Переменные, Выражения и Операторы

Константы

- Неизменяемые значения, такие как: числа, буквы и строки называются «константами», потому что их значение не изменяется
- Числовые константы не обрамляются кавычками
- Для констант типа «Строка» используются одинарные (') или двойные кавычки (")

```
>>> print(123)
123
>>> print(98.6)
98.6
>>> print('Привет, мир!')
Привет, мир!
```

Ключевые слова

Не стоит использовать ключевые слова в качестве имен переменных или других идентификаторов

```
False   class   return  is      finally
None    if      for     lambda  continue
True    def     from    while   nonlocal
and     del     global  not     with
as      elif    try     or      yield
assert  else    import  pass
break   except  in      raise
```

Переменные

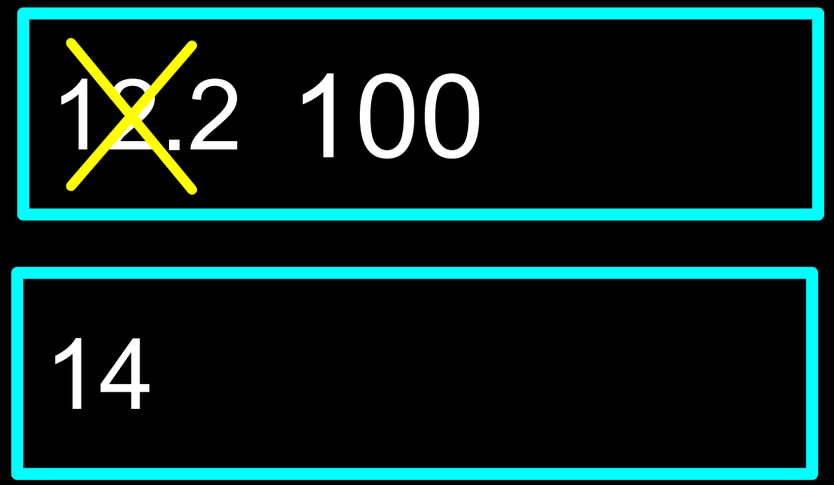
- Переменная — это поименованная область памяти, в которой можно хранить данные и извлекать их, используя «имя» переменной
- Программисты могут задавать имена переменных
- Значение переменной можно изменить

```
x = 12.2
y = 14
```

Переменные

- Переменная — это поименованная область памяти, в которой можно хранить данные и извлекать их, используя «имя» переменной
- Программисты могут задавать имена переменных
- Значение переменной можно изменить

```
x = 12.2
y = 14
x = 100
```



Правила назначения имен переменных в Пайтон

- Имя должно начинаться с буквы или подчеркивания
- Должно состоять из букв, чисел и подчеркиваний
- Пайтон чувствителен к регистру

```
Хорошо:      spam    eggs    spam23     _speed
Плохо:       23spam      #sign  var.12
Это три разные переменные:     spam   Spam   SPAM
```

Мнемонические имена переменных

- Поскольку мы как программисты можем выбирать и назначать имена переменным, существуют рекомендации или «лучшие практики» для именования переменных
- Лучше давать переменной имя, которое бы помогало нам помнить, что мы собираемся хранить в ней («мнемонический» = «облегчающий запоминание»)
- Это может сбивать с толку начинающих студентов, ведь хорошо названные переменные частенько «звучат» настолько хорошо, что могут показаться ключевыми словами

https://ru.wikipedia.org/wiki/Мнемоника

```
x1q3z9ocd = 35.0
x1q3z9afd = 12.50
x1q3p9afd = x1q3z9ocd * x1q3z9afd
print(x1q3p9afd)
```

Что делает этот
фрагмент кода?

```
x1q3z9ocd = 35.0
x1q3z9afd = 12.50
x1q3p9afd = x1q3z9ocd * x1q3z9afd
print(x1q3p9afd)
```

```
a = 35.0
b = 12.50
c = a * b
print(c)
```

Что делают эти
фрагменты кода?

```
x1q3z9ocd = 35.0
x1q3z9afd = 12.50
x1q3p9afd = x1q3z9ocd * x1q3z9afd
print(x1q3p9afd)
```

```
a = 35.0
b = 12.50
c = a * b
print(c)
```

Что делают эти фрагменты кода?

```
hours = 35.0
rate = 12.50
pay = hours * rate
print(pay)
```

