

Python 1: введение в язык Python

Содержание

1 Цель работы.....	2
2 Краткие теоретические сведения.....	2
3 Оборудование рабочего места.....	2
4 Задание.....	3
5 Содержание отчета.....	3
6 Порядок выполнения работы.....	4
6.1 Среда программирования IDLE, режим командной строки.....	4
6.2 Операторы языка Python.....	6
6.3 Переменные и типы данных.....	11
6.4 Встроенные функции Python.....	13
6.5 Определение и преобразование типов данных.....	14
6.6 Написание и сохранение программ, комментарии.....	15
6.7 Пример программы.....	21
6.8 Подключение модулей. Модуль math.....	23
6.9 Самостоятельная работа.....	26
7 Контрольные вопросы.....	29
8 Информационные источники.....	29

1 Цель работы

Изучить основные термины и приемы работы с языком Python в среде разработки IDLE.

2 Краткие теоретические сведения

Программирование — это процесс проектирования, написания и отладки программ. Что такое программа? Программа — это последовательность инструкций, предназначенная для исполнения устройством управления вычислительной машины [1]. Официальное определение Программы дано в нескольких документах, например:

Программа — данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определённого алгоритма. [2]

Программа — представленная в объективной форме совокупность данных и команд, предназначенных для функционирования ЭВМ и других компьютерных устройств с целью получения определённого результата, включая подготовительные материалы, полученные в ходе разработки программы для ЭВМ, и порождаемые ею аудиовизуальные отображения. [3]

Python – язык программирования высокого уровня, сочетающий в себе дружелюбное отношение к пользователю и большое количество библиотек, позволяющих решать прикладные задачи. Это делает язык Python одним из лучших языков для «прикладного программирования» в случаях, когда программу пишут не профессиональные программисты, а ученые, исследователи или «обычные» пользователи, для решения задачи которых не достаточно возможностей уже существующего прикладного ПО.

Python не требует явного объявления переменных, является регистрозависимым (переменная var не эквивалентна переменной Var или VAR — это три разные переменные) объектно-ориентированным языком.

3 Оборудование рабочего места

Персональный компьютер под управлением ОС Linux или Windows с установленной средой программирования IDLE и языком программирования Python версии 3.4 и выше.

4 Задание

- Ознакомиться с интерфейсом среды программирования IDLE.
- Изучить операторы языка Python.
- Изучить способы создания переменных на языке Python.
- Изучить типы данных, используемые в языке Python, способы их определения и изменения.
- Изучить встроенные функции языка Python, в т.ч. функции ввода-вывода.
- Познакомиться с процедурой подключения модулей в языке Python.
- Написать и выполнить программу — пример.
- Выполнить самостоятельные задания.
- Подготовить отчет.

5 Содержание отчета

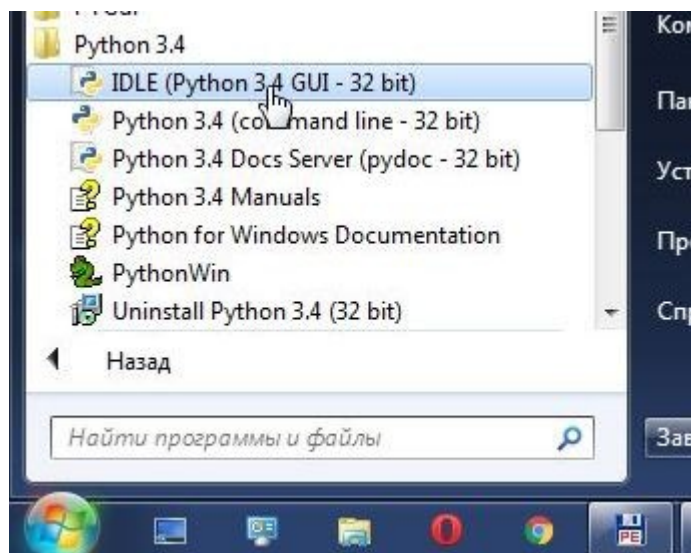
- Файл отчета — текстовый документ.
- Файлы *.py, содержащие решение примеров и самостоятельных задач.

6 Порядок выполнения работы

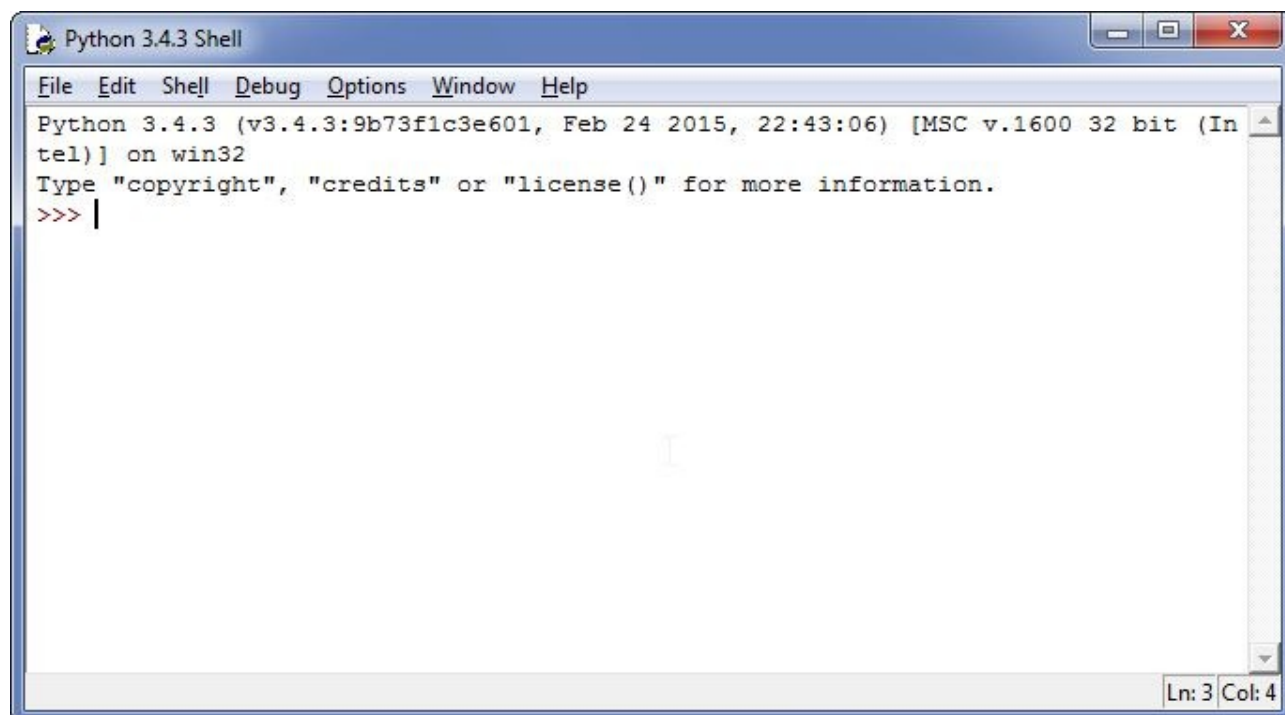
6.1 Среда программирования IDLE, режим командной строки

