Языки программирования

Лекция 3

Операторы и выражения

- 2 + 3
- Операторы – это некий функционал, производящий какие-либо действия, который может быть представлен в виде символов, как например +, или специальных зарезервированных слов. Операторы могут производить некоторые действия над данными, и эти данные называются операндами.

Операторы присваивания

- = – присваивает переменной значение:
- >>> x = 5; x
- 5

- += – увеличивает значение переменной на указанную величину:
- >>> x = 5; x += 10 # Эквивалентно x = x + 10
- 15

- -= – уменьшает значение переменной на указанную величину;
- *=; /=; //=; %=; **=

Приоритет выполнения операторов

- 1. -x, +x, ~x, **
- Например, выражение:
- -10 ** -2 эквивалентно следующей расстановке скобок: -(10 ** (-2))
- 2. *, %, /, //
- 3. +, -
- 4. <<, >> – двоичные сдвиги.
- 5. & – двоичное И.
- 6. ^ – двоичное исключающее ИЛИ.
- 7. | – двоичное ИЛИ.
- 8. =, +=, -=, *=, /=, //=, %=, **= – присваивание.

Математические операторы

- +, −, *, /
- //
- >>> 10 // 3
- 3
- % – остаток от деления:
- >>> 10 % 3
- 1
- ** – возведение в степень:
- >>> 10**2, 10.0**2
- (100,100.0)
- унарный минус (−) и унарный плюс (+):
- >>> +10, +10.0, -10, -(-10)
- (10, 10.0, -10, 10)

Операторы для работы с последовательностями

- +
- >>> print ("Строка1" + "Строка2") # конкатенация строк
- Строка1Строка2

- >>> [1, 2, 3] + [4, 5, 6] # списки
- [1, 2, 3, 4, 5, 6]

- >>> (1, 2, 3) + (4, 5, 6) # кортежи
- (1, 2, 3, 4, 5, 6)

- *
- >>> "s" * 10 # строки
- ’ssssssssss’

- >>> [1, 2] * 2 # списки
- [1, 2, 1, 2]

- >>> (1, 2) * 2 # кортежи
- (1, 2, 1, 2)

- in – проверка на вхождение.
- >>> "Солнце" in "Солнце светит ярко" # строки
- True

- >>> "Луна" in "Солнце светит ярко" # строки
- False

- >>> 2 in [1, 2, 3] # списки
- True

- >>> 6 in (1, 2) # кортежи
- False

- not in – проверка на невхождение

Операторы сравнения

- ==
- >>> 2 == 2, 6 == 5
- (True, False)

- !=, <, >, <=, >=
- >>> 1 >= 1, 1 > 5
- (True, False)

- is
- >>> x = y = [1, 2]
- >>> x is y
- True

- >>> x = [1, 2]; y = [1, 2]
- >>> x is y
- False

- is not

- and
- 1 < 5 and 2 < 5 # True and True = True
- True

- or
- 1 < 5 or 2 > 5 # True or False = True
- True

- not
- >>> x = 1; y = 1
- >>> not (x == y), not x == y
- (False, False)

- >>> x = 10
- >>> 1 < x < 20, 11 < x < 20
- (True, False)

Операторы сравнения в порядке убывания приоритета:

- 1. <, >, <=, >=, ==, !=, <>, is, is not, in, not in.
- 2. not – логическое отрицание.
- 3. and – логическое И.
- 4. or – логическое ИЛИ.

Модуль math. Математические функции

- import math

- math.pi
- 3.141592653589793

- math.e
- 2.718281828459045

- sin(), cos(), tan()
- asin(), acos(), atan()

- degrees()

- math.degrees(math.pi)
- 180.0

- radians()

- math.radians(180.0)
- 3.141592653589793

- exp()

Математические функции

- log(<Число>[, <Основание>])
- log10(), log2(), sqrt()
- ceil()
- math.ceil(5.39), math.ceil(5.51)
- 6, 6
- floor()

- pow(<Число>, <Степень>)
- fabs()

- fmod()
- math.fmod(10,5), math.fmod(10,3) # эквивалентно 10 % 5, 10 %3
- 0.0, 1.0

- factorial() – факториал числа.

