuntu-katerok:~$ bash task5.sh
15 10
GCD is 5
4 2
GCD is 2
7 3
GCD is 1

bye
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.34: Работа программы для задания №19

Напишите скрипт на `bash`, который будет искать наибольший общий делитель ([НОД](#), greatest common divisor, GCD) двух чисел. При запуске ваш скрипт не должен ничего писать на экран, а просто ждет ввода двух натуральных чисел через пробел (для этого можно использовать `read` и указать ему две переменные – см. пример в видеофрагменте). После ввода чисел скрипт считает их НОД и выводит на экран сообщение `"GCD is <посчитанное значение>"`, например, для чисел 15 и 25 это будет `"GCD is 5"`. После этого скрипт опять входит в режим ожидания двух натуральных чисел. Если в какой-то момент работы пользователь ввел вместо этого пустую строку, то нужно написать на экран `"bye"` и закончить свою работу.

Вычисление НОД несложно реализовать с помощью [алгоритма Евклида](#). Вам нужно написать функцию `gcd`, которая принимает на вход два аргумента (назовем их `M` и `N`). Если аргументы равны, то мы нашли НОД – он равен `M` (или `N`), нужно вывести соответствующее сообщение на экран (см. выше). Иначе нужно сравнить аргументы между собой. Если `M` больше `N`, то запускаем ту же функцию `gcd`, но в качестве первого аргумента передаем `(M-N)`, а в качестве второго `N`. Если же наоборот, `M` меньше `N`, то запускаем функцию `gcd` с первым аргументом `M`, а вторым `(N-M)`.

Пример корректной работы скрипта:

```
./script.sh
10 15
gcd is 5
7 3
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](#).

Рис. 3.35: Задание №19 (1)

Напишите программу. Тестируется через stdin → stdout

✓ Отлично!

Верно решили 18 148 учащихся
Из всех попыток 35% верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 # put your shell (bash) code here
2 #!/bin/bash
3
4 while [ true ]
5 do
6     read n1 n2
7     if [ -z $n1 ]; then
8         echo "bye"
9         break
10    else
11        gcd () {
12            remainder=1
13            if [ $n2 -eq 0 ]
14            then
15                echo "bye"
16            fi
17            while [ $remainder -ne 0 ]
18            do
19                remainder=$((n1%n2))
20                n1=$n2
21                n2=$remainder
22            done
23        }
24        gcd $1 $2
25        echo "GCD is $n1"
26    fi
27 done
28
29
30
31
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 4 балла из 4

Рис. 3.36: Задание №19 (2)

Программа для задания №19:

```
#!/bin/bash
```

```
while [ true ]
```

```
do
```

```
    read n1 n2
```

```
if [ -z $n1 ]; then
```

```
    echo "bye"
```

```
    break
```

```
else
```

```
    gcd () {
```

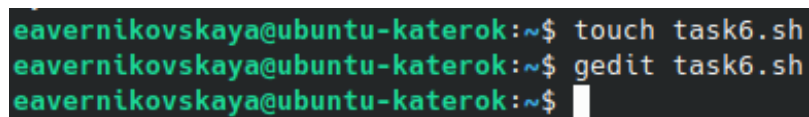
```
        remainder=1
```