Предложения или строки кода

```
x = 2
x = x + 2
print(x)
```

Присвоение значения

Присвоение с выражением

Функция вывода

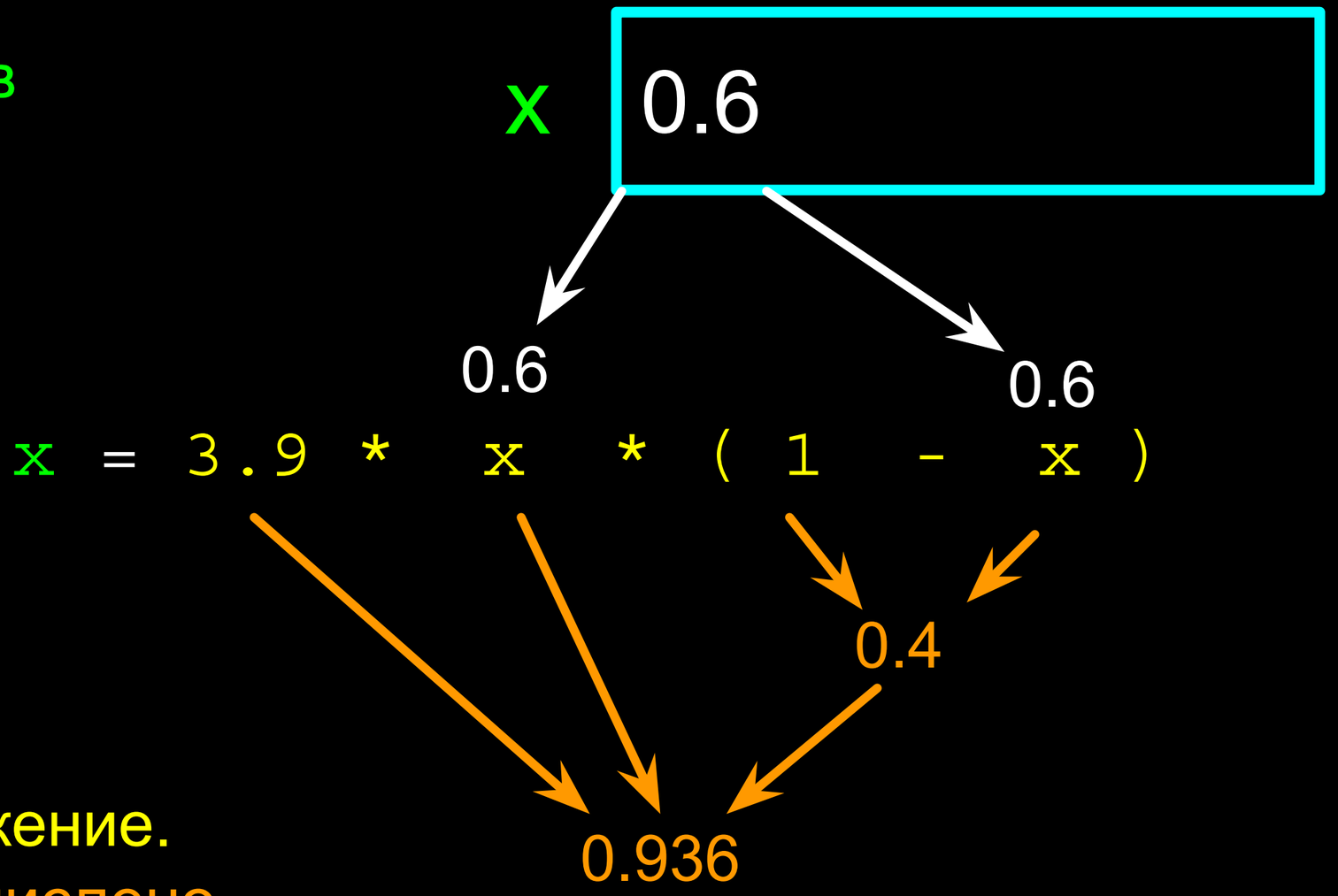
Переменная Оператор Константа Функция

Операция присваивания

- Мы присваиваем значение переменной, используя символ присваивания (=)
- Операция присваивания состоит из выражения с правой стороны и переменной слева для хранения результата