Оператор if

- number = 23
- guess = int(input('Введите целое число : '))
- **if** guess == number:
- **print**('Поздравляю, вы угадали,') *# Здесь начинается новый блок*
- **print**('(хотя и не выиграли никакого приза!)') *# Здесь заканчивается новый блок*
- **elif** guess < number:
- **print**('Нет, загаданное число немного больше этого.') *# Ещё один блок # Внутри блока вы можете выполнять всё, что угодно ...*
- **else**:
- **print**('Нет, загаданное число немного меньше этого.') *# чтобы попасть сюда, guess должно быть больше, чем number*
- **print**('Завершено') *# Это последнее выражение выполняется всегда после выполнения оператора if*

Оператор if

- Введите целое число : 50
- Нет, загаданное число немного меньше этого.
- Завершено

- Введите целое число : 22
- Нет, загаданное число немного больше этого.
- Завершено

- Введите целое число : 23
- Поздравляю, вы угадали,
- (хотя и не выиграли никакого приза.)
- Завершено

```
if True:
    print('Да, это верно.')
```

Оператор while

```
i = 5
while i < 10:
    print(i)
    i = i + 2
```

- 5
- 7
- 9

Оператор while

```
number = 23
running = True

while running:
    guess = int(input('Введите целое число : '))

    if guess == number:
        print('Поздравляю, вы угадали.')
        running = False # это останавливает цикл while
    elif guess < number:
        print('Нет, загаданное число немного больше этого')
    else:
        print('Нет, загаданное число немного меньше этого.')
else:
    print('Цикл while закончен.') # Здесь можете выполнить всё что вам ещё нужно

print('Завершение.')
```

Оператор while

- Введите целое число : 50
- Нет, число несколько меньше.
- Введите целое число : 22
- Нет, число несколько больше.
- Введите целое число : 23
- Поздравляю, вы угадали.
- Цикл while закончен.
- Завершение.

Цикл for

- for i in ’hello world’:
- print(i * 2, end=’’)

- hheellllloo wwoorrlldd

- **for** i **in** range(1, 5):
- **print**(i)
- **else**:
- **print**('Цикл for закончен')

- 1
- 2
- 3
- 4
- Цикл for закончен

Оператор break

- for i in ’hello world’:
- if i == ’o’:
- break
- print(i * 2, end = ’’)

- hheellll

- **while** True:
- s = input('Введите что-нибудь : ')
- **if** s == 'выход':
- **break**
- **print**('Длина строки:', len(s))

print('Завершение')

- Введите что-нибудь : Программировать весело.
- Длина строки: 23
- Введите что-нибудь : Если работа скучна,
- Длина строки: 19
- Введите что-нибудь : Чтобы придать ей весёлый тон –
- Длина строки: 30
- Введите что-нибудь : используй Python!
- Длина строки: 23
- Введите что-нибудь : выход
- Завершение

Оператор continue

```
for i in 'hello world':
    if i == 'o':
        continue
    print(i * 2, end = '')
```

```
hheellll wwrrlldd
```

```
while True:
    s = input('Введите что-нибудь : ')
    if s == 'выход':
        break
    if len(s) < 3:
        print('Слишком мало')
        continue
    print('Введённая строка достаточной длины')
# Разные другие действия здесь...
```

```
Введите что-нибудь : a
Слишком мало
Введите что-нибудь : 12
Слишком мало
Введите что-нибудь : абв
Введённая строка достаточной длины
Введите что-нибудь : выход
```