```
        if [ $n2 -eq 0 ]
```

```
        then
```

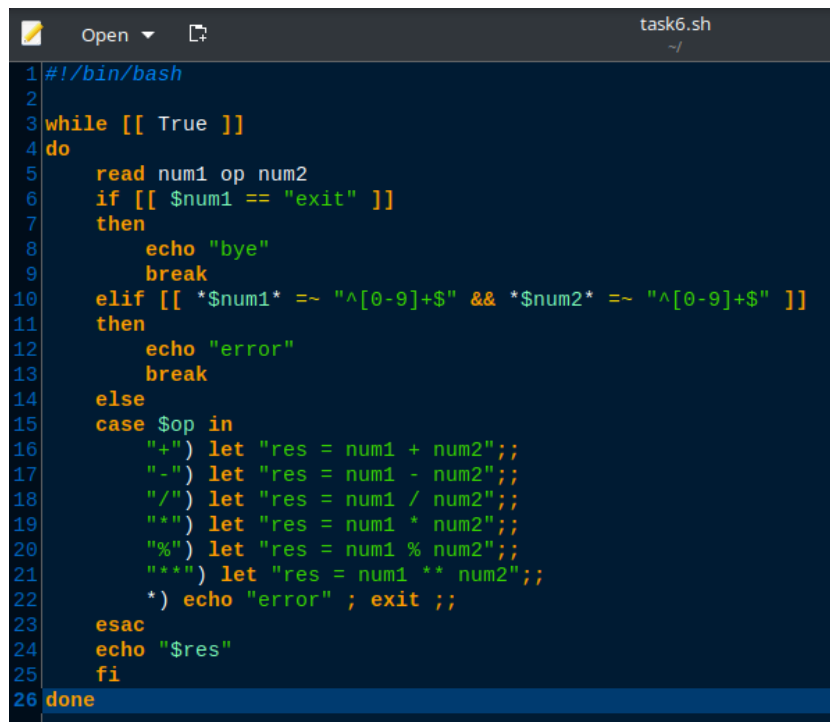
```
    echo "bye"
fi
while [ $remainder -ne 0 ]
do
    remainder=$((n1%n2))
    n1=$n2
    n2=$remainder
done
}
gcd $1 $2
echo "GCD is $n1"
fi
done
```

Задание №20: создаю файл sh, пишу программу и проверяю работу командного файла (рис. 3.37), (рис. 3.38), (рис. 3.39), (рис. 3.40), (рис. 3.41)

A terminal window with a dark background and light green text. The prompt is 'eavernikovskaya@ubuntu-katerok:~\$'. The first command is 'touch task6.sh'. The second command is 'gedit task6.sh'. The third line shows the prompt again with a cursor, indicating the command has been executed.

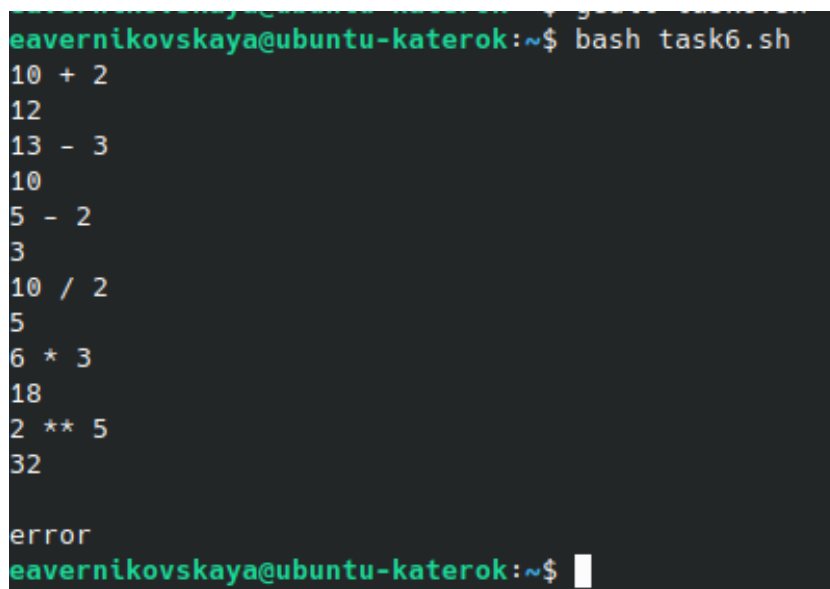
```
eavernikovskaya@ubuntu-katerok:~$ touch task6.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task6.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.37: Создание файла для задания №20



```
1#!/bin/bash
2
3while [[ True ]]
4do
5    read num1 op num2
6    if [[ $num1 == "exit" ]]
7    then
8        echo "bye"
9        break
10    elif [[ *$num1* =~ "^[0-9]+$" && *$num2* =~ "^[0-9]+$" ]]
11    then
12        echo "error"
13        break
14    else
15        case $op in
16            "+") let "res = num1 + num2" ;;
17            "-") let "res = num1 - num2" ;;
18            "/" ) let "res = num1 / num2" ;;
19            "*" ) let "res = num1 * num2" ;;
20            "%" ) let "res = num1 % num2" ;;
21            "**" ) let "res = num1 ** num2" ;;
22            *) echo "error" ; exit ;;
23        esac
24        echo "$res"
25    fi
26done
```