IDLE (Integrated DeveLopment Environment) — интегрированная среда программирования для работы с языком Python, которая поставляется вместе с Python и может использоваться на многих платформах, среди которых Windows, Mac OS и Unix-подобные ОС.

Для запуска среды программирования в ОС Windows 7 необходимо выбрать в главном меню команду **Пуск - IDLE (Python 3.4 GUI - 32 bit)** (версия языка Python или разрядность могут отличаться):

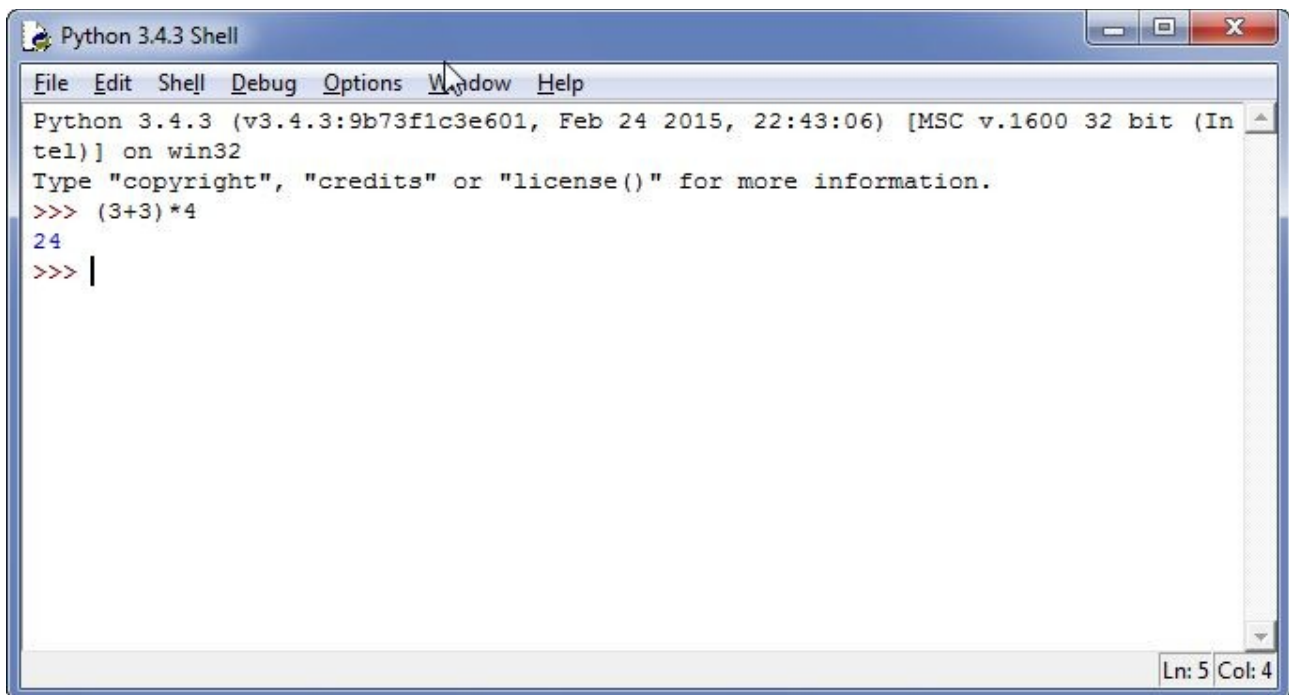


После этого откроется главное окно программы — т. е. командное окно. Его особенностью является то, что оно предоставляет интерфейс, схожий с интерфейсом командной строки Windows или терминала Linux, позволяющий работать с языком Python в интерактивном режиме:



В этом окне можно выполнять простейшие операции с использованием языка Python. Для подтверждения ввода используется клавиша Enter, результат работы, если он не записывается в переменную, автоматически выводится на экран.

Например, вычислим значение математического выражения:

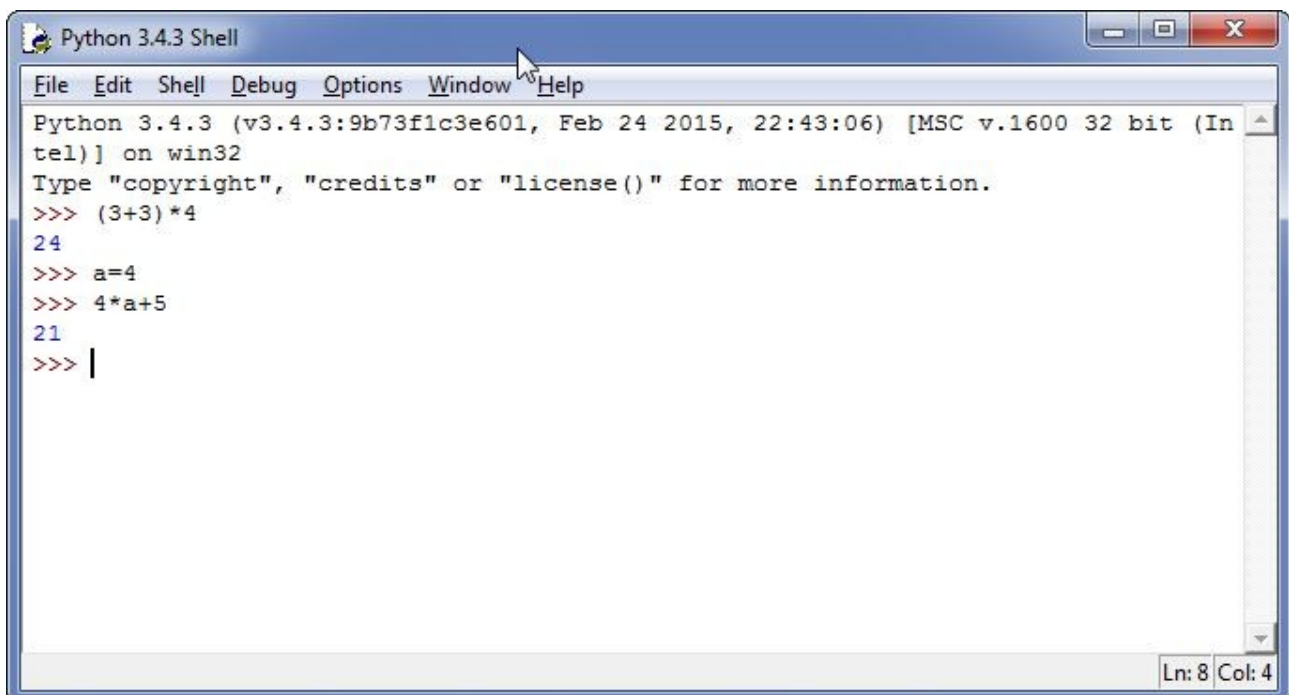


The screenshot shows a window titled "Python 3.4.3 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The text area contains the following text:

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> (3+3)*4
24
>>> |
```

The status bar at the bottom right indicates "Ln: 5 Col: 4".

зададим переменную и используем её в дальнейших вычислениях:



The screenshot shows the same "Python 3.4.3 Shell" window. The text area now contains:

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> (3+3)*4
24
>>> a=4
>>> 4*a+5
21
>>> |
```

The status bar at the bottom right indicates "Ln: 8 Col: 4".

В приведенном примере мы объявили переменную, присвоили ей значение (при этом задали тип данных благодаря динамической типизации) используя оператор присваивания и выполнили несколько математических операций благодаря арифметическим операторам. Рассмотрим операторы языка более подробно.

Для перемещения по ранее введенным командам можно использовать сочетания клавиш **Alt + P** и **Alt + N**.

6.2 Операторы языка Python

Оператор — это наименьшая автономная часть языка программирования, его команда. Операнды — объекты, к которым применяется действие оператора. В языке Python можно выделить следующие типы операторов:

- Арифметические операторы
- Операторы сравнения (реляционные)
- Операторы присваивания
- Побитовые операторы
- Логические операторы
- Операторы членства (Membership operators)
- Операторы тождественности (Identity operators)

Арифметические операторы

оператор	описание	примеры
+	Сложение - суммирует значения слева и справа от оператора	15 + 5 в результате будет 20 20 + -3 в результате будет 17 13.4 + 7 в результате будет 20.4
-	Вычитание - вычитает правый операнд из левого	15 - 5 в результате будет 10 20 - -3 в результате будет 23 13.4 - 7 в результате будет 6.4
*	Умножение - перемножает операнды	5 * 5 в результате будет 25 7 * 3.2 в результате будет 22.4 -3 * 12 в результате будет -36
/	Деление - делит левый операнд на правый	15 / 5 в результате будет 3 5 / 2 в результате будет 2 5.0 / 2 в результате будет 2.5
%	Деление по модулю - делит левый операнд на правый и возвращает остаток	6 % 2 в результате будет 0 7 % 2 в результате будет 1 13.2 % 5 в результате будет 3.2
**	Возведение в степень - возводит левый операнд в степень правого	5 ** 2 в результате будет 25 2 ** 3 в результате будет 8 -3 ** 2 в результате будет -9
//	Целочисленное деление - делит левый операнд на правый и возвращает целую часть результата (дробная часть отбрасывается)	12 // 5 в результате будет 2 4 // 3 в результате будет 1 25 // 6 в результате будет 4

Операторы сравнения

оператор	описание	примеры
==	Проверяет равны ли оба операнда. Если да, то условие становится истинным.	5 == 5 в результате будет True True == False в результате будет False "hello" == "hello" в результате будет True
!=	Проверяет равны ли оба операнда. Если нет, то условие становится истинным.	12 != 5 в результате будет True False != False в результате будет False "hi" != "Hi" в результате будет True
>	Проверяет больше ли значение левого операнда, чем значение правого. Если да, то условие становится истинным.	5 > 2 в результате будет True. True > False в результате будет True. "A" > "B" в результате будет False.
<	Проверяет меньше ли значение левого операнда, чем значение правого. Если да, то условие становится истинным.	3 < 5 в результате будет True. True < False в результате будет False. "A" < "B" в результате будет True.
>=	Проверяет больше или равно значение левого операнда, чем значение правого. Если да, то условие становится истинным.	1 >= 1 в результате будет True. 23 >= 3.2 в результате будет True. "C" >= "D" в результате будет False.
<=	Проверяет меньше или равно значение левого операнда, чем значение правого. Если да, то условие становится истинным.	4 <= 5 в результате будет True. 0 <= 0.0 в результате будет True. -0.001 <= -36 в результате будет False.

Операторы присваивания

оператор	описание	примеры
=	Присваивает значение правого операнда левому.	$c = 23$ присвоит переменной c значение 23
+=	Прибавит значение правого операнда к левому и присвоит эту сумму левому операнду.	$c = 5$ $a = 2$ $c += a$ равносильно: $c = c + a$. c будет равно 7
-=	Отнимает значение правого операнда от левого и присваивает результат левому операнду.	$c = 5$ $a = 2$ $c -= a$ равносильно: $c = c - a$. c будет равно 3
*=	Умножает правый операнд с левым и присваивает результат левому операнду.	$c = 5$ $a = 2$ $c *= a$ равносильно: $c = c * a$. c будет равно 10
/=	Делит левый операнд на правый и присваивает результат левому операнду.	$c = 10$ $a = 2$ $c /= a$ равносильно: $c = c / a$. c будет равно 5
%=	Делит по модулю операнды и присваивает результат левому.	$c = 5$ $a = 2$ $c \% = a$ равносильно: $c = c \% a$. c будет равно 1
**=	Возводит в левый операнд в степень правого и присваивает результат левому операнду.	$c = 3$ $a = 2$ $c ** = a$ равносильно: $c = c ** a$. c будет равно 9
//=	Производит целочисленное деление левого операнда на правый и присваивает результат левому операнду.	$c = 11$ $a = 2$ $c //= a$ равносильно: $c = c // a$. c будет равно 5

Побитовые операторы

Побитовые операторы перед выполнением операций переводят значения чисел в двоичный формат. Предположим, что у нас есть два числа $a = 60$ и $b = 13$. В двоичном формате они будут иметь следующий вид: $a = 00111100$, $b = 00001101$.