Упражнения и контрольные вопросы

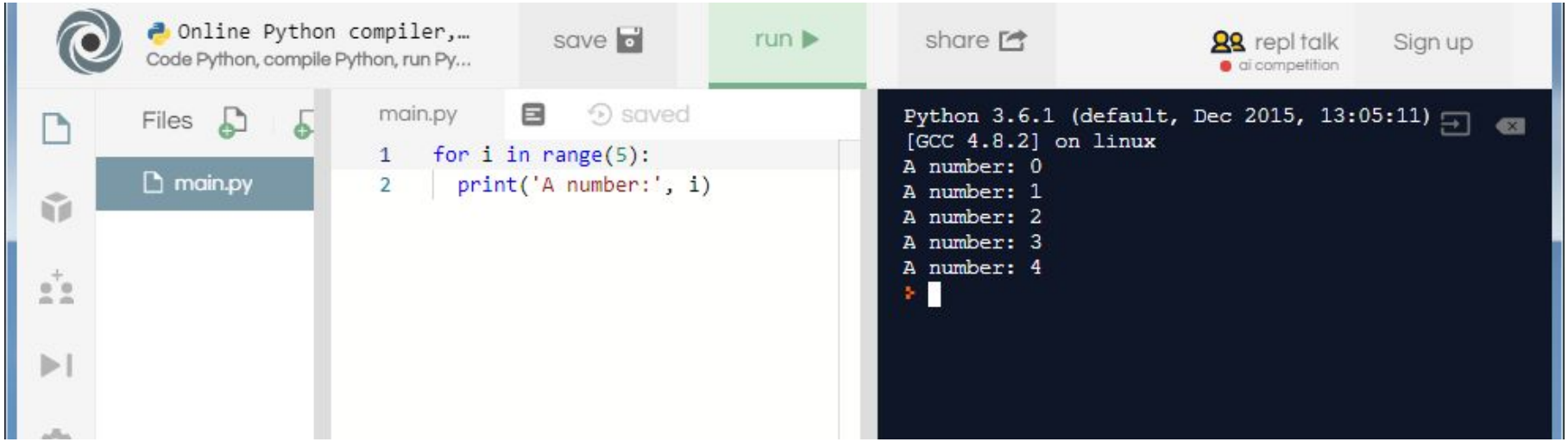
- 1. Для чего нужны операторы?
- 2. Какой результат получится после выполнения следующего кода:
- >>> True + 2
- 3. Какой результат получится после выполнения следующего кода:
- >>> bool(-20)
- 4. Какой результат получится после выполнения следующего кода:
- >>> bool(0.1)
- 5. Какой результат получится после выполнения следующего кода:
- >>> bool("0")
- 6. Какой результат получится после выполнения следующего кода:
- >>> bool(0.0)
- 7. Какой результат получится после выполнения следующего кода:
- >>> bool("")
- 8. Какой результат получится после выполнения следующего кода:
- >>> bool([])
- 9. Какой результат получится после выполнения следующего кода:
- >>> bool(())
- 10. Какой результат получится после выполнения следующего кода:
- >>> bool(None)
- 11. Какой результат получится после выполнения следующего кода:
- >>> 20 and 10
- 12. Какой результат получится после выполнения следующего кода:
- >>> 10 and 20
- 13. Какой результат получится после выполнения следующего кода:
- >>> 1 > 5 or 2 < 5
- 14. Какой результат получится после выполнения следующего кода:
- >>> 10 or 20
- 15. Какой результат получится после выполнения следующего кода:
- >>> 20 or 0
- 16. Какой результат получится после выполнения следующего кода:
- >>> math.pow(11,2)
- 17. Какой результат получится после выполнения следующего кода:
- >>> math.factorial(5)

Ответы

- 1. Операторы позволяют произвести с данными определенные действия.
- 2. Ответ: 3.
- 3. Ответ: True.
- 4. Ответ: True.
- 5. Ответ: True.
- 6. Ответ: False.
- 7. Ответ: False.
- 8. Ответ: False.
- 9. Ответ: False.
- 10. Ответ: False.
- 11. Ответ: 10.
- 12. Ответ: 20.
- 13. Ответ: True.
- 14. Ответ: 10.
- 15. Ответ: 20.
- 16. Ответ: 121.0.
- 17. Ответ: 120.