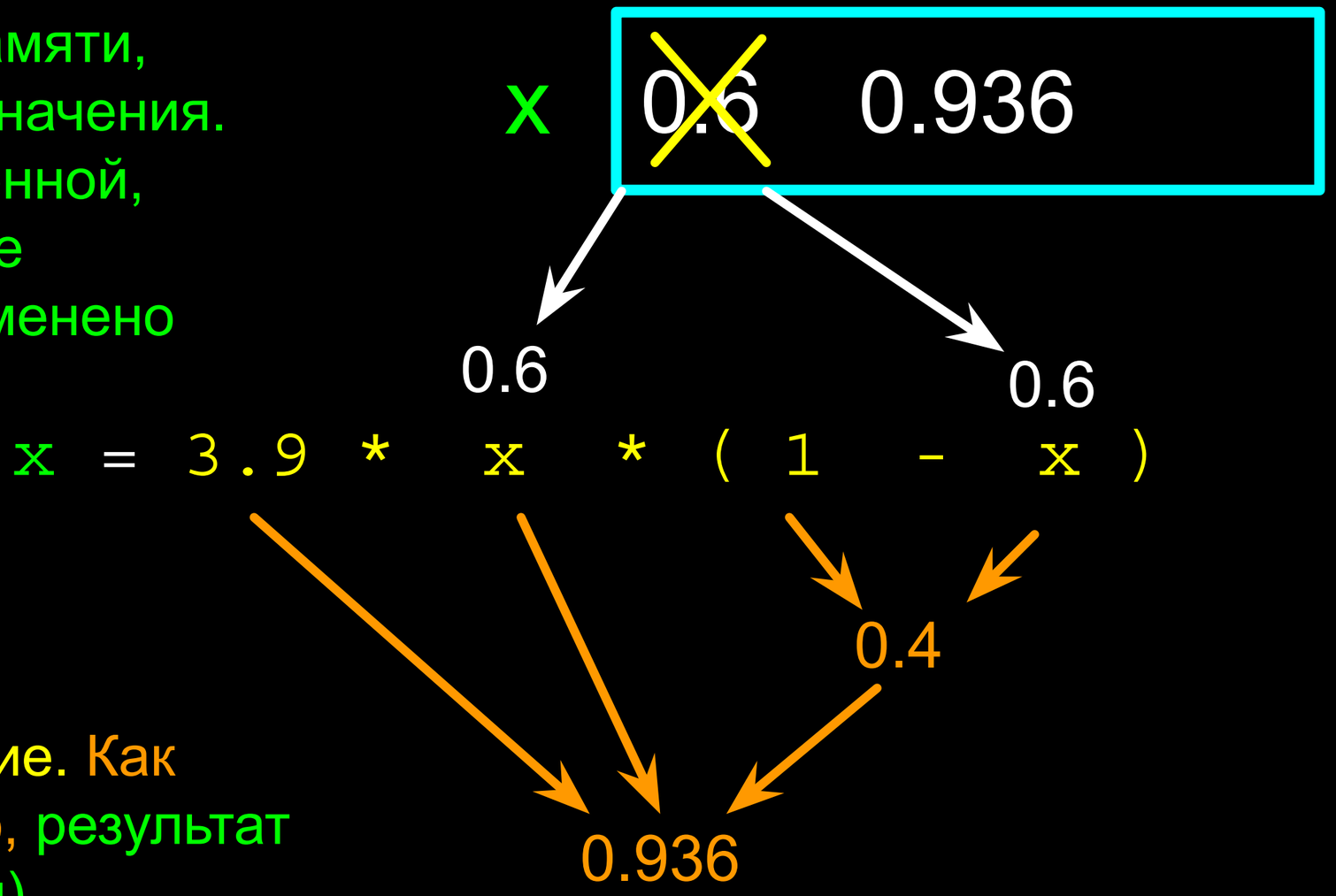
```
x =   3.9 * x * ( 1 - x )
```

Переменная — это место в памяти, используемое для хранения значения (0.6)



Правая часть — это выражение.
Как только выражение вычислено, результат помещается в (присваивается) x.

Переменная — это место в памяти, используемое для хранения значения. Значение, хранимое в переменной, может быть обновлено, старое значение (0.6) может быть заменено новым (0.936).



Правая часть — это выражение. Как только выражение вычислено, результат помещается в (присваивается) переменной в левой части (т.е., x).

