
Яндекс

Академия Яндекса

Типы данных. Операции над числами

Типы данных

Типы данных

У каждого элемента данных, который встречается в программе, есть свой тип.

```
hello == 'привет'
```

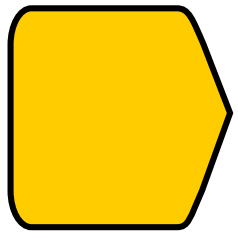
Строка

```
int_number == 42
```

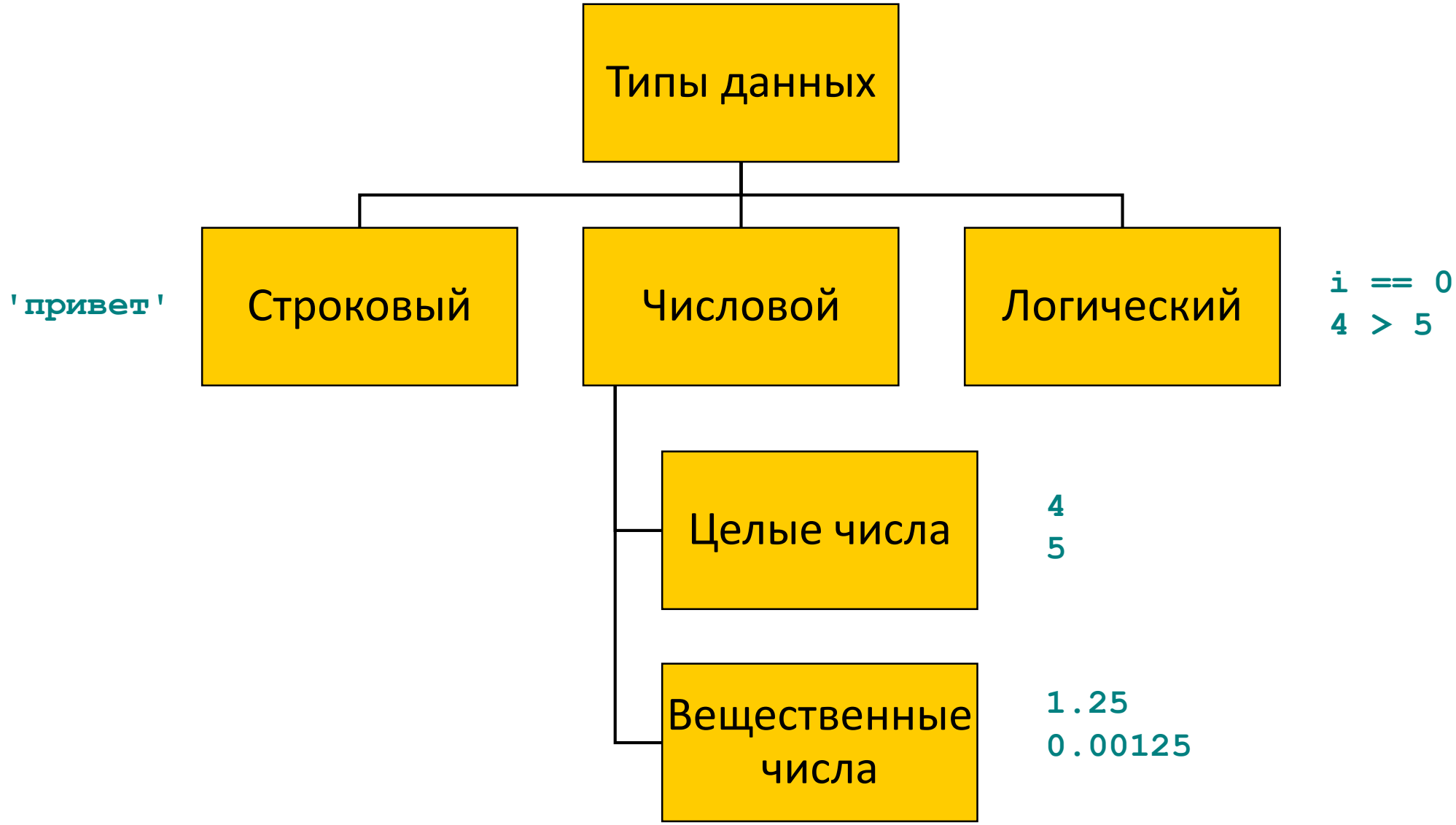
Целое число

```
float_number == 0.75
```