оператор	описание	примеры
$\&$	Побитовое логическое И.	$(a \& b)$ возвращает 12 (00001100 в двоичном представлении)
$ $	Побитовое логическое ИЛИ.	$(a b)$ возвращает 61 (00111101)
\wedge	Побитовое исключающее ИЛИ.	$(a \wedge b)$ возвращает 49 (00110001)
\sim	Унарный оператор. Побитовое логическое НЕ.	$(\sim a)$ возвращает -61 (11000011), двоичное число записано в дополнительном коде (eng. Two's complement).
\ll	Побитовый сдвиг влево. Биты левого операнда сдвигаются влево на кол-во знаков, указанное в правом операнде.	$a \ll 2$ возвращает 240 (11110000)
\gg	Побитовый сдвиг вправо. Биты левого операнда сдвигаются вправо на кол-во знаков, указанное в правом операнде.	$a \gg 2$ возвращает 15 (00001111)

Логические операторы

оператор	описание	примеры
and	Логическое И.	согласно таблице истинности логического оператора
or	Логическое ИЛИ.	согласно таблице истинности логического оператора
not	Логическое НЕ.	согласно таблице истинности логического оператора

Операторы членства

Используются для проверки на наличие элемента в составных типах данных (строки, списки, кортежи, словари). Подробнее будут рассмотрены при работе с данными соответствующих типов.

оператор	описание	примеры
in	Возвращает истину, если элемент присутствует в последовательности, иначе возвращает ложь.	"cad" in "cadillac" возвращает True. 1 in [2,3,1,6] возвращает True.
not in	Возвращает истину если элемента нет в последовательности.	"cad" in "cadillac" возвращает False. 1 in [2,3,1,6] возвращает False.

Операторы тождественности

Операторы сравнивают уникальные идентификаторы (id) объектов (id соответствуют адресу объекта в памяти компьютера). Возможны случаи, когда объекты с одинаковым значением содержимого имеют разные id.

оператор	описание	примеры
is	Возвращает истину, если id операндов совпадают.	<pre>x = y = [1, 2, 3] z = [1, 2, 3] print (x == y) возвращает True print (x == z) возвращает True print (x is y) возвращает True print (x is z) возвращает False print (x is not y) возвращает False print (x is not z) возвращает True</pre>
is not	Возвращает истину, если id операндов не совпадают.	

Приоритет операторов

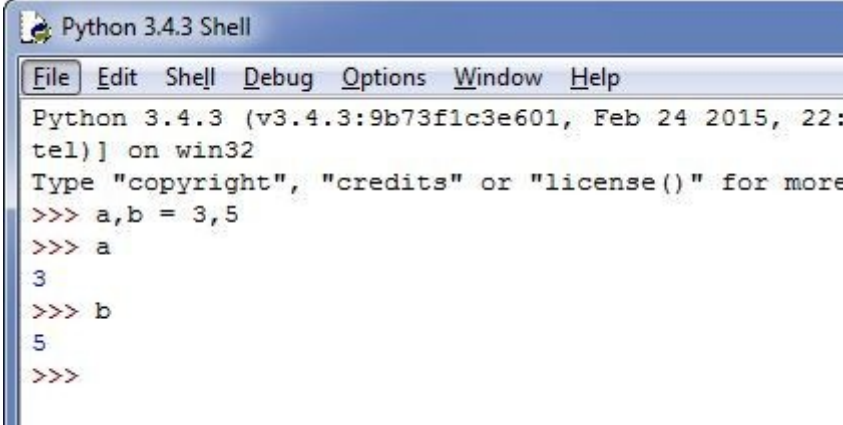
Операторы Python имеют следующие приоритеты выполнения (“;” разделяет операторы):

1	**
2	~ ; + ; - (унарные операторы)
3	* ; / ; % ; //
4	+ ; - (бинарные операторы)
5	>> ; <<
6	&
7	^ ;
8	<= ; < ; > ; >=
9	== ; !=
10	= ; %= ; /= ; //= ; -= ; += ; *= ; **=
11	is ; is not
12	in ; not in
13	not ; or ; and

6.3 Переменные и типы данных

Переменная — это поименованная область памяти, к которой можно обратиться по имени для чтения или записи значения в ходе выполнения программы. Для того, чтобы присвоить переменной новое значение, используются операторы присваивания.

Возможно присваивание значений нескольким переменным с использованием одного оператора присваивания, например:

A screenshot of a Python 3.4.3 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the following code and output:

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:
tel) on win32
Type "copyright", "credits" or "license()" for more
>>> a,b = 3,5
>>> a
3
>>> b
5
>>>
```

Python не требует явного объявления переменных — это значит, что переменная объявляется в любом месте программы, в момент первого присваивания какого-либо значения. В этот же момент происходит определения типа данных переменной.

Python имеет 5 стандартных типов данных:

- Числа (Numbers)
- Строка (String)
- Список (List)
- Кортеж (Tuple)
- Словарь (Dictionary)

Числовой тип данных

Предназначен для хранения числовых значений. Это неизменяемый тип данных, что означает, что изменение значения числового типа данных приведет к созданию нового объекта в памяти (и удалению старого).

Python поддерживает 4 различных числовых типа:

- `int` — целое число,
- `long` — длинное целое число,
- `float` — число с плавающей точкой,
- `complex` — комплексное число.

В наших работах мы будем, в основном, использовать типы данных `int` и `float`.

Тип данных строка

Под строками в Python подразумевается набор символов между кавычками. Для выделения строк можно использовать пары одинарных либо двойных кавычек.

Более подробно о числовых и строковых данных, а также о других типах данных, мы поговорим в следующих работах.

6.4 Встроенные функции Python

Помимо операторов и переменных при работе без подключения модулей можно использовать встроенные функции языка Python. Функция - это фрагмент программного кода (подпрограмма), к которому можно обратиться из другого места программы. Большинство функций принимают на вход аргумент (или аргументы) и возвращают (англ. return) результат действий над ними или с их использованием.

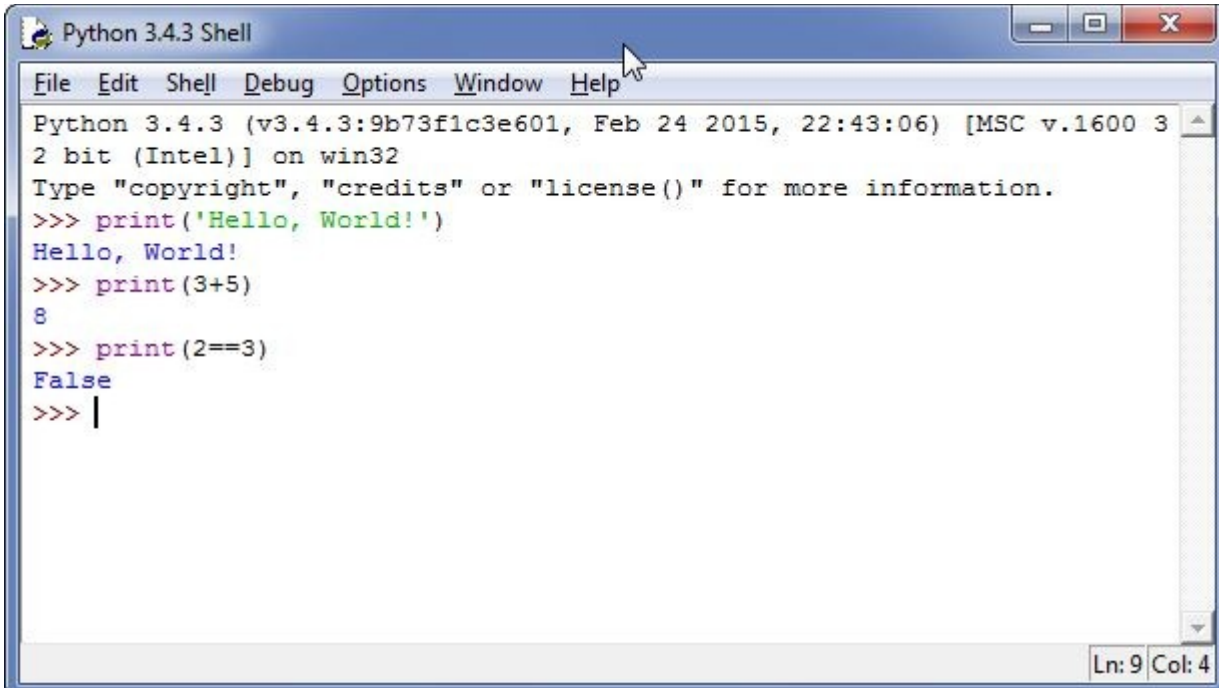
Язык Python версии 3.5.0 содержит 68 встроенных функций. По мере необходимости мы будем рассматривать некоторые из них. Рассмотрим несколько универсальных функций, которые могут быть полезны при написании как простых программ, так и сложных проектов. Для более полного понимания работы функций рекомендуем использовать официальную документацию языка [4] и информацию в сети Интернет.

print

```
print(*objects, sep=' ', end='\n')
```

Функция выводит на экран объекты (objects). Необязательные аргументы sep – разделитель объектов, end – окончание вывода.

Например:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 3
2 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print('Hello, World!')
Hello, World!
>>> print(3+5)
8
>>> print(2==3)
False
>>> |
```

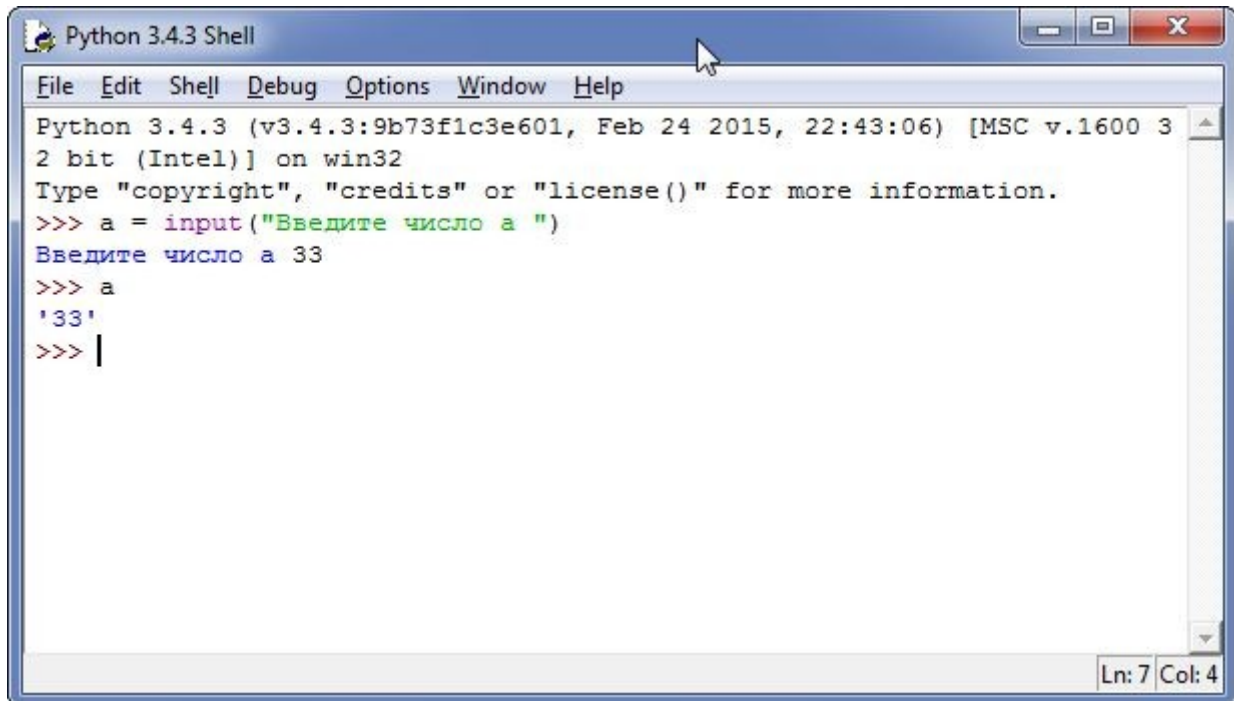
Ln: 9 Col: 4

input

```
var = input([string])
```

Функция выводит на экран приглашение `string` и присваивает переменной `var` строку, введенную пользователем. Ввод завершается нажатием клавиши Enter.

