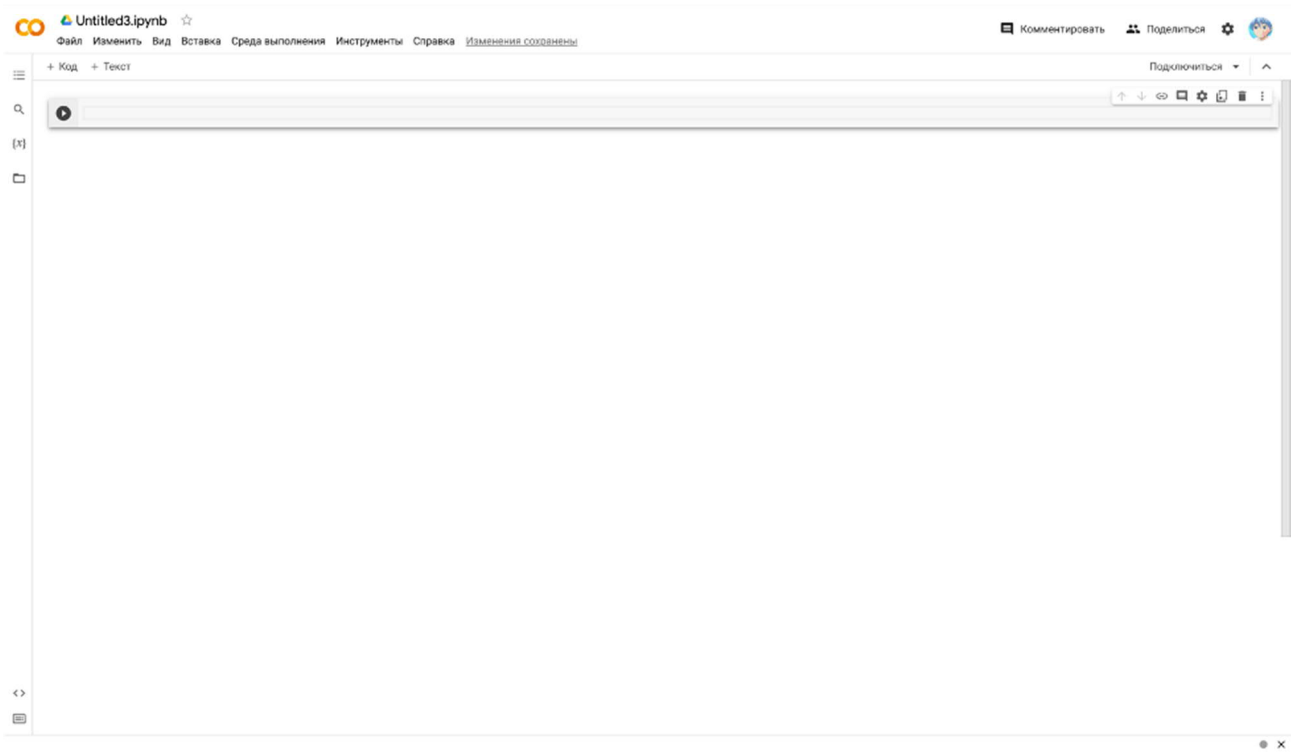
Лабораторная работа №3

Основы работы с языком Python

Создание переменных

Для работы с Python можно выбрать любой редактор кода. Мы же предлагаем поработать в Google Colab. Для работы в нем необходимо только подключение к Интернету и Google **Аккаунт**. Интерфейс среды достаточно прост.

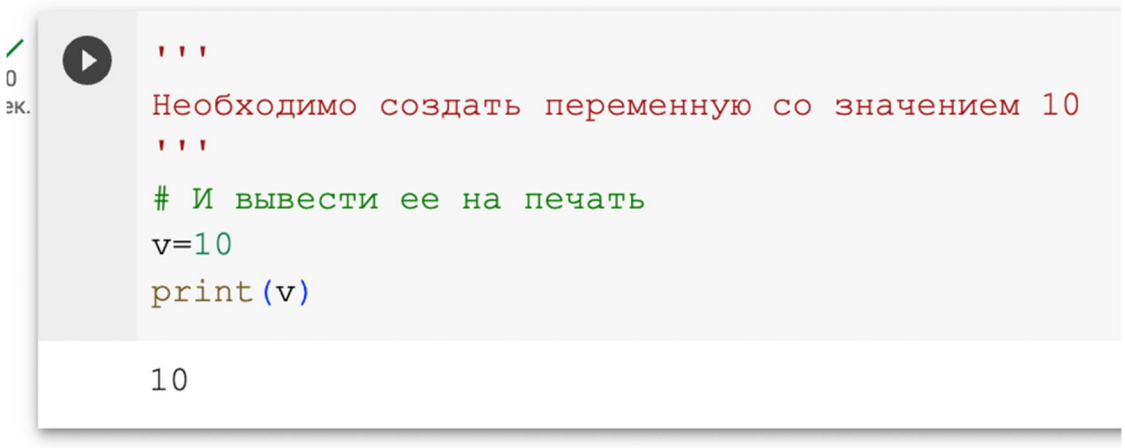


Сверху строка меню, в разделе *Файл* можно содавать новые блокноты, сохранять, переименовывать текущий и тд. Важно! Сохранение блокнота происходит на Гугл-диск, поэтому убедитесь, что у вас есть на нем свободное место.