Рис. 3.38: Программа для задания №20



```
eavernikovskaya@ubuntu-katerok:~$ bash task6.sh
10 + 2
12
13 - 3
10
5 - 2
3
10 / 2
5
6 * 3
18
2 ** 5
32
error
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.39: Работа программы для задания №20

Напишите **калькулятор** на `bash`. При запуске ваш скрипт должен ожидать ввода пользователем команды (при этом на экран выводить ничего не нужно). Команды могут быть трех типов:

1. Слово **"exit"**. В этом случае скрипт должен вывести на экран слово **"bye"** и завершить работу.
2. **Три аргумента через пробел** – первый операнд (целое число), операция (одна из **"+"**, **"-"**, **"*"**, **"/"**, **"%"**, **"**"**) и второй операнд (целое число). В этом случае нужно произвести указанную операцию над заданными числами и вывести результат на экран. После этого переходим в режим ожидания новой команды.
3. **Любая другая команда** из одного аргумента или из трех аргументов, но с операцией не из списка. В этом случае нужно вывести на экран слово **"error"** и завершить работу.

Чтобы проверить работу скрипта, вы можете записать сразу несколько команд в файл и передать его скрипту на `stdin` (т.е. выполнить `./script.sh < input.txt`). В этом случае он должен вывести сразу все ответы на экран.

Например, если входной файл будет следующего содержания:

```
10 + 1
2 ** 10
exit
```

то на экране будет:

```
11
1024
bye
```

Если же на вход поступит следующий файл:

```
3 - 5
2/10
exit
```

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](#).

Рис. 3.40: Задание №20 (1)

Напишите программу. Тестируется через stdin → stdout

✓ Верно. Так держаты!

Верно решили 16 980 учащихся
Из всех попыток 36% верных

Теперь вам доступен [Форум решений](#), где вы можете сравнить свое решение с другими или спросить совета.

```
1 # put your shell (bash) code here
2 #!/bin/bash
3
4 while [[ True ]]
5 do
6     read num1 op num2
7     if [[ $num1 == "exit" ]]
8     then
9         echo "bye"
10        break
11    elif [[ *$num1* =~ "^[0-9]+$" && *$num2* =~ "^[0-9]+$" ]]
12    then
13        echo "error"
14        break
15    else
16        case $op in
17            "+") let "res = num1 + num2";;
18            "-") let "res = num1 - num2";;
19            "/" let "res = num1 / num2";;
20            "*" let "res = num1 * num2";;
21            "%" let "res = num1 % num2";;
22            "**") let "res = num1 ** num2";;
23            *) echo "error" ; exit ;;
24        esac
25        echo "$res"
26    fi
27 done
28
29
30
31
```

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 5 баллов из 5

Рис. 3.41: Задание №20 (2)

Программа для задания №20:

```
#!/bin/bash
```

```
while [[ True ]]
```

```
do
```

```
    read num1 op num2
```

```
    if [[ $num1 == "exit" ]]
```

```
    then
```

```
        echo "bye"
```

```
        break
```

```
    elif [[ *$num1* =~ "^[0-9]+$" && *$num2* =~ "^[0-9]+$" ]]
```

```
    then
```

```
        echo "error"
```

```
        break
```

```

else
case $op in
    "+") let "res = num1 + num2";;
    "-" ) let "res = num1 - num2";;
    "/" ) let "res = num1 / num2";;
    "*" ) let "res = num1 * num2";;
    "%" ) let "res = num1 % num2";;
    "**") let "res = num1 ** num2";;
    *) echo "error" ; exit ;;
esac
echo "$res"
fi
done

```

Задание №21: -iname ищет без учета регистра, а -name в точности как в запросе. Звездочка стоит после слова, значит после слова бесконечное количество символов, до слова символов не должно быть. (рис. 3.42)

Пусть в директории /home/bi лежат файлы Star_Wars.avi, star_trek OST.mp3, STARS.txt, stardust.mpeg, Eddard_Stark_biography.txt .

Отметьте все файлы, которые **найдет** команда `find /home/bi -iname "star*"`, но **НЕ найдет** команда `find /home/bi -name "star*"` ?

Выберите все подходящие ответы из списка

Верно решили 20 547 учащихся
Из всех попыток 36% верных

✓ Правильно.

- ☒ Star_Wars.avi
- ☒ STARS.txt
- ☐ star_trek OST.mp3
- ☐ Eddard_Stark_biography.txt
- ☐ stardust.mpeg

[Следующий шаг](#) [Решить снова](#)

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.42: Задание №21

Задание №22: на скрине всё видно (рис. 3.43)

Задание на понимание работы опций `-path` и `-name` команды `find`. Отметьте **все верные** утверждения из перечисленных ниже.

Выберите все подходящие ответы из списка

✓ Прекрасный ответ.

Верно решили 18 450 учащихся
Из всех попыток 22% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☒ Если заменить в команде поиска `-name`, на `-path`, то результат поиска иногда может остаться таким же
- ☐ Опция `-path` аналогична `-name`, но игнорирует размер букв (строчные/прописные) в имени файла
- ☐ Если заменить в команде поиска `-name`, на `-path`, то результат поиска всегда останется неизменным
- ☐ Опция `-path` используется только для поиска директорий, а `-name` только для поиска файлов
- ☐ Опции `-path` и `-name` всегда работают одинаково

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.43: Задание №22

Задание №23: текущий каталог - это `depth=1`, а остальное считается просто:
`/home/bi` -> `depth=1` `/home/bi/dir1` -> `depth=2` `/home/bi/dir1/dir2` -> `depth=3` etc.
(рис. 3.44)

Предположим, что в директории `/home/bi/` есть следующая структура файлов и поддиректорий:

```
/home/bi/  
├── dir1  
│   ├── file1  
│   └── dir2  
│       ├── file2  
│       └── dir3  
│           └── file3
```

Какие(ой) из трех файлов (`file1`, `file2`, `file3`) будут найдены по команде `find /home/bi -mindepth 2 -maxdepth 3 -name "file*"`?

Выберите один вариант из списка

✓ Верно.

Верно решили 20 711 учащихся
Из всех попыток 41% верных

- ☐ Только `file1`
- ☒ Все кроме `file3`
- ☐ Только `file2`
- ☐ Ни один файл найден не будет
- ☐ Все три файла

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.44: Задание №23

Задание №24: на скрине всё видно (рис. 3.45)

Задание на понимание работы опций `-A`, `-B` и `-C` команды `grep`. Пусть у вас есть файл `file.txt` из 10 строк, причем в каждой строке есть слово `"word"`. Если вы выполните на этом файле команды:

```
grep "word" file.txt > results.txt
grep -A 1 "word" file.txt > results.txt
grep -B 1 "word" file.txt > results.txt
grep -C 1 "word" file.txt > results.txt
```

то какая(ие) из них создаст файл `results.txt` наибольшего размера?

Выберите один вариант из списка

☒ Отлично!

Верно решили 20 237 учащихся
Из всех попыток 41% верных

- ☐ `grep -A 1 "word" file.txt > results.txt` и `grep -B 1 "word" file.txt > results.txt`
- ☐ `grep -A 1 "word" file.txt > results.txt`
- ☐ Все, кроме `grep "word" file.txt > results.txt`
- ☐ `grep -C 1 "word" file.txt > results.txt`
- ☒ `results.txt` будет одинакового размера во всех случаях

Следующий шаг

Решить снова

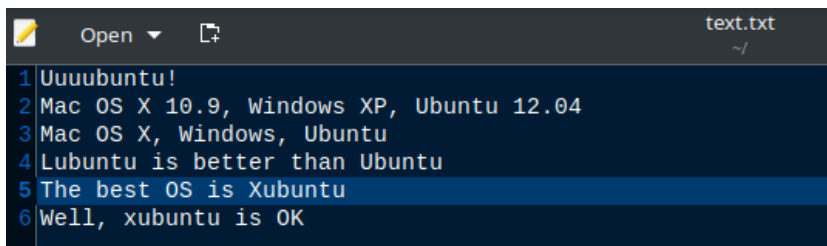
Ваши решения Вы получили: 1 балл из 1

Рис. 3.45: Задание №24

Задание №25: создаю файл `txt` и записываю в него строки, показанные среди вариантов ответа. Далее использую команду `grep -E "[xklXKL]?[uU]buntu$" text.txt` и получаю ответ (рис. 3.46), (рис. 3.47), (рис. 3.48), (рис. 3.49)

```
eavernikovskaya@ubuntu-katerok:~$ touch text.txt
eavernikovskaya@ubuntu-katerok:~$ gedit text.txt
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.46: Создание файла



The screenshot shows a text editor window titled 'text.txt' with the following content:

```
1Uuuubuntu!
2Mac OS X 10.9, Windows XP, Ubuntu 12.04
3Mac OS X, Windows, Ubuntu
4Lubuntu is better than Ubuntu
5The best OS is Xubuntu
6Well, xubuntu is OK
```

Рис. 3.47: Файл

```
eavernikovskaya@ubuntu-katerok:~$ grep -E "[xklXKL]?[uU]buntu$" text.txt
Mac OS X, Windows, Ubuntu
Lubuntu is better than Ubuntu
The best OS is Xubuntu
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.48: Вывод ответа

Предположим, что в файле `text.txt` записаны строки, показанные среди вариантов ответа. Отметьте только те из них, которые выведет на экран команда `grep -E "[xkLXKL]?[uu]buntu$" text.txt`.

Выберите все подходящие ответы из списка

☒ Так точно!

Верно решили **18 768** учащихся
Из всех попыток **23%** верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ Uuuubuntu!
- ☐ Mac OS X 10.9, Windows XP, Ubuntu 12.04
- ☒ Mac OS X, Windows, Ubuntu
- ☒ Ubuntu is better than Ubuntu
- ☒ The best OS is Xubuntu
- ☐ Well, xubuntu is OK

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **2 балла** из 2

Рис. 3.49: Задание №25

Задание №26: опция `-n` отключает автоматическую печать, что означает, что строки, которые вы специально не указываете на печать, не печатаются, а строки, которые вы явно указываете на печать (например, с помощью `p`), печатаются только один раз (рис. 3.50), (рис. 3.51)