Выражения…

Числовые выражения

- Из-за нехватки математических символов на клавиатуре компьютера, мы используем понятные компьютеру символы для передачи математических операций
- Звездочка — это умножение
- Возведение в степень выглядит не так, как в математике

Операция	Действие
+	Сложение
-	Вычитание
*	Умножение
/	Деление
**	Степень
%	Остаток

Числовые выражения

```
>>> xx = 2
>>> xx = xx + 2
>>> print(xx)
4
>>> yy = 440 * 12
>>> print(yy)
5280
>>> zz = yy / 1000
>>> print(zz)
5.28
```

```
>>> jj = 23
>>> kk = jj % 5
>>> print(kk)
3
>>> print(4 ** 3)
64
```

```
   4 R 3
  ------
5 | 23
    20
    --
     3
```

Операция	Действие
+	Сложение
-	Вычитание
*	Умножение
/	Деление
**	Степень
%	Остаток

Порядок вычислений

- Когда мы используем несколько операций, Пайтон должен знать, с какого начать
- Это называется «приоритет»
- Какой оператор имеет больший приоритет над остальными?

```
x = 1 + 2 * 3 - 4 / 5 ** 6
```

Правила приоритета

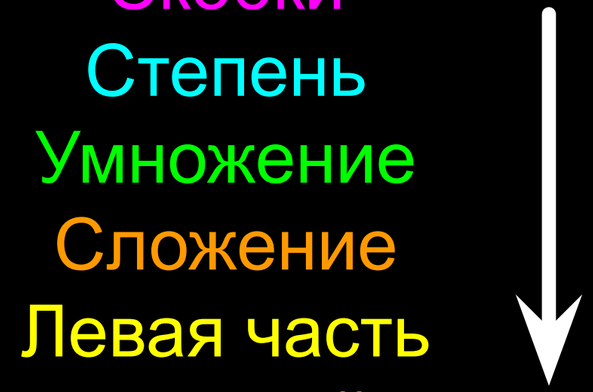
От наивысшего приоритета к низшему:

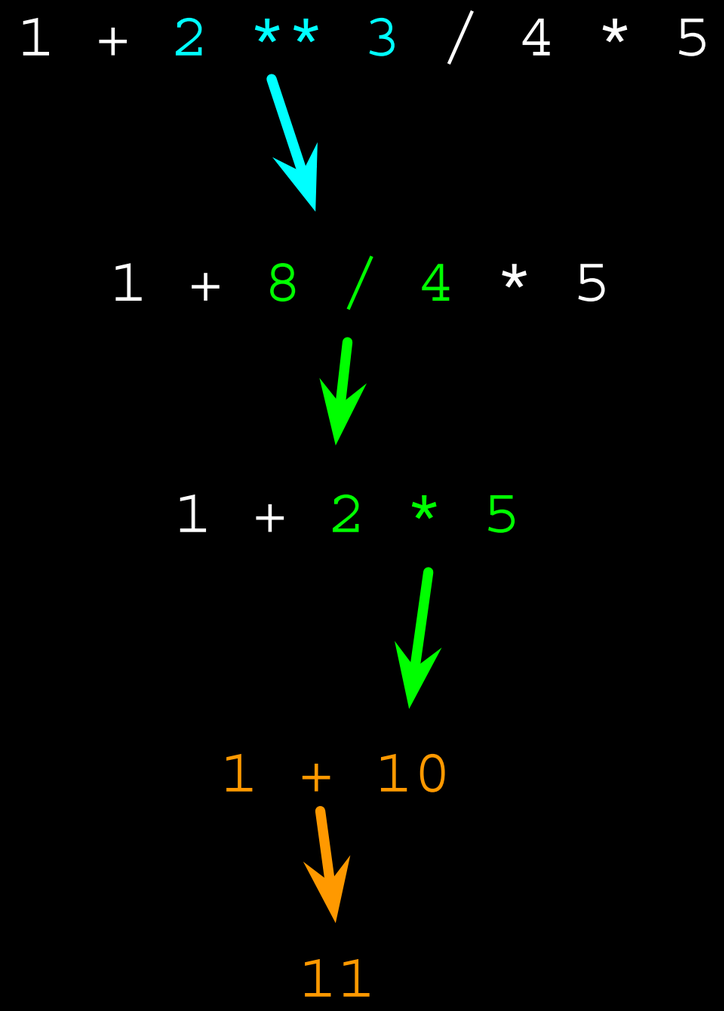
- Круглые скобки
- Возведение в степень
- Умножение, Деление, Остаток
- Сложение и Вычитание
- Левая часть по отношению к правой

Скобки
Степень
Умножение
Сложение
Левая часть
к правой

```
>>> x = 1 + 2 ** 3 / 4 * 5
>>> print(x)
11.0
>>>
```

Скобки
Степень
Умножение
Сложение
Левая часть
к правой





Приоритет операций

- Запомните правила приоритета (сверху вниз)
- При написании кода используйте скобки
- При написании кода старайтесь сохранять математические операции простыми, чтобы их было легко понять
- Разбивайте длинные серии математических операций, чтобы сделать их более понятными

Скобки
Степень
Умножение
Сложение
Левая часть к правой

Что такое «Тип»?