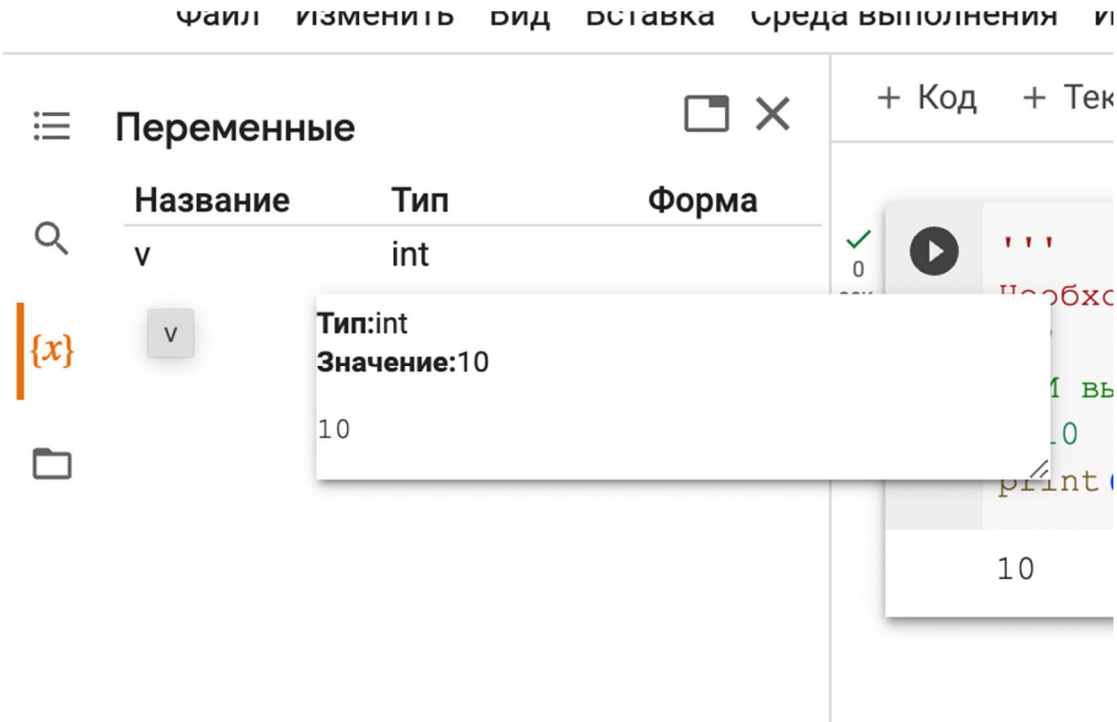
По умолчанию в блокноте уже создано поле для написания кода, можно добавлять дополнительные как для кода, так и для текста.

Запустить написанный код можно по специальной кнопке в левом верхнем углу блока. Результат выполнения будет выведен ниже.

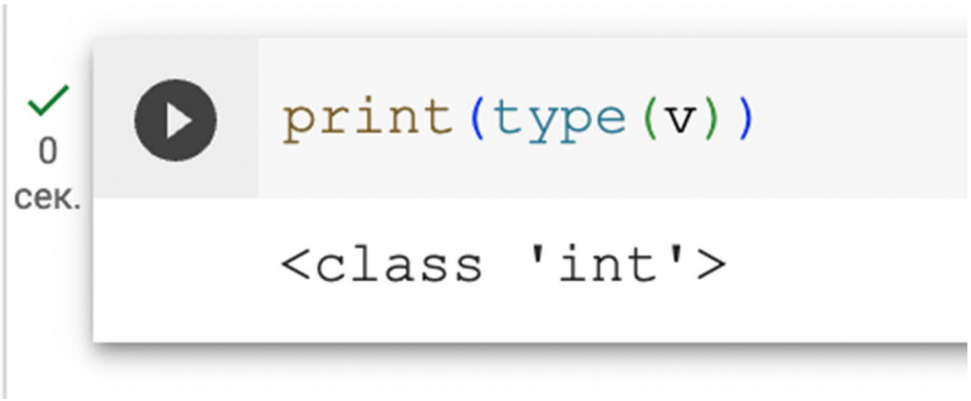
Текст, взятый в """ воспринимается как комментарии и на работе программы не отражаются. Аналогично любую строчку можно исключить из выполнения программы знаком «#». Для отражение объекта в консоли необходимо использовать функцию print().



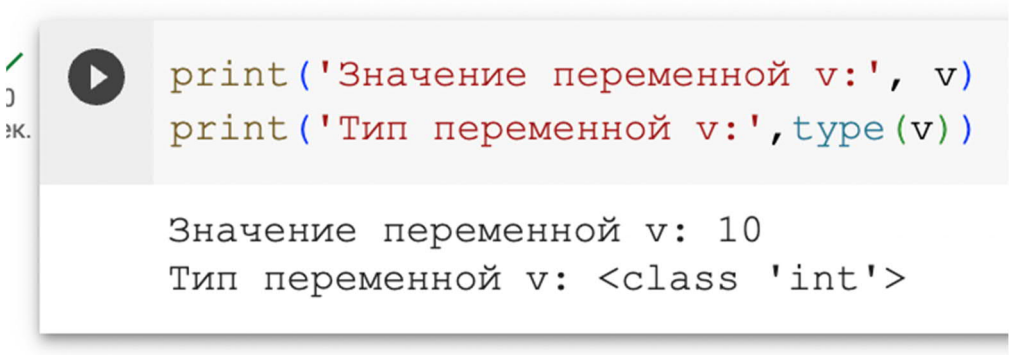
Если перейти в меню переменных, можно увидеть, что была создана переменная v со значением 10. Переменной был присвоен тип integer (int) – целое число.



Также тип объекта можно определить с помощью функции type().