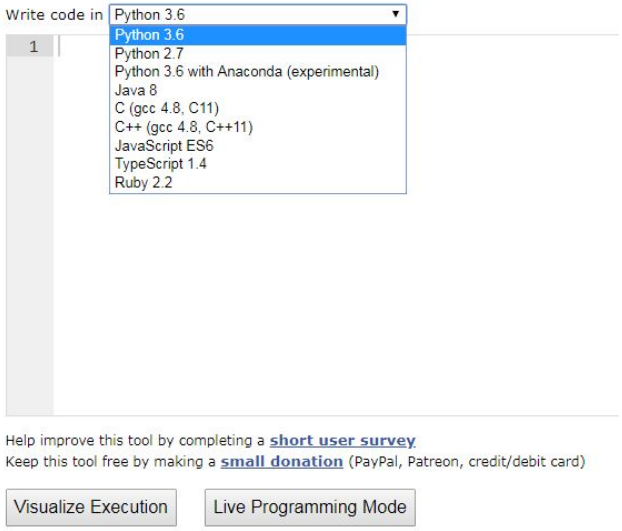
Интерпретаторы

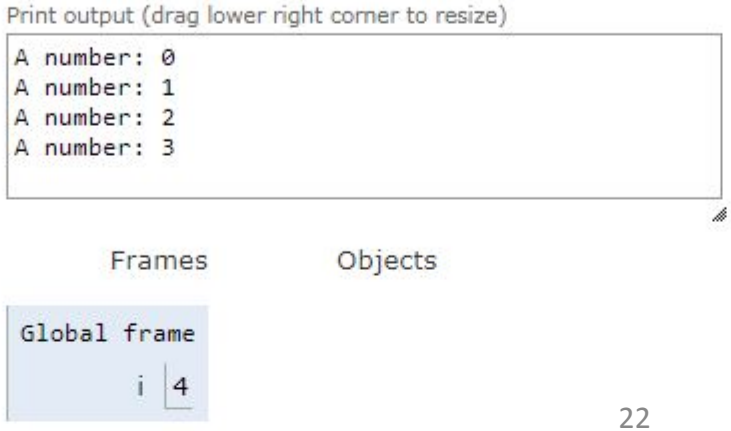
- https://repl.it/languages/python3



Интерпретаторы

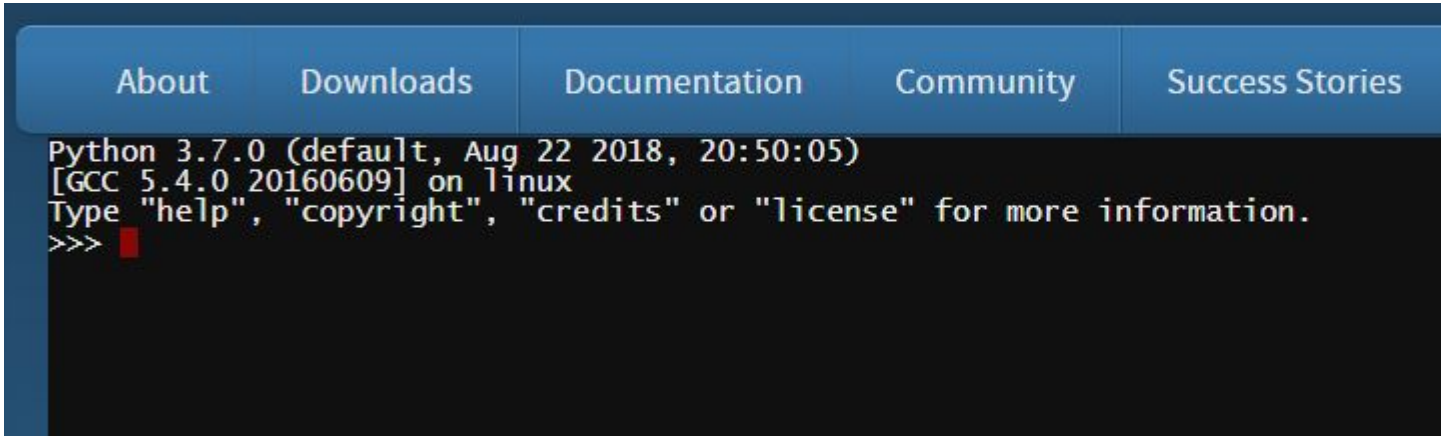
- www.pythontutor.com/visualize.html





Интерпретаторы

- https://www.python.org/shell/ - python консоль



Интерпретаторы

- http://pythonfiddle.com/

Интерпретаторы

- https://ideone.com/

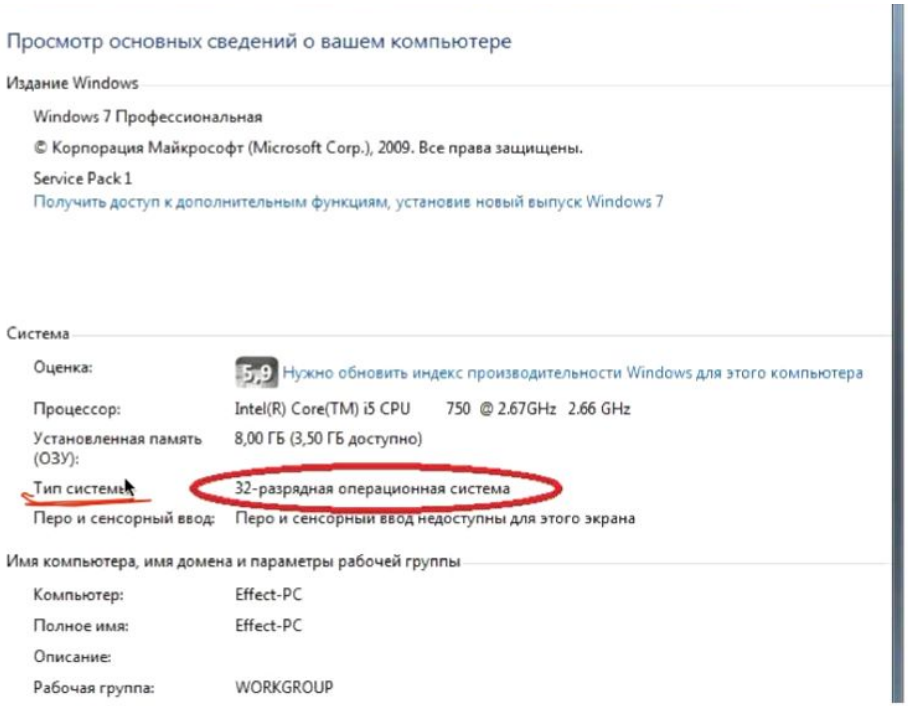