Например:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 3
2 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a = input("Введите число a ")
Введите число a 33
>>> a
'33'
>>> |
```

Обратите внимание: в данном примере «33» в памяти компьютера — это строка, а не число. Для работы с числами необходимо изменить тип данных после их чтения с клавиатуры. Изменение типов данных мы рассмотрим позже.

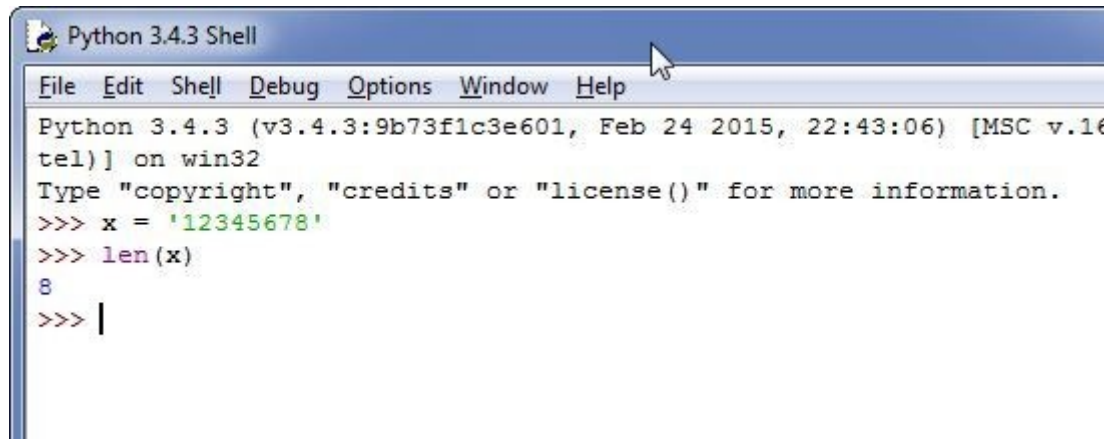
Возможно использование функции без строки-приглашения. В этом случае интерфейс будет не дружелюбен пользователю, но при автоматизированной проверке программ часто требуется именно такой способ ввода.

len

`len(s)`

Функция возвращает длину объекта `s`. В качестве объекта могут выступать строка, список, список, кортеж и другие.

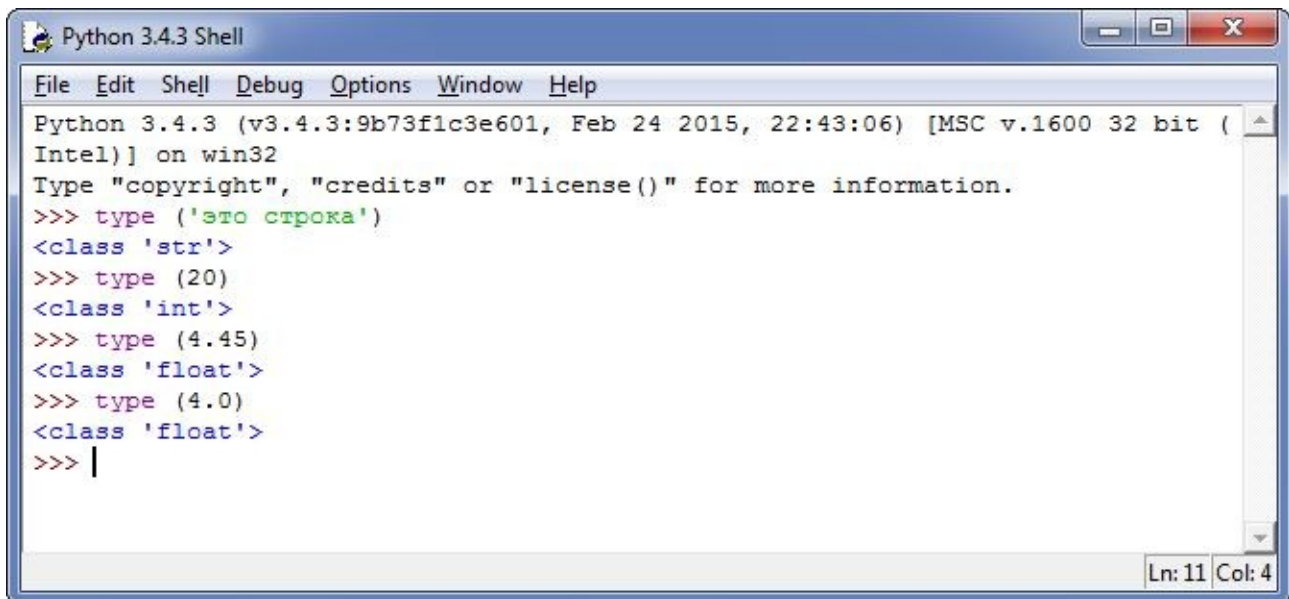
Например:

A screenshot of a Python 3.4.3 Shell window. The title bar reads "Python 3.4.3 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following content:

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.16000  
tel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> x = '12345678'  
>>> len(x)  
8  
>>> |
```

6.5 Определение и преобразование типов данных

Для определения типа данных используется встроенная функция **type**. Например:



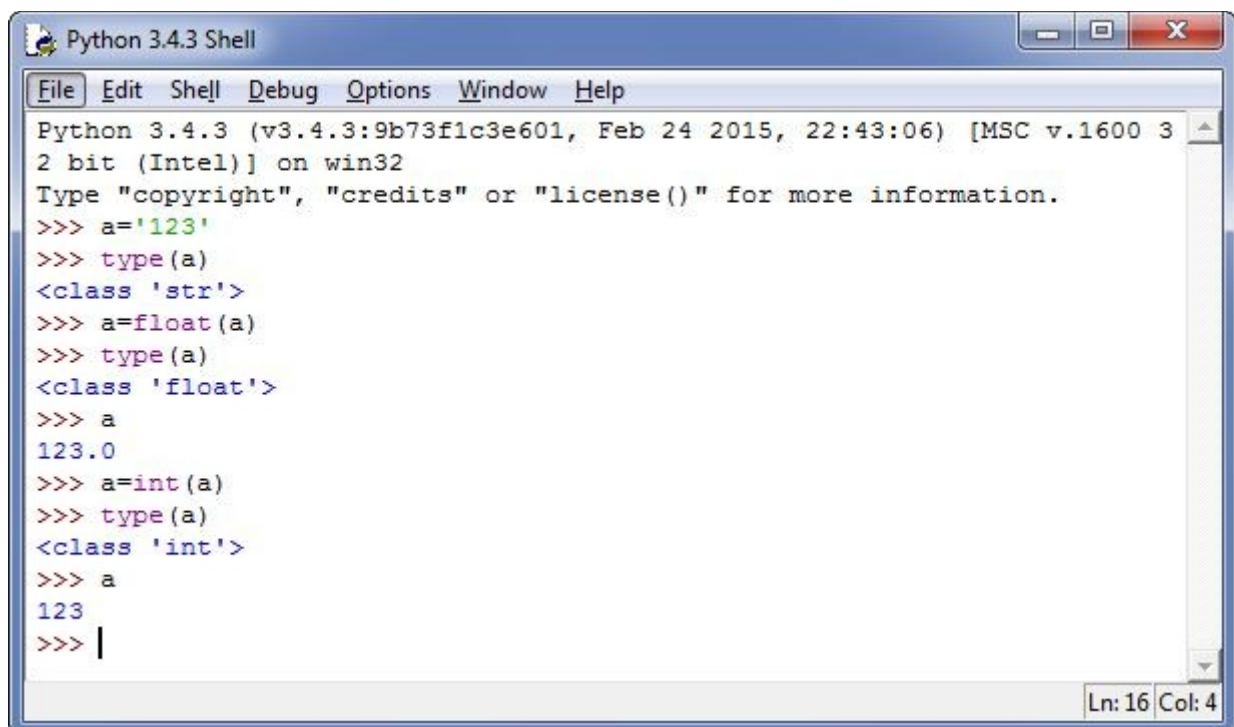
```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> type ('это строка')
<class 'str'>
>>> type (20)
<class 'int'>
>>> type (4.45)
<class 'float'>
>>> type (4.0)
<class 'float'>
>>> |
```

Обратите внимание: целое число 4.0 воспринимается языком как тип данных float.

Для преобразования между основными типами данных используются следующие встроенные функции:

- int – для преобразования в целое число,
- float – для преобразования в число с плавающей точкой,
- str – для преобразования в строку.

Например:

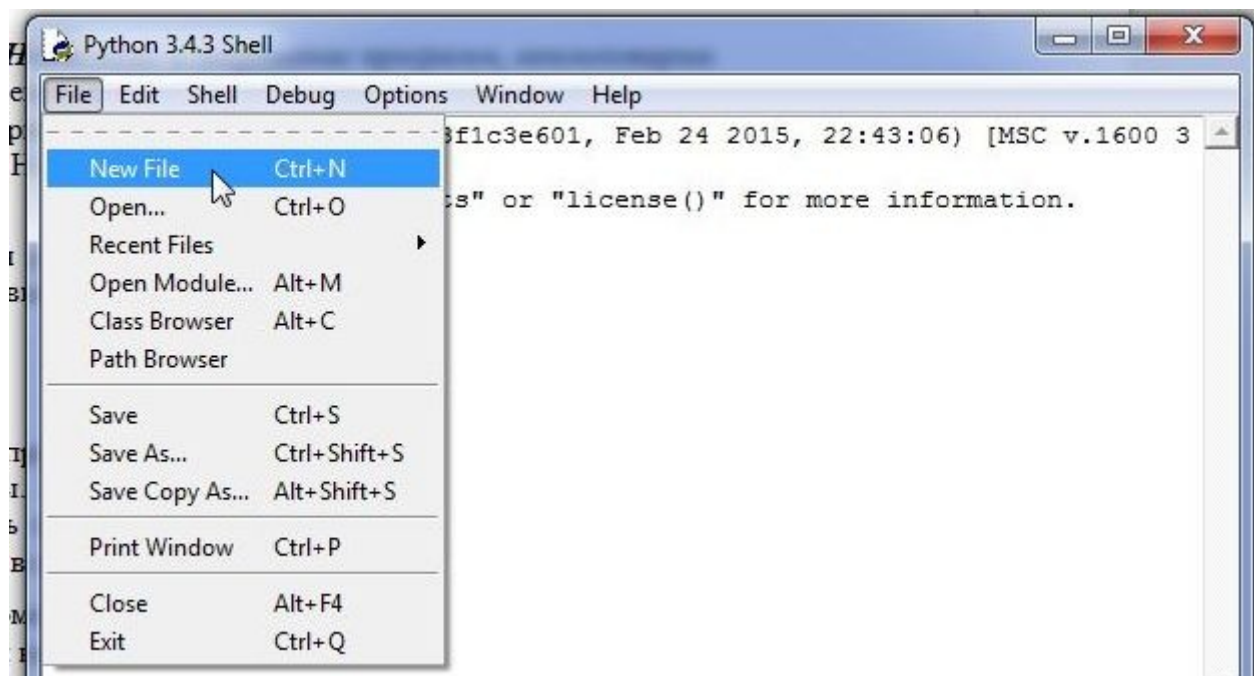


```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a='123'
>>> type(a)
<class 'str'>
>>> a=float(a)
>>> type(a)
<class 'float'>
>>> a
123.0
>>> a=int(a)
>>> type(a)
<class 'int'>
>>> a
123
>>> |
```