```
-n, --quiet, --silent
suppress automatic printing of pattern space
```

Рис. 3.50: Справка по команде `sed`

Что произойдет, если в команде `sed -n "[a-z]*/p" text.txt` не указывать опцию `-n` ?

Выберите один вариант из списка

☒ Так точно!

Верно решили **19 784** учащихся
Из всех попыток **39%** верных

- ☐ Появится сообщение об ошибке
- ☒ Каждая строка будет выведена два раза
- ☐ На экран ничего не напечатается
- ☐ Будут выведены все строки файла `text.txt`, в которых есть только большие буквы латинского алфавита

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **1 балл** из 1

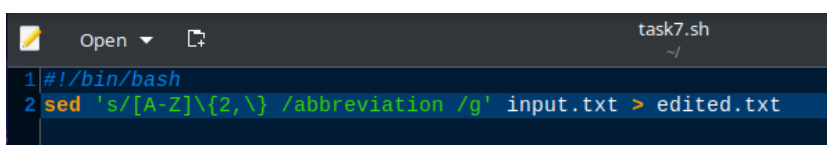
Рис. 3.51: Задание №26

Задание №27: создаю файл `sh` и пишу нужную программу. Далее создаю файл `txt` и записываю в него строки из примера. Далее проверяю работу командного

файла. В процессе создался файл edited.txt, где абривиатуры заменены на слово abbreviation (рис. 3.52), (рис. 3.53), (рис. 3.54), (рис. 3.55), (рис. 3.56), (рис. 3.57), (рис. 3.58)

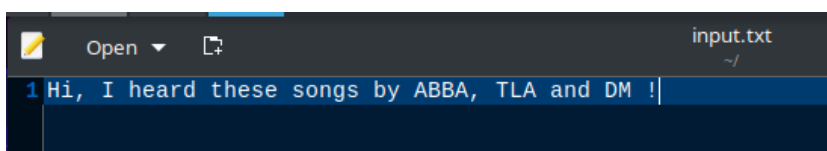
```
eavernikovskaya@ubuntu-katerok:~$ touch task7.sh
eavernikovskaya@ubuntu-katerok:~$ gedit task7.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.52: Создание файла для задания №27



```
task7.sh
~/
1#!/bin/bash
2sed 's/[A-Z]\{2,\} /abbreviation /g' input.txt > edited.txt
```

Рис. 3.53: Программа для задания №27



```
input.txt
~/
1Hi, I heard these songs by ABBA, TLA and DM !
```

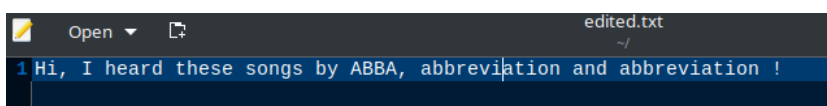
Рис. 3.54: Файл input.txt

```
eavernikovskaya@ubuntu-katerok:~$ bash task7.sh
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.55: Проверка работы программы для задания №27

```
eavernikovskaya@ubuntu-katerok:~$ gedit edited.txt
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.56: Открытие созданного файла



```
edited.txt
~/
1Hi, I heard these songs by ABBA, abbreviation and abbreviation !
```

Рис. 3.57: Файл edited.txt

Запишите в форму ниже инструкцию `sed`, которая заменит все "аббревиатуры" в файле `input.txt` на слово "abbreviation" и запишет 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).

Напишите текст

✓ Отличное решение!

Верно решили 16 632 учащихся
Из всех попыток 34% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в комментариях, отвечая на их вопросы, или сравнить своё решение с другими на форуме решений.

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

Следующий шаг

Решить снова

Ваши решения Вы получили: 3 балла из 3

Рис. 3.58: Задание №27

Программа для задания №27:

```
#!/bin/bash
```

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

Задание №28: `-p`, `--persist` позволяет окнам графиков сохраняться после выхода из основной программы `gnuplot` (рис. 3.59)

Вы можете скачать и попробовать применить `gnuplot` к файлу, который мы показали в видеофрагменте: [authors.txt](#).