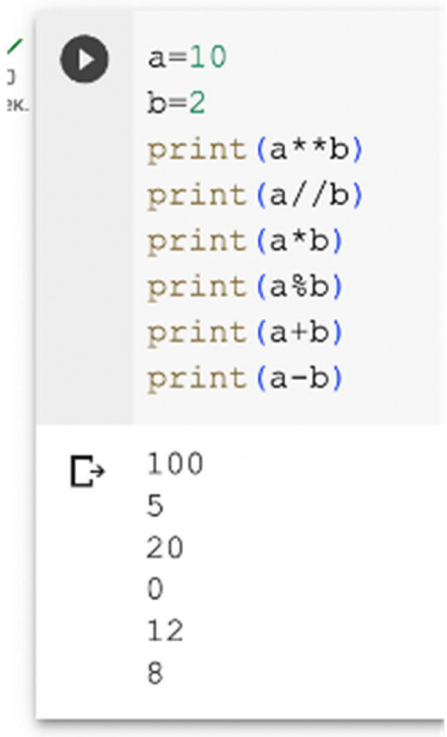


Для более удобного вывода результата работы можно добавлять комментарии внутри функции print().



Для манипулирования данными используются следующие операторы:

- Возведение в степень **
- Деление /
- Целочисленное деление //
- Умножение *
- Остаток от деления %
- Сложение +
- Вычитание –



```
a=10
b=2
print(a**b)
print(a//b)
print(a*b)
print(a%b)
print(a+b)
print(a-b)
```

```
100
5
20
0
12
8
```

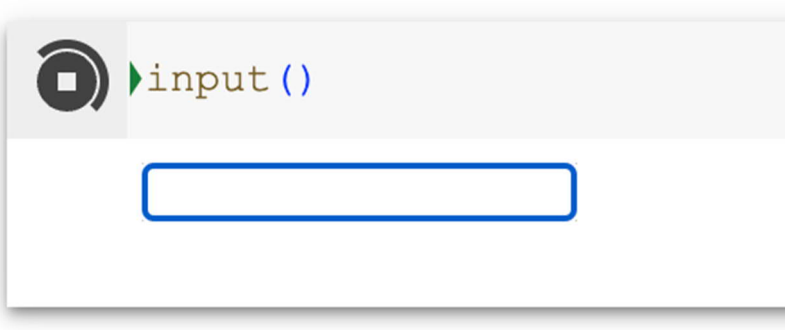
Полный функционал языка программирования можно посмотреть в документации.

Задание для самостоятельной тренировки

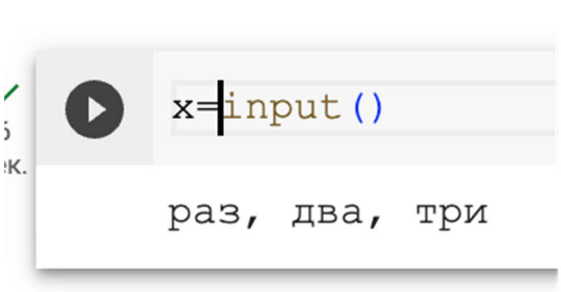
1. Создать объект a со значением 10
2. Создать объект b со значением 30
3. Изменить объект a на 8
4. Объект c определить как a^b
5. Вывести на экран объекты a, b, c
6. Определить тип объекта
7. Изменить тип объекта a на float
8. Объект b изменить на -6,8
9. Вывести на экран объекты a, b, c
10. Определить модуль b (функция abs())
11. Найти целую часть от деления b на a
12. Найти остаток от деления b на c и округлить до целых
13. Определить максимальное значение из a, b , c
14. Определить минимальное значение из a, b , c
15. Перевести переменные в восьмеричное представление

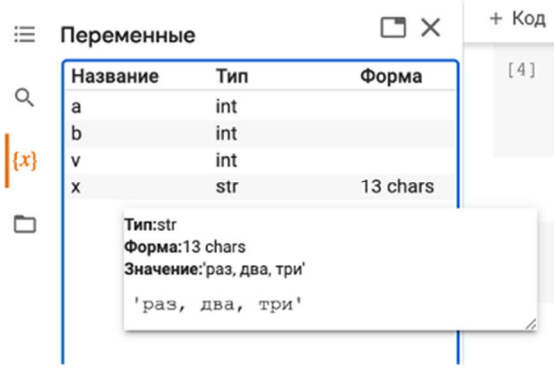
Ввод и вывод данных

За ввод в программу данных с клавиатуры в Python отвечает функция input(). Когда вызывается эта функция, программа останавливает свое выполнение и ждет, когда пользователь введет текст.

