Строки

Строка – это последовательность символов. Чаще всего строки – это просто некоторые наборы слов.

Одинарные ковычки:

```
string = 'hello world'
print(string)
```

```
C:\Users\admin\Py
hello world
```

Двойные ковычки:

```
string = "hello world"
print(string)
```

```
C:\Users\admin\Pych
hello world
```

Как вы заметили строки в двойных ковычках работают так же, как и в одинарных.

Тройные ковычки:

Можно указывать «многострочные» строки с использованием тройных кавычек (""" или'''). В пределах тройных кавычек можно свободно

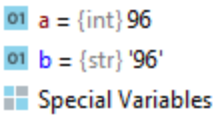
использовать одинарные и двойные кавычки. Для большого текста. Например:

```
'''Это многострочная строка. Это её первая строка.
Это её вторая строка.
"What's your name?", - спросил я.
Он ответил: "Bond, James Bond."
'''
```

Преобразование типов данных с помощью функции str()

Вы можете преобразовывать другие типы данных Python в строки с помощью функции str():

```
a = 96
b = str(a)
print(b)
```



Объединяем строки с помощью символа +:

Вы можете объединить строки или строковые переменные в Python с помощью
оператора +, как показано далее:

```
string1 = "hello"
string2 = "world"
print(string1 + string2)
```

```
C:\Users\admin\Py
helloworld
```

```
string1 = "hello"
string1 = string1 + "world"
print(string1)
```

```
C:\Users\admin\P
helloworld
```

Python недобавляет пробелы за вас при конкатенации строк, поэтому впредыдущем примере нужно явно добавить пробелы. Далее мы добавляем пробелы между каждым аргументом выражения print(), а также символ новой строки в конце:

```
a = 'Duck.'
b = a
c = 'Grey Duck!'
print(a + b + c)
```

```
C:\Users\admin\PycharmProj
Duck.Duck.Grey Duck!
```

Размножаем строки с помощью символа *:

Оператор * можно использовать для того, чтобы размножить строку. Попробуйте
ввести в интерактивный интерпретатор следующие строки и посмотреть, что получится:

```
name = "Petya"
print(name * 4)
```

```
C:\Users\admin\PycharmProjects
PetyaPetyaPetyaPetya
```

Извлекаем подстроки с помощью оператора [start : end : step](slice):

Из строки можно извлечь подстроку (часть строки) с помощью функции slice. Вы определяете slice с помощью квадратных скобок, смещения начала подстроки
start и конца подстроки end, а также опционального размера шага step. Некоторые из этих параметров могут быть исключены. В подстроку будут включены символы, расположенные начиная с точки, на которую указывает смещение start, и заканчивая точкой, на которую указывает смещение end.

1) Оператор [:] извлекает всю последовательность от начала до конца.

```
name = "Petya"
print(name[:])
```

```
C:\Users\admin\
Petya
```

2) Оператор [start :] извлекает последовательность с точки, на которую указывает смещение start, до конца

```
name = "Petya"
print(name[2:])
```

```
C:\Users\admin\Pycha
tya
```

3) Оператор [: end] извлекает последовательность от начала до точки, на которую
указывает смещение end минус 1

```
name = "Petya"
print(name[:3])
```

```
C:\Users\admin\Pycharm
Pet
```

4) Оператор [start : end] извлекает последовательность с точки, на которую указывает смещение start, до точки, на которую указывает смещение end минус 1.

```
name = "Petya"
print(name[1:4])
```

```
C:\Users\admin\PycharmPr
ety
```

5) Оператор [start : end : step] извлекает последовательность с точки, на которую
указывает смещение start, до точки, на которую указывает смещение end минус 1,
опуская символы, чье смещение внутри подстроки кратно step.

```
name = "Petya"
print(name[1:4:2])
```

```
C:\Users\admin\Pychar
ey
```

Смещение слева направо определяется как 0, 1 и т. д., а справа налево — как -1, -2 и т. д. Если вы не укажете смещение start, функция будет использовать в качестве его значения 0 (начало строки). Если вы не укажете смещение end, функция будет использовать конец строки.

Длина строки с помощью len()

Функция len() подсчитывает символы в строке:

```
name = "Petya"
print(len(name))
```

```
C:\Users\admin\PycharmPrc
5
```

Разделение строки с помощью split()

Вы можете использовать встроенную функцию split(), чтобы разбить строку на список небольших строк, основываясь на разделителе. Список — это последовательность значений, разделенных запятыми и окруженных квадратными скобками:

```
string = "hello my name is"
string_split = string.split(" ")
print(string_split)
```

```
C:\Users\admin\PycharmProjects\L
['hello', 'my', 'name', 'is']
```

Если вы не укажете разделитель, функция split() будет использовать любую последовательность пробелов, а также символы новой строки и табуляцию

Если вы вызываете функцию split без аргументов, вам все равно нужно добавлять
круглые скобки — именно так Python узнает, что вы вызываете функцию.

Объединение строки с помощью функции join()

Функция join() является противоположностью функции split(): она объединяет список строк в одну строку. Вызов функции выглядит немного запутанно, поскольку сначала вы указываете строку, которая объединяет остальные, а затем — список строк для объединения: string.join(list). Для того чтобы объединить список строк lines, разделив их символами новой строки, вам нужно написать '\n'.join(lines). В следующем примере мы объединим несколько имен в список, разделенный запятыми и пробелами:

```
string = "hello my name is"
string_split = string.split(" ")
string_join = " ".join(string_split)
print(string_join)
```

```
C:\Users\admin\PycharmProjects\Le
hello my name is
```

Если надо объединить по запятым, то ставим “,”

```
string = "hello my name is"
string_split = string.split(" ")
string_join = ",".join(string_split)
print(string_join)
```

```
C:\Users\admin\PycharmPro
hello,my,name,is
```

Замена элементов строки с помощью replace():

Вы можете использовать функцию replace() для того, чтобы заменить одну подстроку другой.

```
string = "hello my name is"
new_string = string.replace("hello", "hi")
print(new_string)
```

```
C:\Users\admin\Pychar
hi my name is
```

```
m replace(self, __old, __new, __count)
```

S.isdigit() - Состоит ли строка из цифр.
S.isalpha() - Состоит ли строка из букв.

```
print('A'.isdigit())

print('A'.isalpha())
```

```
False
True

Process finished with exit code 0
```

Строки.pdf

Задачи:

1) Напишите программу, которая запрашивает у пользователя его имя, а затем выводит строку «Привет, ...», где вместо многоточия имя пользователя. А вторая строка выведет имя пользователя с повтором 3 раза.

2) Вычислить сумму цифр случайного трёхзначного числа (тут необходимо применить работу со строками)

3) На вход подается непустая строка S. В строке хотя бы два символа.

1) В первой строке распечатайте каждый 3-й символ, начиная с нулевого (подряд, не разделяя символы пробелами).

2) Во второй строке распечатайте первый и последний символы (подряд, не разделяя символы пробелами).

3) В третей строке распечатайте S в верхнем регистре.

4) В четвертой строке распечатайте S в обратном порядке.

5) В пятой строке напечатайте True, если все символы в строке S — цифры и False в противном случае.

4) Разделить строку "***Apples, Oranges, Pears, Bananas, Berries***" по запятым и вывести на экран. Затем объединить элементы с использованием пробела, чтобы программа вывела "***Apples Oranges Pears Bananas Berries***".

5) Замените “#” на “/” в 'www.my_site.com#about '

6) Напиши программу, которая принимает строку от пользователя и выводит все слова этой строки в обратном порядке