Какую опцию нужно указать при запуске `gnuplot`, чтобы при его закрытии не были автоматически закрыты и все нарисованные в нём графики?

Выберите один вариант из списка

Верно решили 18 785 учащихся
Из всех попыток 51% верных

☒ Правильно, молодец!

- ☒ `-p, --persist`
- ☐ Такой опции не существует
- ☐ `-s, --show-plots-after-exit`
- ☐ `-raise`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.59: Задание №28

Задание №29: на скрине всё видно (рис. 3.60)

Предположим у вас есть файл `data.csv` с двумя столбцами по 10 чисел в каждом. В первой строке не записаны названия столбцов, т.е. ряды данных начинаются прямо с первой строки. Вы запускаете `gnuplot` и вводите в него две команды:

```
set key autotitle columnhead
plot 'data.csv' using 1:2
```

Какое в этом случае будет название у построенного ряда данных и сколько будет нарисовано точек на графике?

Выберите один вариант из списка

Верно решили 17 975 учащихся
Из всех попыток 32% верных

☒ Всё получилось!

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☐ Название – первое значение из первого столбца, нарисовано 10 точек
- ☒ Название – первое значение из второго столбца, нарисовано 9 точек (точка из первой строки пропущена)
- ☐ Название – первое значение из первого столбца, нарисовано 9 точек (точка из первой строки пропущена)
- ☐ Название "data.csv" using 1:2, нарисовано 10 точек
- ☐ Название "no name", нарисовано 10 точек

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.60: Задание №29

Задание №30: на скрине всё видно (рис. 3.61)

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 ОХ (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!

Напишите текст

✓ Прекрасный ответ.

Верно решили **13 935** учащихся
Из всех попыток **44%** верных

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

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **2 балла** из 2

Рис. 3.61: Задание №30

Задание №31: создала файлы `animated.gnu` и `move.rot`. Написала нужные программы и далее проверила с помощью команды `gnuplot -persist animated.gnu` (рис. 3.62), (рис. 3.63), (рис. 3.64), (рис. 3.65), (рис. 3.66), (рис. 3.67), (рис. 3.68)

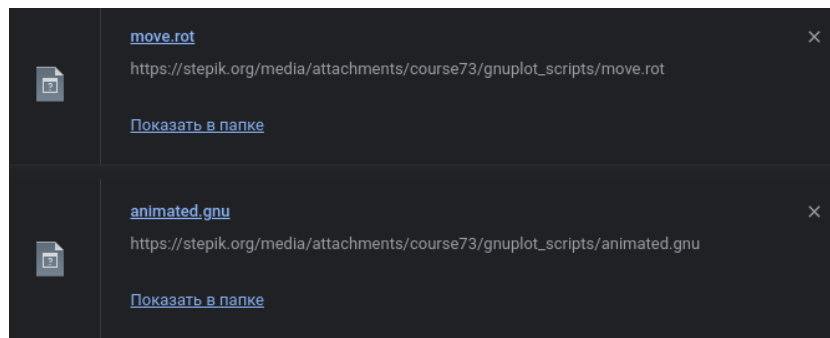
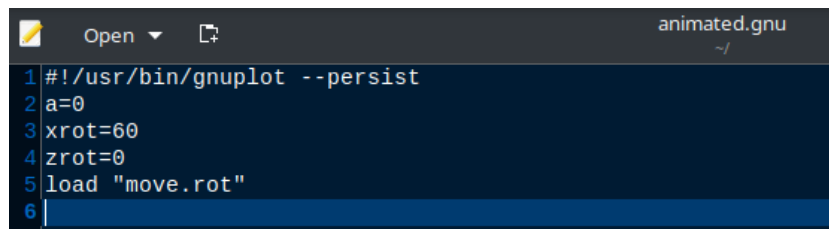