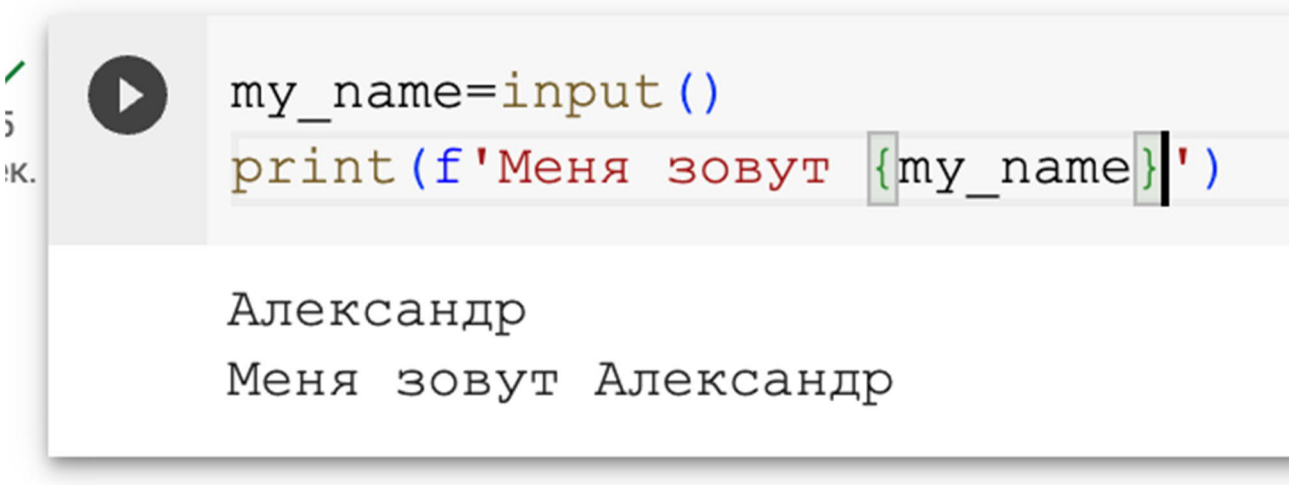


После этого, когда он нажмет Enter, функция input() заберет введенный текст и передаст его программе, которая уже будет обрабатывать его согласно своим алгоритмам. С помощью функции input() можно создавать переменные.

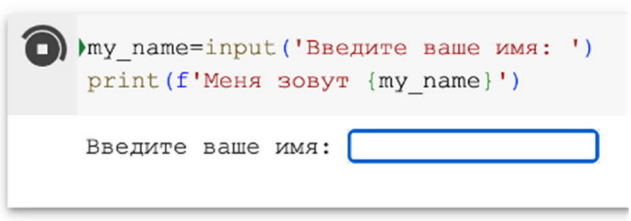


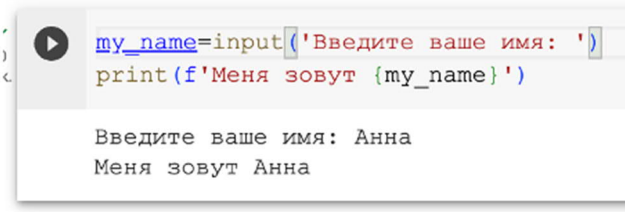


Введенные переменные можно использовать сразу при выводе результатов. Например:

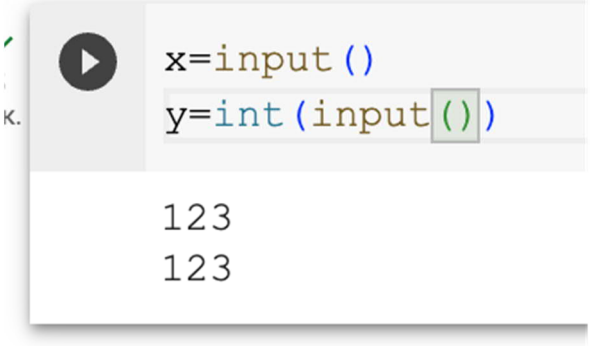


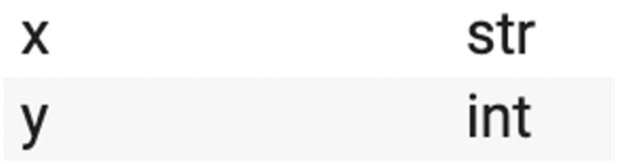
Внутри функции input() можно прописывать сообщения, которые будут выводиться в консоли:



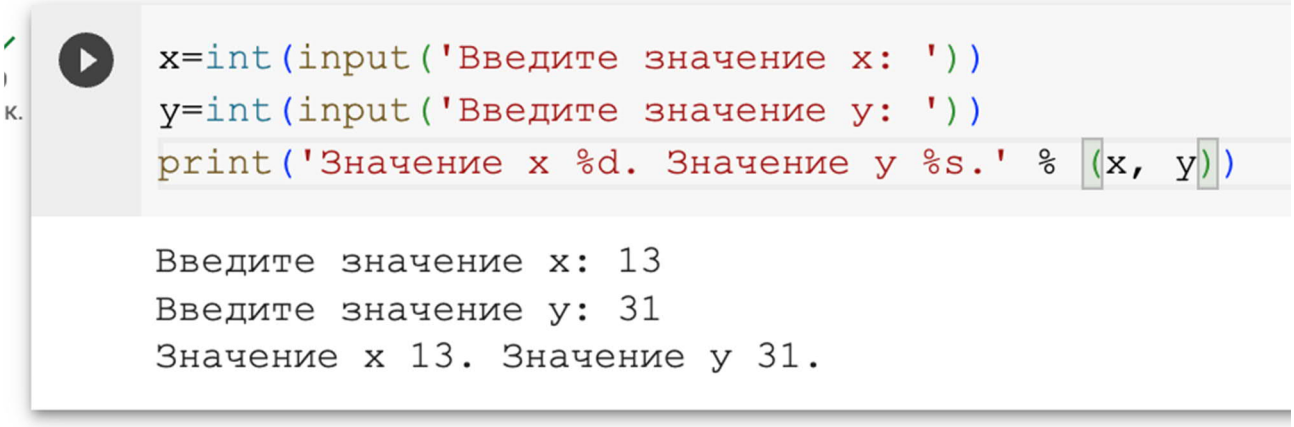


Тип переменных по умолчанию создается string. Чтобы создавать цифровые значения необходимо сразу передавать нужный формат:





Для вывода на экран числовых переменных можно использовать следующие конструкции:



На месте % буду выведены значения переменных, перечисленных в скобках после «%».

%d — это заполнитель для числовых или десятичных значений.

%s — заполнитель для строк.