- В Пайтон переменные, литералы и константы имеют «тип» данных
- Пайтон различает целые числа и строки
- Например, символ “+” означает “сложение” в случае с числом, и “объединение”, если имеет дело с данными типа «строка»

```
>>> ddd = 1 + 4
>>> print(ddd)
5
>>> eee = 'Привет ' + 'всем'
>>> print(eee)
Привет всем
```

конкатенировать = объединять

Тип имеет значение

- Пайтон автоматически определяет «тип» всех объектов
- Некоторые операции запрещены
- Вы не можете сложить единицу (число) и строку
- В Пайтон мы можем узнать тип любого элемента, используя функцию type()

```
>>> eee = 'Привет ' + 'всем'
>>> eee = eee + 1
Traceback (most recent call last):
File "<stdin>", line 1, in
<module>TypeError: Can't convert
'int' object to str implicitly
>>> type(eee)
<class'str'>
>>> type('Привет')
<class'str'>
>>> type(1)
<class'int'>
>>>
```

Несколько числовых типов

- Числа бывают двух основных типов:
  - Целые числа:
    -14, -2, 0, 1, 100, 401233
  - Числа с плавающей точкой имеют десятичную часть: -2.5 , 0.0, 98.6, 14.0
- Существуют и другие числовые типы, но они — вариации чисел с плавающей точкой и целых чисел

```
>>> xx = 1
>>> type (xx)
<class 'int'>
>>> temp = 98.6
>>> type(temp)
<class'float'>
>>> type(1)
<class 'int'>
>>> type(1.0)
<class'float'>
>>>
```

Преобразование типов

- Если вы помещаете в одно выражение целое число и число с плавающей точкой, целое число преобразуется в число с плавающей точкой
- Вы можете контролировать тип с помощью встроенных функций: int() и float()

```
>>> print(float(99) + 100)
199.0
>>> i = 42
>>> type(i)
<class'int'>
>>> f = float(i)
>>> print(f)
42.0
>>> type(f)
<class'float'>
>>>
```

Деление целых чисел

Деление целых чисел дает результат с плавающей точкой

```
>>> print(10 / 2)
5.0
>>> print(9 / 2)
4.5
>>> print(99 / 100)
0.99
>>> print(10.0 / 2.0)
5.0
>>> print(99.0 / 100.0)
0.99
```

В отличие от Пайтон версии 2.x

Преобразование строк

- Вы так же можете использовать int() и float() для преобразования строк и чисел

- Если строка не содержит числовых символов, вы получите сообщение об ошибке

```
>>> sval = '123'
>>> type(sval)
<class 'str'>
>>> print(sval + 1)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object
to str implicitly
>>> ival = int(sval)
>>> type(ival)
<class 'int'>
>>> print(ival + 1)
124
>>> nsv = 'hello bob'
>>> niv = int(nsv)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: invalid literal for int()
with base 10: 'x'
```

Пользовательский ввод

- Мы можем дать указание Пайтон приостановиться и прочесть данные от пользователя, используя функцию input()

```
nam = input('Кто ты? ')
print('Привет', nam)
```

- Функция input() возвращает строку

Кто ты? Чак
Привет Чак

Преобразование пользовательского ввода

- Если мы запрашиваем у пользователя число, нам необходимо преобразовать его из строки в число, используя функцию преобразования типа

- Позже мы разберем ситуации с некорректными входными данными

```
inp = input('Этаж в Европе?')
usf = int(inp) + 1
print('Этаж в США', usf)
```

Этаж в Европе? 0
Этаж в США 1

Комментарии в Пайтон

- Все написанное после символа # игнорируется Пайтон
- Зачем нужно комментировать?
  - Описать, что будет происходить в блоке кода
  - Указать автора кода или другую вспомогательную информацию
  - Заблокировать строку кода, возможно, временно

```
# Запрашивает имя файла и открывает его
name = input('Введите имя файла:')
handle = open(name, 'r')

# Подсчитывает частоту появления каждого
слова
counts = dict()
for line in handle:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word,0) + 1

# Находит самое часто встречающееся слово
bigcount = None
bigword = None
for word,count in counts.items():
    if bigcount is None or count > bigcount:
        bigword = word
        bigcount = count

# Выводит результат
print(bigword, bigcount)
```

Задание

Напишите программу, предлагающую пользователю ввести количество часов и почасовую ставку для расчета заработной платы.

```
Укажите количество часов: 35
Укажите ставку: 2.75

Оплата: 96.25
```