Установка iPython

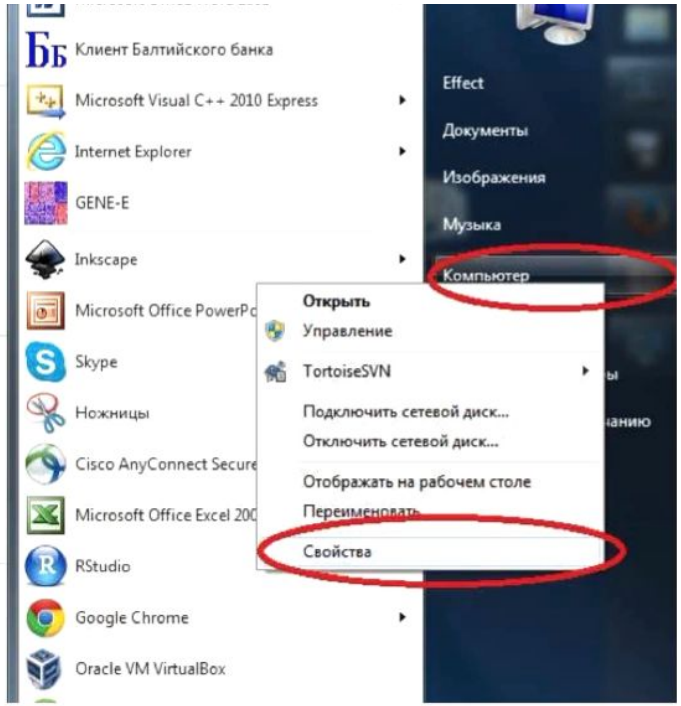
- 1. Установка Miniconda по ссылке с сайта
- 2. Запуск командной строки из меню Пуск - Anaconda Prompt с правами администратора.
- В командой строке необходимо выполнить команды:
- 3. Обновление pip командой: python -m pip install --upgrade pip
- 4. Установка jupyter командой: pip install jupyter
- 5. Запуск оболочки: jupyter notebook.