Для вывода значения в формате float можем округлить до нужного количество знаков:

```
x=int(input('Введите значение x: '))
y=int(input('Введите значение y: '))
print('Значение x %d. Значение y %.2f.' % (x, y))
```

```
Введите значение x: 13
Введите значение y: 31
Значение x 13. Значение y 31.00.
```

Аналогичный результат можно получить, используя конструкцию:

```
x=int(input('Введите значение x: '))
y=int(input('Введите значение y: '))
print('Значение x {0:}. Значение y {1:.2f}.' .format(x, y))
```

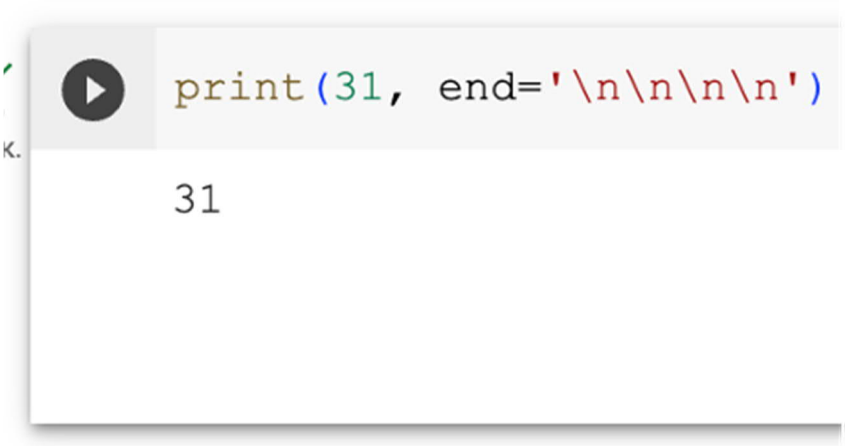
```
Введите значение x: 13
Введите значение y: 31
Значение x 13. Значение y 31.00.
```

В print() предусмотрены дополнительные параметры. Например, через параметр sep можно указать отличный от пробела разделителsь строк:

```
print(1, 2, 3, sep='-')
```

```
1-2-3
```

Параметр end позволяет указывать, что делать, после вывода строки. По-умолчанию происходит переход на новую строку. Однако это действие можно отменить, указав любой другой символ или строку:



Символ \n означает переход на новую строку.

Задание для самостоятельной тренировки

1. Напишите программу, которая запрашивала бы у пользователя:

- Имя
- Фамилия
- Возраст
- Место жительства

После этого выводила бы три строки, заменяя выделенные слова введенными:

- Меня зовут Имя Фамилия.
- Мне Возраст лет.
- Я живу в Место жительства.

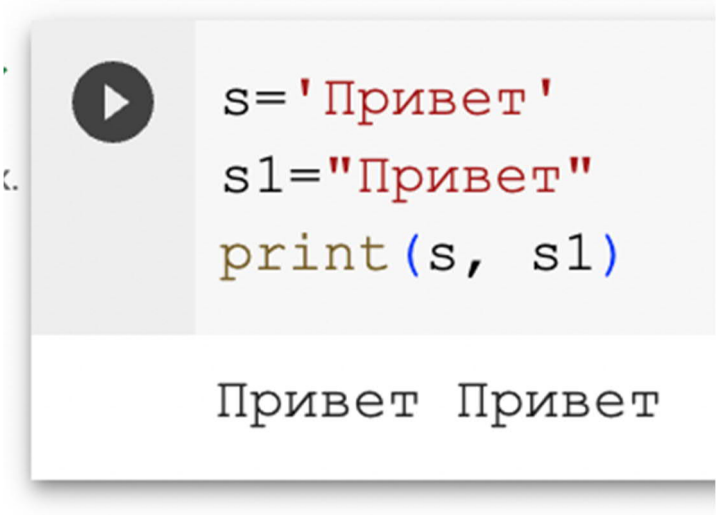
2. Напишите программу, которая предлагала бы пользователю решить пример 5 * 100 - 24. Потом выводила бы на экран правильный ответ и ответ пользователя. Подумайте, нужно ли здесь преобразовывать строку в число.

3. Запросите у пользователя три числа. Отдельно сложите первые два и отдельно второе и третье. Разделите первую сумму на вторую. Выведите результат на экран так, чтобы ответ содержал две цифры после запятой.

Строки

Строки в Python - упорядоченные неизменяемые последовательности символов, используемые для хранения и представления текстовой информации, поэтому с помощью строк можно работать со всем, что может быть представлено в текстовой форме.

Строки можно создать несколькими способами:

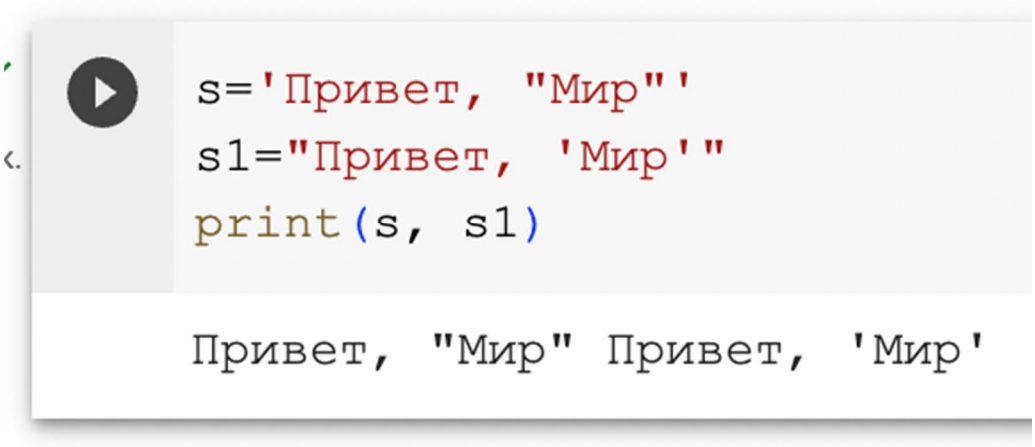


1. С помощью одинарных и двойных кавычек.