Рис. 3.62: Скачивание нужных файлов

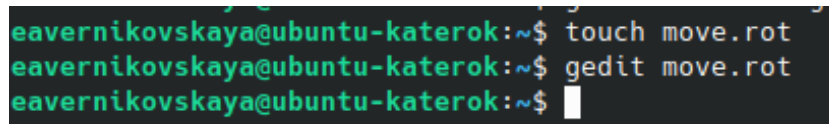
```
eavernikovskaya@ubuntu-katerok:~$ touch animated.gnu
eavernikovskaya@ubuntu-katerok:~$ gedit animated.gnu
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.63: Создание файла `animated.gnu`



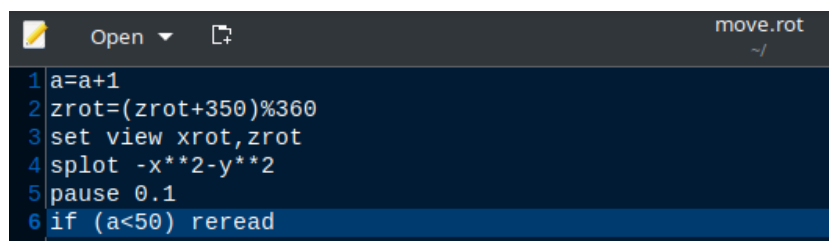
```
1#!/usr/bin/gnuplot --persist
2a=0
3xrot=60
4zrot=0
5load "move.rot"
6
```

Рис. 3.64: Программа для animated.gnu



```
eavernikovskaya@ubuntu-katerok:~$ touch move.rot
eavernikovskaya@ubuntu-katerok:~$ gedit move.rot
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.65: Создание файла move.rot



```
1a=a+1
2zrot=(zrot+350)%360
3set view xrot,zrot
4splot -x**2-y**2
5pause 0.1
6if (a<50) reread
```

Рис. 3.66: Программа для move.rot

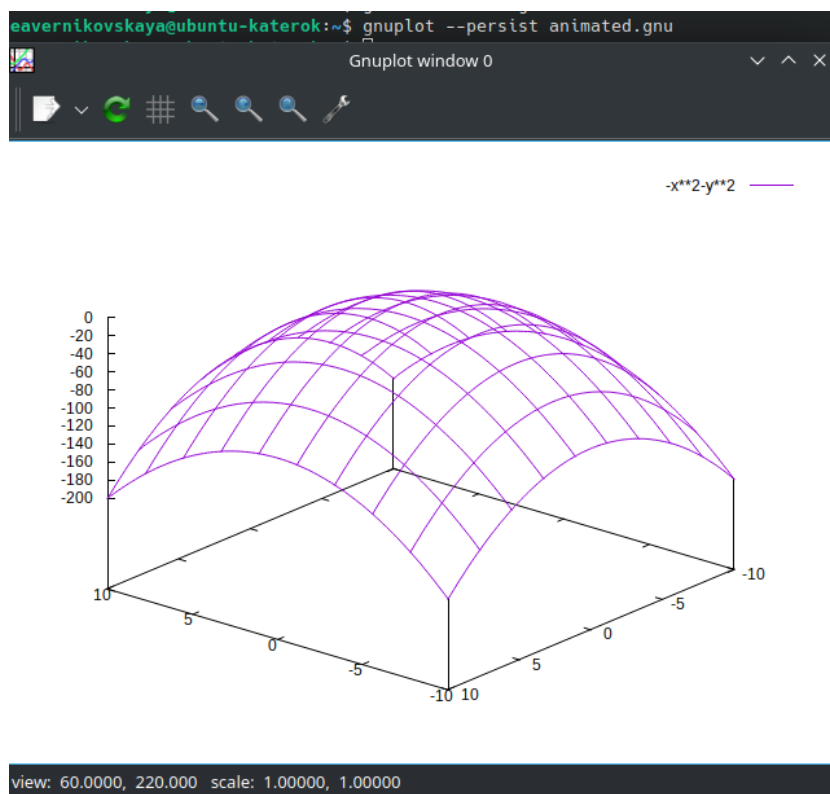


Рис. 3.67: Работа программы для задания №31

Если вы не скачали на предыдущем шаге файлы [animated.gnu](#) и [move.rot](#), то скачайте их теперь, т.к. они понадобятся для выполнения задания.

Указанные файлы использовались в последнем видеофрагменте для создания вращающегося графика. Измените инструкции в файле `move.rot` (т.е. **добавлять** и **удалять** инструкции **нельзя**) таким образом, чтобы:

- График **отразился зеркально** относительно горизонтальной поверхности. То есть там, где была точка (10, 10, 200), станет точка (10, 10, -200), где была точка (-10, -10, 200) станет (-10, -10, -200) и т.д. При этом точка (0, 0, 0) останется на месте.
- Изображение стало **вращаться в обратную сторону**. То есть если раньше вращалось "влево", то теперь станет "вправо".
- Вращение стало **в два раза быстрее**. То есть станет в два раза больше перерисовок графика на каждую секунду вращения.

Измененный файл загрузите в форму ниже.