Установка iPython

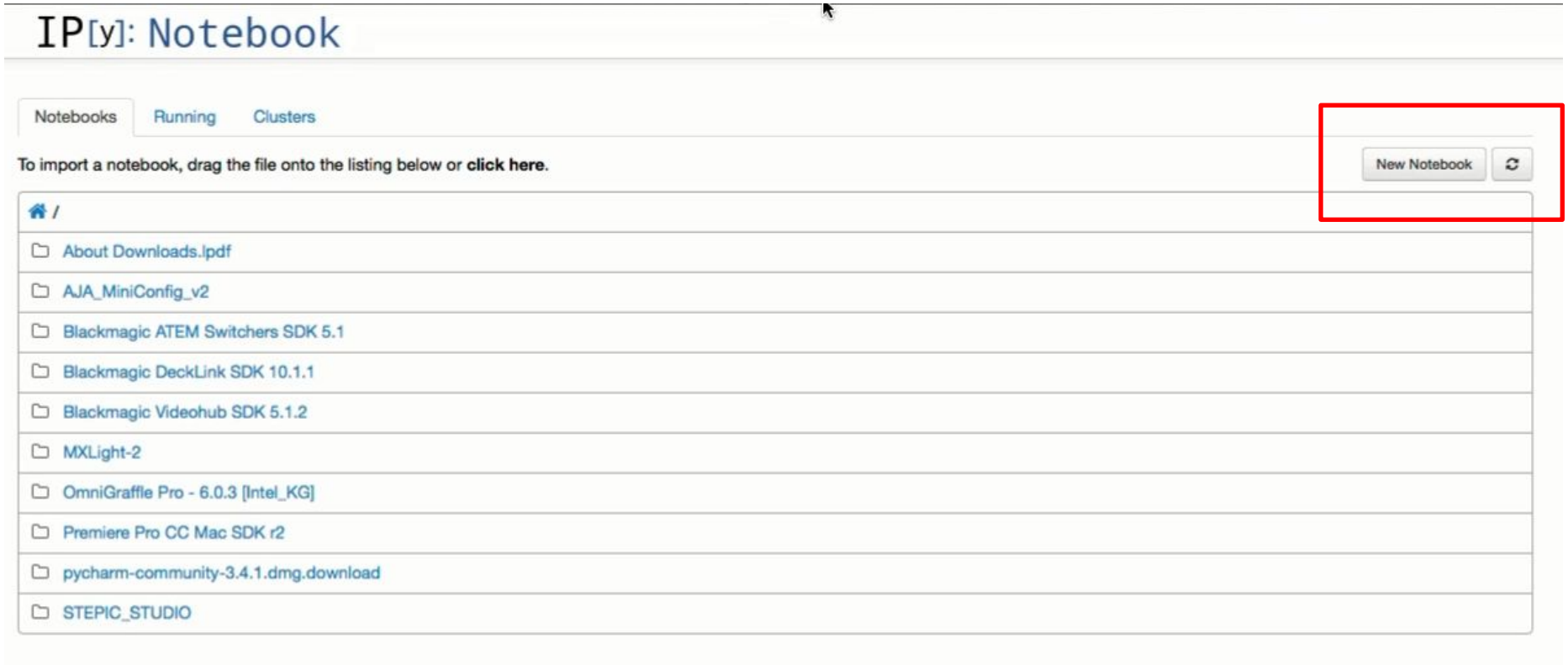
Miniconda

	Windows	Mac OS X	Linux
Python 3.7	64-bit (exe installer) 32-bit (exe installer)	64-bit (bash installer) 64-bit (.pkg installer)	64-bit (bash installer) 32-bit (bash installer)
Python 2.7	64-bit (exe installer) 32-bit (exe installer)	64-bit (bash installer) 64-bit (.pkg installer)	64-bit (bash installer) 32-bit (bash installer)