В списке переменных должны появиться 2 новые одинаковые переменные:

s	str	6 chars	'Привет'
s1	str	6 chars	'Привет'

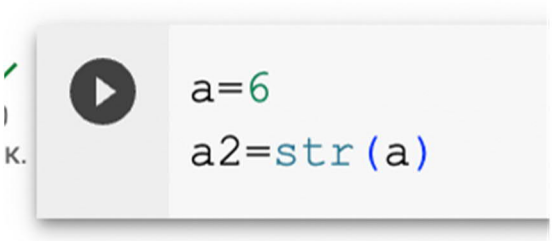
Строки в одинарных и двойных кавычках - одно и то же. Причина наличия двух вариантов в том, чтобы позволить вставлять в строки символы кавычек, не используя экранирование. Например, вот так (обратите внимание на кавычки внутри строки):



2. С помощью тройных кавычек.

Тройные кавычки можно использовать для записи многострочных блоков текста. Внутри такой строки возможно присутствие кавычек и апострофов, главное, чтобы не было трех кавычек подряд. Пример:

3. С помощью специальной функции str().



Функция str() преобразует переменную любого типа в строковую. В списке переменных у объекта a тип int (целое число), а у объекта s2 тип – str (строка).

Экранирование строк

Для преобразования переменных используются специальные символы для того, чтобы при выводе на печать текст приобретал нужный формат:

```
s='Специалист по обработке больших данных'
s1='Специалист \n по обработке больших данных'
print(s)
print(s1)
```

```
Специалист по обработке больших данных
Специалист
 по обработке больших данных
```

\n	Перевод строки
\t	Горизонтальная табуляция
\r	Возврат каретки

Возврат каретки убирает весь текст перед знаком \r

```
s1='Специалист \r по обработке больших данных'
print(s1)
```

```
 по обработке больших данных
```

Горизонтальная табуляция добавляет отступ в начале строки:

```
s1='\t Специалист по обработке больших данных'
print(s1)
```

```
        Специалист по обработке больших данных
```

Получение среза

Срез (slice) — извлечение из данной строки одного символа или некоторого фрагмента подстроки или подпоследовательности.

Индекс – номер символа в строке (а также в других структурах данных: списках, кортежах). Обратите внимание, что нумерация начинается с 0. Если указать отрицательное значение индекса, то номер будет отсчитываться с конца, начиная с номера -1.

0	1	2	3	4	5
П	р	и	м	е	р
-6	-5	-4	-3	-2	-1

Для обращения к элементам строк используются квадратные скобки, в которых указывается индекс нужного элемента. Так, для выбора первого элемента строки «Пример» необходимо указать индекс «0».

```
f='пример'
print(f[0])
```

```
п
```

```
f='пример'
print(f[-1])
```

```
р
```

Для получения среза необходимо указать диапазон по индексам. Так, для выбора первых 3 элементов необходимо указать диапазон от 0 до 3. Элемент с индексом «3» не будет включен в выборку.Также при получении срезу можно указать шаг для выбора необходимых символов. А для вывода символов в обратном порядке можно указать шаг «-1».

```
f='пример'
print(f[0:3])
```

```
при
```

```
f='пример'
print(f[::2])
```

```
пие
```

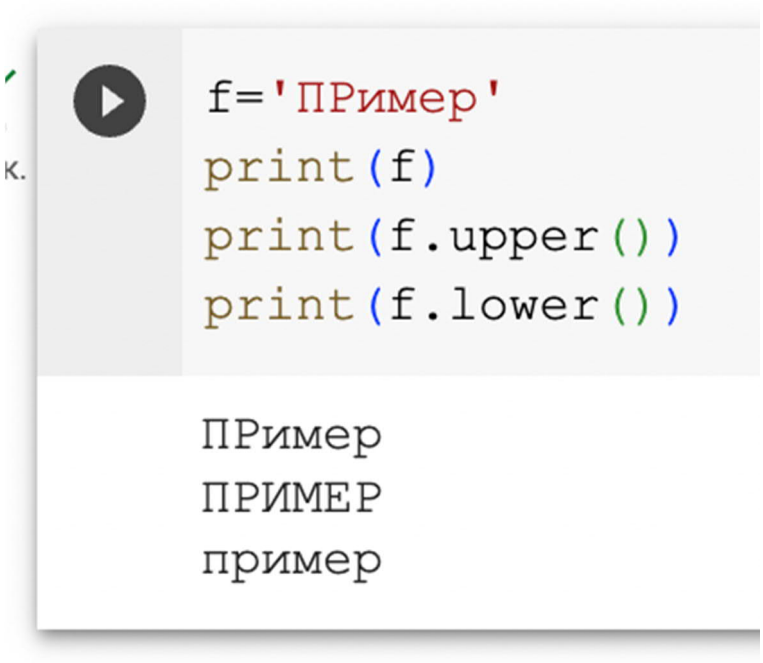
```
f='пример'
print(f[::-1])
```

```
ремирп
```

Методы преобразования строк

Для удаления лишних пробелов в начале строки используется функция lstrip. Аналогично работают функции rstrip (удаление пробелов в конце) и strip (удаление и в начале, и в конце).