Примечание: наша система проверки **не может** запустить на вашем файле `move.rot` программу `gnuplot` и сравнить полученный график с заданным. Вместо этого **мы анализируем команды**, которые вы указали в файле. Поэтому если вы видите, что ваш скрипт в `gnuplot` работает точно по условию, а мы отвечаем "Incorrect/Неверно", то попробуйте упростить свою модификацию `move.rot` и отправить его еще раз.

Напишите текст

✓ Хорошая работа.

Верно решили 12 854 учащихся
Из всех попыток 47% верных

```
a=a+1
zrot=(zrot+350)%360
set view xrot,zrot
splot -x**2-y**2
pause 0.1
if (a<50) reread
```

Следующий шаг

Решить снова

Ваши решения Вы получили: 3 балла из 3

Рис. 3.68: Задание №31

Программы для задания №31:

1) animated.gnu

```
#!/usr/bin/gnuplot --persist
a=0
xrot=60
zrot=0
load "move.rot"
```

2) move.rot

```
a=a+1
zrot=(zrot+350)%360
set view xrot,zrot
splot -x**2-y**2
pause 0.1
if (a<50) reread
```

Задание №32: на скрине всё видно (рис. 3.69)

Какая команда(ы) установят файлу `file.txt` права доступа `rwrxrw-r--`, если изначально у него были права `r--r--r--`. Укажите **все верные** варианты ответа!

Примечание: запись вида `команда1; команда2; команда3` означает, что в терминале последовательно выполнялись все три команды (сначала `команда1`, затем `команда2` и, наконец, `команда3`).

Выберите все подходящие ответы из списка

Верно решили **16 484** учащихся
Из всех попыток **21%** верных

☒ Абсолютно точно.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☒ `chmod u+wx file.txt; chmod g+w file.txt`
- ☐ `chmod o-wx file.txt; chmod g-x file.txt; chmod a+wx file.txt`
- ☐ `chmod u-wx file.txt; chmod g-w file.txt`
- ☐ `chmod rwxrw-r-- file.txt`
- ☐ `chmod 467 file.txt`
- ☒ `chmod 764 file.txt`

[Следующий шаг](#) [Решить снова](#)

[Ваши решения](#) Вы получили: **1 балл** из 1

Рис. 3.69: Задание №32

Задание №33: на скрине всё видно (рис. 3.70)

Предположим вы использовали команду `sudo` для создания директории `dir`. По умолчанию для `dir` были выставлены права доступа `rw-r--r--` (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` позволят даже редактировать этот файл, но сделают 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!**

Выберите все подходящие ответы из списка

Верно решили **14 683** учащихся
Из всех попыток **15%** верных

✓ Правильно.

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить свое решение с другими на [форуме решений](#).

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

Следующий шаг Решить снова

Рис. 3.70: Задание №33

Задание №34:

- `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. 3.71)

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

Выберите все подходящие ответы из списка

☒ Так точно!

Верно решили 17 158 учащихся
Из всех попыток 21% верных

Вы решили сложную задачу, поздравляем! Вы можете помочь остальным учащимся в [комментариях](#), отвечая на их вопросы, или сравнить своё решение с другими на [форуме решений](#).

- ☒ Размер файла в байтах
- ☒ Количество слов
- ☒ Количество символов
- ☒ Количество строк
- ☒ Длину самой длинной строки

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 1 балл из 1

Рис. 3.71: Задание №34

Задание №35: на скрине всё видно (рис. 3.72), (рис. 3.73)

```
eavernikovskaya@ubuntu-katerok:~$ du -h -s
88G
eavernikovskaya@ubuntu-katerok:~$
```

Рис. 3.72: Команда `du -h -s`

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 2948 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 880 Кбайт и две поддиректории в каждой из которой лежит по файлу в 400 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).

Напишите текст

☒ Хорошие новости, верно!

Верно решил 16 381 учащихся
Из всех попыток 53% верных

`du -h -s`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: 2 балла из 2

Рис. 3.73: Задание №35

Задание №36: на скрине всё видно (рис. 3.74)

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!

Напишите текст

✓ Так точно!

Верно решили **16 720** учащихся
Из всех попыток **40%** верных

`mkdir dir{1..3}`

Следующий шаг

Решить снова

[Ваши решения](#) Вы получили: **2 балла** из 2

Рис. 3.74: Задание №36

4 Выводы

В ходе выполнения 3 этапа внешних курсов на stepik я освоила linux, научилась пользоваться редактором vim, научилась писать скрипты на bash, а также строить графики в gnuplot.

5 Список литературы

1. Курс на stepik. Продвинутые темы [Электронный ресурс] URL: <https://stepik.org/course/73>.