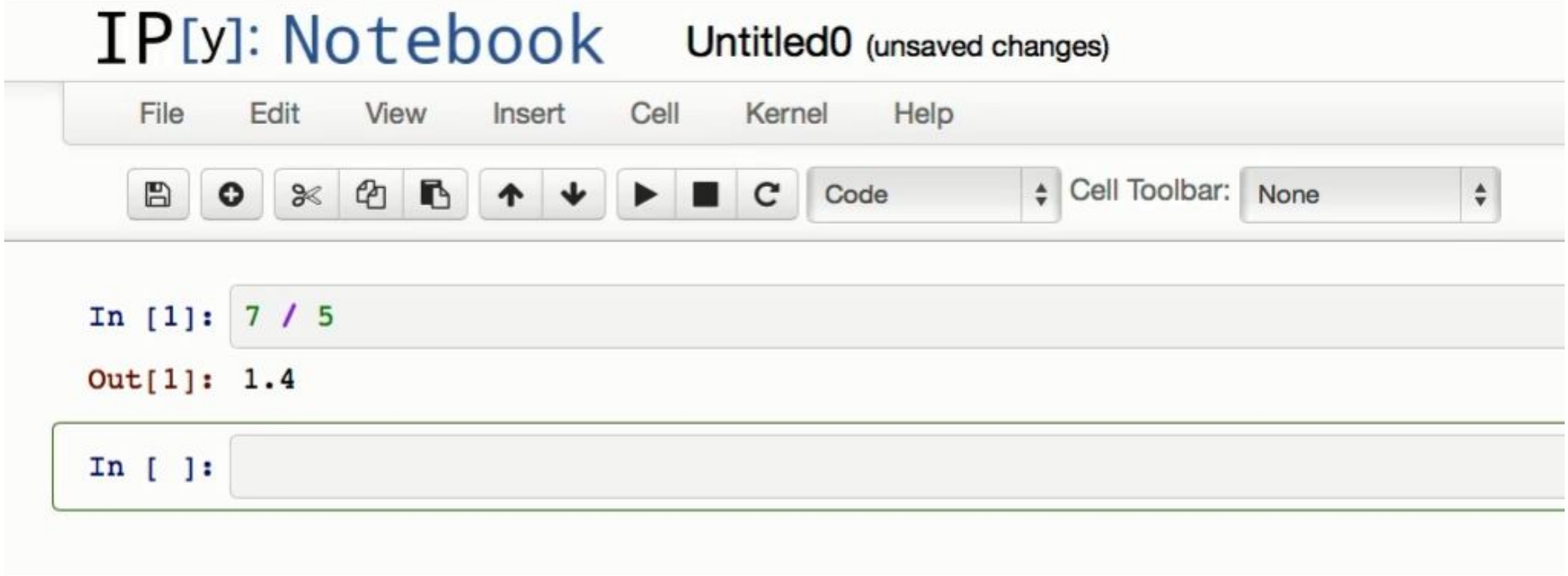




Установка iPython



Установка iPython



Операции с целыми числами

IP[y]: Notebook Untitled1 (unsaved changes)

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None Env: np18py33-1.9

```
In [3]: 2 + 5
Out[3]: 7

In [4]: 10 - 5
Out[4]: 5

In [5]: 6 * 7
Out[5]: 42

In [6]: 12345678 * 98765432876543218765
Out[6]: 1219326231824416331956247670

In [7]: 3 + 5 * 4
Out[7]: 23

In [8]: (3 + 5) * 4
Out[8]: 32

In [ ]:
```

Операции с целыми числами

```
In [10]: 40 // 8
Out[10]: 5

In [11]: 42 // 8
Out[11]: 5

In [12]: 42 % 8
Out[12]: 2
```