Вещественное число



Даже если данные не записаны прямо в программе, а получаются откуда-то ещё, у них есть совершенно определённый тип. Например, на место `input()` всегда подставляется строка, а `2 + 2` даст именно число `4`, а не строку `'4'`.

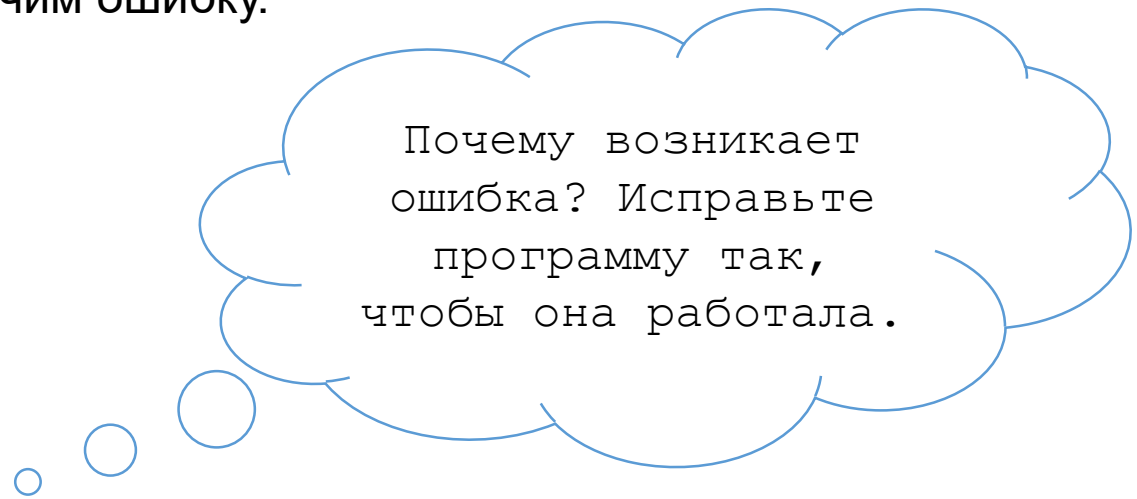

Типы данных
Строковый
Числовой
Логический
'привет'
i == 0
4 > 5
Целые числа
Вещественные числа
4
5
1.25
0.00125

Типы данных

Пользователь может ввести с клавиатуры какие-то цифры, но в результате **`input()`** вернёт строку, состоящую из этих цифр. Если мы попытаемся, например, прибавить к этой строке 1, то получим ошибку.

Давайте попробуем это сделать:

```
a = input()
print(a + 1)
```

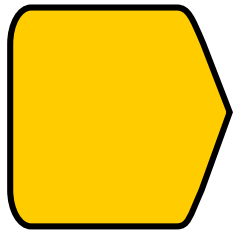


Сохраните и запустите программу.

Введите любое число и посмотрите, что получится.

Числовые типы данных

Числовые типы



Когда речь идет о числовых данных – они записываются **без кавычек**.

А для вещественных чисел – для разделения целой и дробной части используют **точку**.

Помните, мы складывали две строки:

```
print('10' + '20')
```

И получали результат – строку ‘1020’.

Давайте попробуем в этом примере убрать кавычки.

И результатом функции

```
print(10 + 20)
```

будет целое число 30.

А если мы попробуем сложить два вещественных числа

```
print(10.0 + 20.0)
```

, то результатом будет вещественное число 30.0.

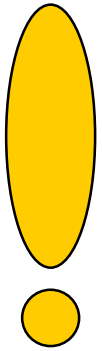
Что будет, если сложить вещественное число и целое число `print(10.0 + 20)`?