Функция upper делает все символы строки прописными, lower – строчными. Функция title делает первую букву каждого слова прописной, а capitalize – только первую букву первого слова.



Задание для самостоятельной тренировки

Создать объект D: « My name is »

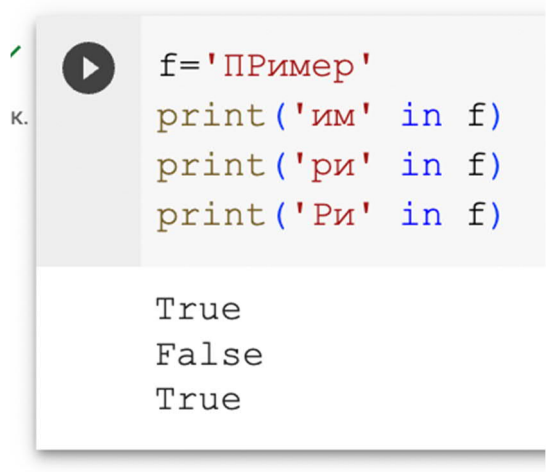
1. Убрать пробелы лишние
2. Вывести на печать «My name is ФИО» (ФИО – ваше собственное)
3. Вывести на печать все слова п.2 через знак табуляции
4. Вывести на печать все слова п.2 после слова «is» через символ возврата каретки
5. Повторить фразу 8 раз
6. Вывести все символы п.2 с заглавных букв
7. Вывести все символы п.2 прописными
8. Вывести все символы п.2 строчными

9. Вывести из п.2 символ «у» через обращение по индексам
10. Вывести только Ваше ФИО через получение среза
11. Вывести каждый второй символ п.2

Операции со строками

Проверка на вхождение

Для проверки вхождения элемента в строку используется «in». Если элемент присутствует в последовательности, то возвращает True, иначе – False. Аналогично работает конструкция «not in», на наоборот. Если элемент отсутствует в последовательности, то возвращает True, иначе – False.

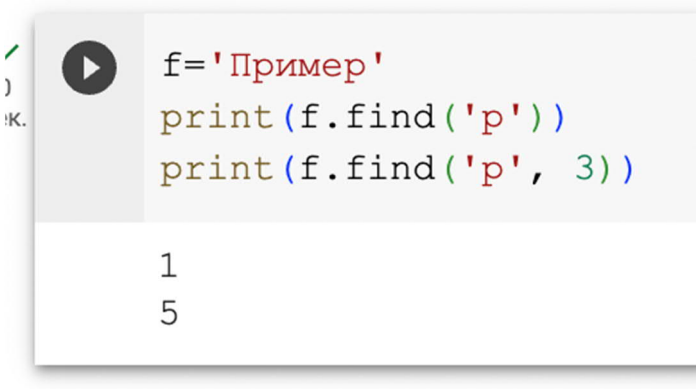


Для определения количества символов в строке используется функция len.

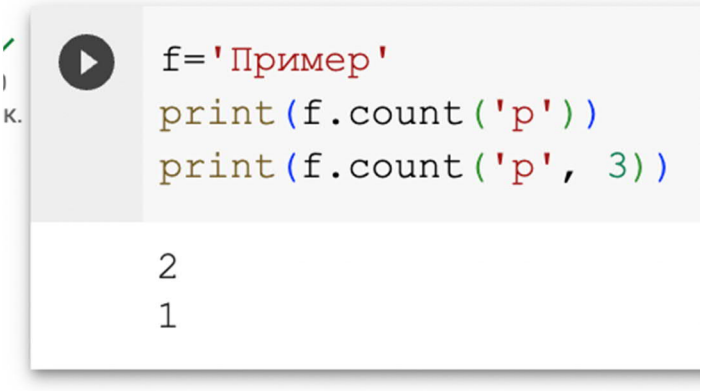
index Возвращает индекс элемента, имеющего указанное значение. Дополнительно мы можем указать часть списка, в котором выполняется поиск.

В первом случае будет осуществлен поиск с начала строки и будет определен индекс первого вхождения «р» в строку. Во втором случае мы начинаем поиск «р» с элемента с индексом 3. Тогда определяется, что символ «р» находится на позиции с индексом 5.

Аналогично работает функция find.



Функция count определяет количество элементов с указанным значением. Аналогично мы можем указать с какого элемента нужно начать поиск. Так, начиная с элемента с индексом 3 «р» встречается всего 1 раз.



Функция replace заменяет задаваемый фрагмент на новый.На первом месте указываем какой фрагмент будем заменять, на второй позиции указываем на что будем его заменять, а на 3 позиции указывается количество замен.

```
f='Пример'
print(f.replace('р', '---'))
print(f.replace('р', '---', 1))
```

```
П---име---
П---имер
```

Для строк используется ряд функций для проверки содержимого строки:

isdigit() – строка содержит только цифры;

isalpha() – строка содержит только буквы;

isalnum() – строка содержит только буквы и (или) только цифры

islower() – строка содержит только прописные буквы

isupper() – строка содержит только заглавные буквы

istitle() – первые буквы всех слов являются заглавными

split разбивает строку на части, используя специальный разделитель x, и возвращает эти части в виде списка.

rsplit работает аналогично, но поиск осуществляется справа налево.

Функция join возвращает строку, собранную из элементов указанного объекта, поддерживающего итерирование (например, список строк).

```
l= ['п', 'р', 'и', 'м', 'е', 'р']
print('-'.join(l))
print(''.join(l))
```

```
п-р-и-м-е-р
пример
```

Посмотреть полный список с операциями и методами для строк можно в документации.

