Самое важное

Сравнение строк
Операции сравнения можно производить не только между числами, но и между строками. Наиболее популярными в этом случае являются операторы сравнения “==” и “!=”. Пока что мы познакомимся только с ними.

Синтаксис

<строка1> == <строка2>
<строка1> != <строка2>

Результатами таких сравнений, как обычно, могут быть True/False (или ошибка, если, например, программист пробует сравнить число и строку).

Пример

```
user_password = input(‘Введите пароль: ‘)
if user_password == ‘skillbox123’:
  print(‘Welcome!’)
else:
  print(‘Пароль введен с ошибкой, попробуйте ещё раз!’)
```

Итерирование по строке
Строка — это та же последовательность, только уже не чисел, а любых символов. Следовательно, возможно запустить цикл по строке:

```
for letter in ‘Hello, world!’:
  print(letter)
```

Как обычно, Python в начале каждого цикла будет записывать по одному элементу в переменную цикла. Распечатав её в цикле, мы увидим:

```
H
e
l
l
o
,

w
o
r
l
d
```

!

Важно: пробел для Python — это не пустое место, это такой же символ, как и все остальные!

Функция print и параметр end
Функция print уже знакома, но ранее мы не использовали её дополнительные параметры. Эти параметры важны, они помогают произвести более точную настройку и получить необходимый н вывод.

Параметр end:

- по умолчанию равен символу «начало новой строки»;
- записывается он так — ‘\n’.

Это значит, что когда мы пишем print(‘hello’), Python видит print(‘hello’, end=’\n’).
Можно изменить print(‘hello’, end=’ ‘) — вместо начала новой строки мы указали символ пробела. На первый взгляд в консоли ничего не изменилось, но если мы добавим ещё один принт print(‘world’), то в первом случае world появится на новой строчке, ниже нашей фразы hello, а во втором случае world появится на этой же строчке, вместе с hello.

В этот параметр можно передавать любой элемент строки, кроме того, строку можно передавать не явно, а через переменную!

Не допускай следующих ошибок!

Не забывайте, что строки и числа — это разные типы данных, даже ‘1’ и 1 — это разные типы данных для Python, хоть для нас они и выглядят похожими. ‘1’ + 1 Python выполнить не сможет.
Поэтому если мы вводим строку чисел через input:

```
numbers = input(‘введите последовательность чисел’)
summ = 0
for number in numbers:
  summ += int(number)
```

— не забывайте про функцию int, которая позволит привести тип str к типу int.

Кстати, есть аналог этой функции, который выполняет обратное действие:
str(1) -> ‘1’.