Операции над числами

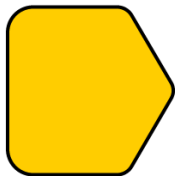
Сложение

Мы выполняли сложение двух чисел внутри функции **print**, но мы можем переменным давать нужные значение и выполнять действия над переменными.

Давайте напишем программу, которая задаст нужные значения двум переменным (10 и 20), потом вычислит их сумму, положит это значение в третью переменную и выведет на экран полученный результат. Допишите начальные строки, чтобы программа решала поставленную задачу:



```
...
print(summ)
```



Обратите внимание, что если в качестве имени переменной для суммы взять **sum**, то оно выделяется цветом. Это означает, что такое имя знакомо среде и принадлежит какой-то функции, в качестве имени переменной его лучше не использовать.

Другие операции над числами

Еще числа можно вычитать, умножать, делить, возводить в степень.

```
print(30 - 10)
print(30.0 - 10)
print(3 * 3)
```

Возведение в степень обозначается двумя звездочками **, которые должны записываться без разделителей.

```
print(9 ** 2)
```

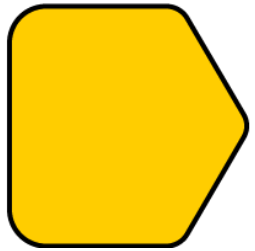
Обратите внимание, что результат деления – всегда вещественный, даже если мы делим два целых числа, которые делятся нацело.

```
print(10 / 2)
```

Попробуйте поделить на 0. Посмотрите, как будет выглядеть ошибка деления на 0.

Целочисленное деление

Целочисленное деление



Для реализации целочисленного деления существуют два действия – деление нацело и остаток от деления нацело. Получение целой части от деления обозначается как удвоенный знак деления `//`, а остатка от деления нацело – `%`.

Что будет выведено в результате этих действий?

```
print(10 // 3, 10 % 3)
print(10 // 5, 10 % 5)
print(10 // 11, 10 % 11)
```

Делим 10 на:	Целая часть	Остаток
3		
5		
11		

Пример

Допустим, вам известны результаты **`a // b`**, **`a % b`** и число **`b`**, напишите формулу, как найти число **`a`**?

Давайте проверим вашу формулу:

```
a = 10
b = 3
print(…А сюда напишем формулу…)
```

Обратите внимание на порядок выполнения действий в вашей формуле. Целочисленное деление имеет тот же приоритет, что и обычное деление, значит, будет выполняться раньше, чем вычитание и сложение. Для изменения приоритета выполнения операций используются скобки, все так же, как и в математике.

Пример

Попробуйте предположить, что выведется на экран после выполнения следующего куска кода:

```
print(10 // 3, 10 % 3)
print(-10 // 3, -10 % 3)
```

Пример

Попробуйте предположить, что выведется на экран после выполнения следующего куска кода:

```
print(10 // 3, 10 % 3)
print(-10 // 3, -10 % 3)
```

Знак остатка всегда равен знаку делителя и по модулю меньше его. При делении на положительное число – остаток всегда положительный, поэтому

-10 = -4 * 3 + 2

Пример

Определите, что будет выведено на экран?

```
a = 4
b = 15
c = b / 5 * 3 - a
print(c)
```

Деление на 1

Как вы думаете, что будет выведено?

```
print(5.2 // 1, 5.2 % 1)
```

Без `if`

Если в задаче нельзя пользоваться условным оператором,
можно применить целочисленное деление:

```
n = int(input())
print(n + n % 2)
```

Что будет выведено, если
n чётное? Нечётное?

Ввод чисел с клавиатуры

Для решения задач этого урока нам потребуется получать числа, вводимые пользователем. Но при вводе мы всегда получаем строку. Значит, эту строку нужно преобразовать в число. Подробнее с преобразованием типов данных мы познакомимся на следующем уроке, а пока получить целое или вещественное число из введенной строки можно так:

```
a = int(input()) # в переменную записали целое число
b = float(input()) # в переменную записали вещественное число
```


Яндекс