Задание для самостоятельной тренировки

1. Создайте строку с Вашей ФИО
- Определите количество символов в строке.
- Сколько раз встречается буква «а»? «б»? «в»?
- На позиции с каким индексом находится символы «а», «б», «в»?
- Определите сколько раз встречаются символы «а», «б», «в» если поиск осуществлять с элемента с индексом 5
- На позиции с каким индексом находится символы «а», «б», «в» если поиск осуществлять с элемента с индексом 5
- Проверить строку:

- строка содержит только цифры;

- строка содержит только буквы;

- строка содержит только буквы и (или) только цифры;

- строка содержит только прописные буквы;

- строка содержит только заглавные буквы;

- первые буквы всех слов являются заглавными

2. Создать объект E: a1 a2 a3

- Разделить строку на подстроки и добавить их в список
- Соединить разъединенные объекты из п.1 через «,»
- Разбить объект п.3 на подстроки через знак «,» на 2, при этом поиск знака осуществлялся справа налево

Списки

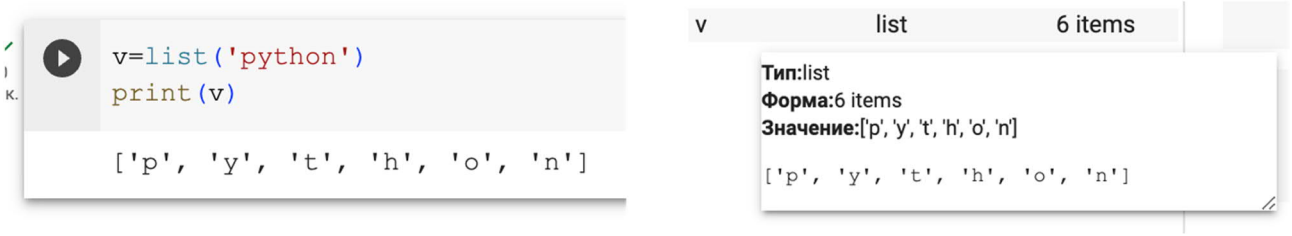
В языке программирования Python существует четыре типа данных для хранения последовательностей:

- *List (**список**)* — упорядоченная последовательность, которую можно изменять. Допускаются одинаковые элементы.
- Tuple (**кортеж**) — последовательность, которая упорядочена, но не изменяемая. Допускаются одинаковые элементы.
- Set (**множество**) — неупорядоченная изменяемая последовательность. Одинаковые элементы удаляются.
- Dict (**словарь**) — неупорядоченная изменяемая последовательность, состоящая из пар ключ, значение. Ключи не дублируются.

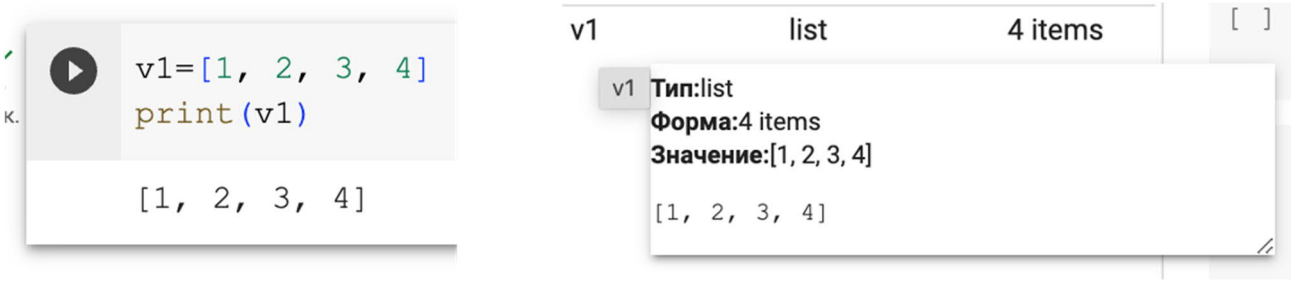
Списки

Списки в Python - упорядоченные изменяемые коллекции объектов произвольных типов (почти как массив, но типы могут отличаться).

Чтобы использовать списки, их нужно создать. Создать список можно несколькими способами. Например, можно обработать любой итерируемый объект (например, строку) встроенной функцией list. В результате будет выведен список букв и сохранен объект с типом list в списке переменных.



Также в Python создать списки можно с помощью квадратных скобок. В результате будет выведен список, а также сохранен объект:



Обратиться к элементам списков можно через квадратные скобки (как и к элементам строк).

Посмотреть полный список операций со списками можно в документации.

Задание для самостоятельной тренировки

Создайте список v: 2,3,4,2,3, 2,3,4,1,1,1,5,6,6,6.

1. Добавьте в конец вектора элементы 5,7,8
2. Отсортировать вектор по возрастанию.
3. Выбрать элементы с 5 по 8 включительно
4. Выбрать элементы с 3 по 10 не включительно
5. Выбрать 3, 5 и 12 элементы
6. Удалить элемент, стоящий на 4 месте в отсортированной списке.
7. На вторую позицию вставить значение 6.
8. Проверить есть ли значение 9 и 1 в в екторе.
9. Создать вектор x как копию вектора v
10. Выстроить элементы вектора x в обратном порядке
11. Удалить из x последний элемент

12. Добавить на 5 позицию значение 12 в вектор x
13. Удалить 5 и 3 элементы из x
14. Объединить векторы x и y и вывести их 3 раза на печать
15. Сколько раз в векторе x встречается значение 6
16. Преобразовать вектор x в строку, предусмотрев следующие разделители: «,», «_», «/».
17. Найти корень из каждого значения вектора x
18. Найти максимальное и минимальное значение вектора x
19. Создать вектор y от 3 до 15
20. Создать вектор z от 20 до 3