Операция целочисленного деления

Остаток от деления

Задание. Составьте выражение для вычисления указанной ниже формулы и вставьте в поле ответа вывод интерпретатора после вычисления этого выражения.

$$\frac{42}{4 + 2 \cdot (-2)}$$

```
Traceback (most recent call last):
  File "main.py", line 1, in <module>
    print (42/0)
ZeroDivisionError: division by zero
```

Операции с целыми числами

```
In [21]: 239 % 10
Out[21]: 9

In [22]: 239 // 10
Out[22]: 23

In [23]: 2 ** 5
Out[23]: 32
```

Вычисление последней цифры числа

Само число без последней цифры

Возведение в степень

Задание. Составьте выражение для вычисления указанной ниже формулы и вставьте в поле ответа вывод интерпретатора после вычисления этого выражения.

$$\frac{42}{4 + 2 \cdot (-2)}$$

```
Traceback (most recent call last):
  File "main.py", line 1, in <module>
    print (42/0)
ZeroDivisionError: division by zero
```

Операции с целыми числами

Задание. Составьте выражение для вычисления в интерпретаторе Python 3 и вставьте в поле ответа результат вычисления:

$$2014^{14}$$

Ответ. 18064773765607550801425251468864907833685590016

Заметьте, насколько большое это число. Во многих других языках программирования работать со значениями такого порядка гораздо сложнее.

Унарные операторы

```
In [24]: - (42)
Out[24]: -42

In [25]: + (42)
Out[25]: 42

In [26]: +-+42
Out[26]: -42

In [27]: -*42
  File "<ipython-input-27-85d397208f73>", line 1
    -*42
     ^
SyntaxError: invalid syntax
```

Операции с целыми числами

```
In [28]: print('test')
         * 23
```

Ошибка во второй строке

```
test
  File "<ipython-input-28-21eaeacd21f5>", line 2
    * 23
      ^
SyntaxError: can use starred expression only as assignment target
```

```
In [29]: 5 // 0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-29-8798db347eeb> in <module>()
----> 1 5 // 0

ZeroDivisionError: integer division or modulo by zero
```

Числа с плавающей точкой

```
In [1]: 0.5 + 0.3
Out[1]: 0.8

In [2]: 5 / 2
Out[2]: 2.5

In [3]: 5 // 2
Out[3]: 2

In [4]: 1 / 3
Out[4]: 0.3333333333333333

In [5]: 0.3 + 0.3 + 0.3
Out[5]: 0.8999999999999999
```

Целая часть и дробная часть

Результат без дробной части

Потеря точности

Возможна погрешность

```
In [7]: 9 ** 0.5
Out[7]: 3.0

In [8]: 5e-1
Out[8]: 0.5

In [10]: 1234e2
Out[10]: 123400.0
```

Квадратный корень из числа

Экспоненциальная запись

Числа с плавающей точкой

Задание 1. Запишите число 1.2345e3 в виде десятичной дроби.

Задание 2. Запишите число 1.2345e-3 в виде десятичной дроби.

Задание 3. Составьте выражение для вычисления в интерпретаторе Python 3 и вставьте в поле ответа результат вычисления:

```
7//3
```

Некоторые стандартные типы

Числовые

целые числа – int

вещественные (с плавающей точкой) – float

логические – bool

Строковые

строки – str

Все перечисленные типы являются неизменяемыми

Преобразование типов

int(x) – преобразование к целому числу

int (2.3) → 2

float(x) – преобразование к числу с плавающей точкой

float(5) → 5.0

Типы данных

- Задание 1. Составьте выражение для вычисления в интерпретаторе Python 3 и вставьте в поле ответа результат вычисления. **Приведите к целому типу число 2.99**
- Задание 2. Составьте выражение для вычисления в интерпретаторе Python 3 и вставьте в поле ответа результат вычисления.
- **Приведите к целому типу число -1.6**

- Заметьте, что приведение к целому типу вещественного числа соответствует отбрасыванию дробной составляющей, что соответствует округлению в сторону 00.