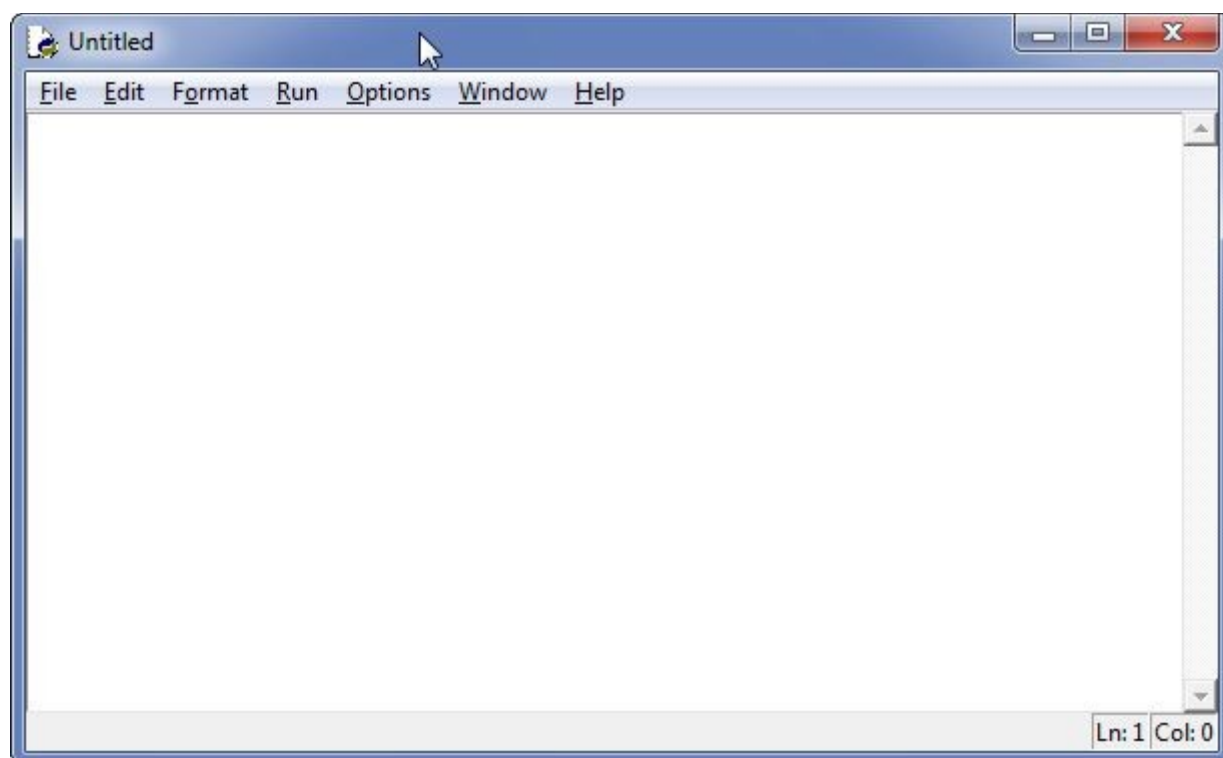

6.6 Написание и сохранение программ, комментарии

Интерактивный режим оболочки IDLE позволяет быстро выполнить несколько команд. Это удобно для проверки работы какой-то функции, несложных операций, использования встроенных функций. Но для решения более серьезных задач необходимо написать текст программы.

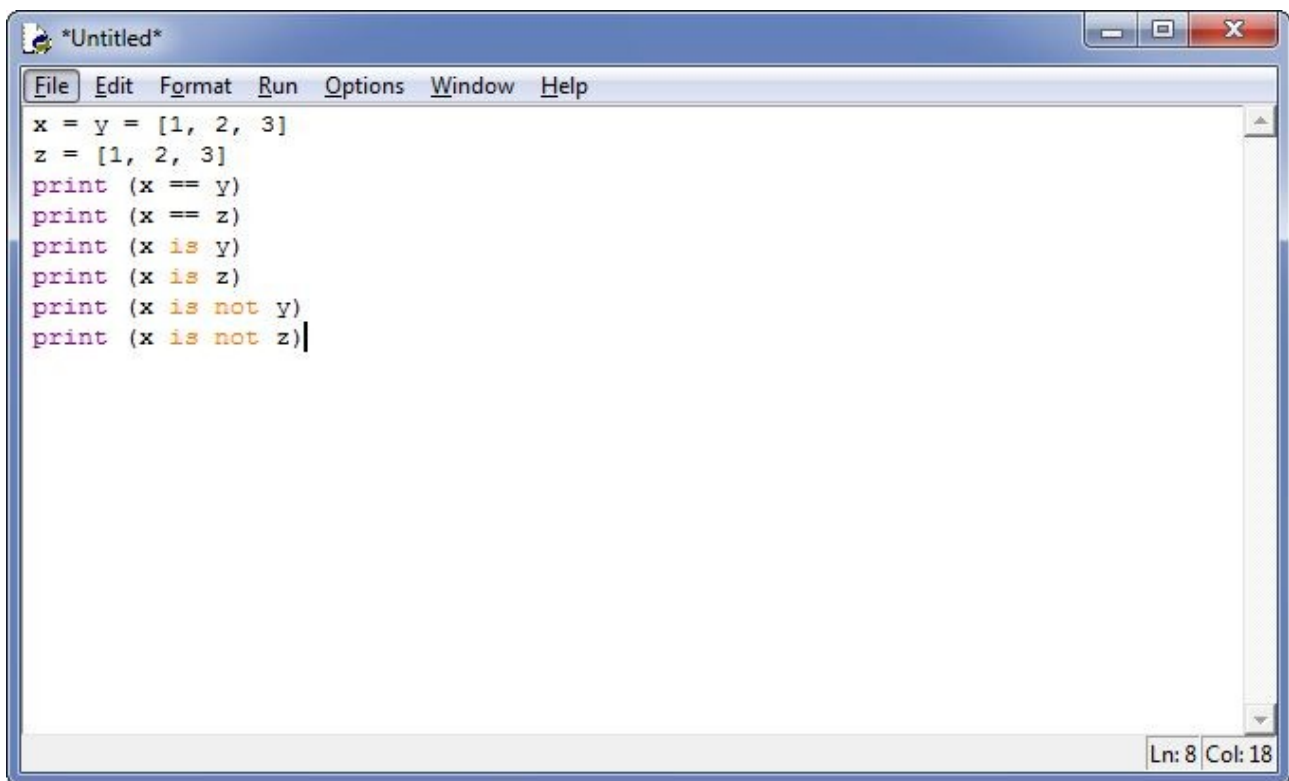
В среде IDLE для работы с текстом программ используется встроенный редактор. Для его вызова достаточно выполнить команду **File – New File**:



После этого откроется окно редактора:



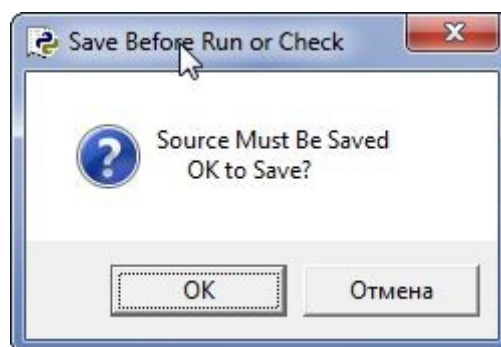
В окне редактора можно набрать (или скопировать и вставить) текст программы. Например, команды, которые мы использовали для проверки работы операторов тождественности:



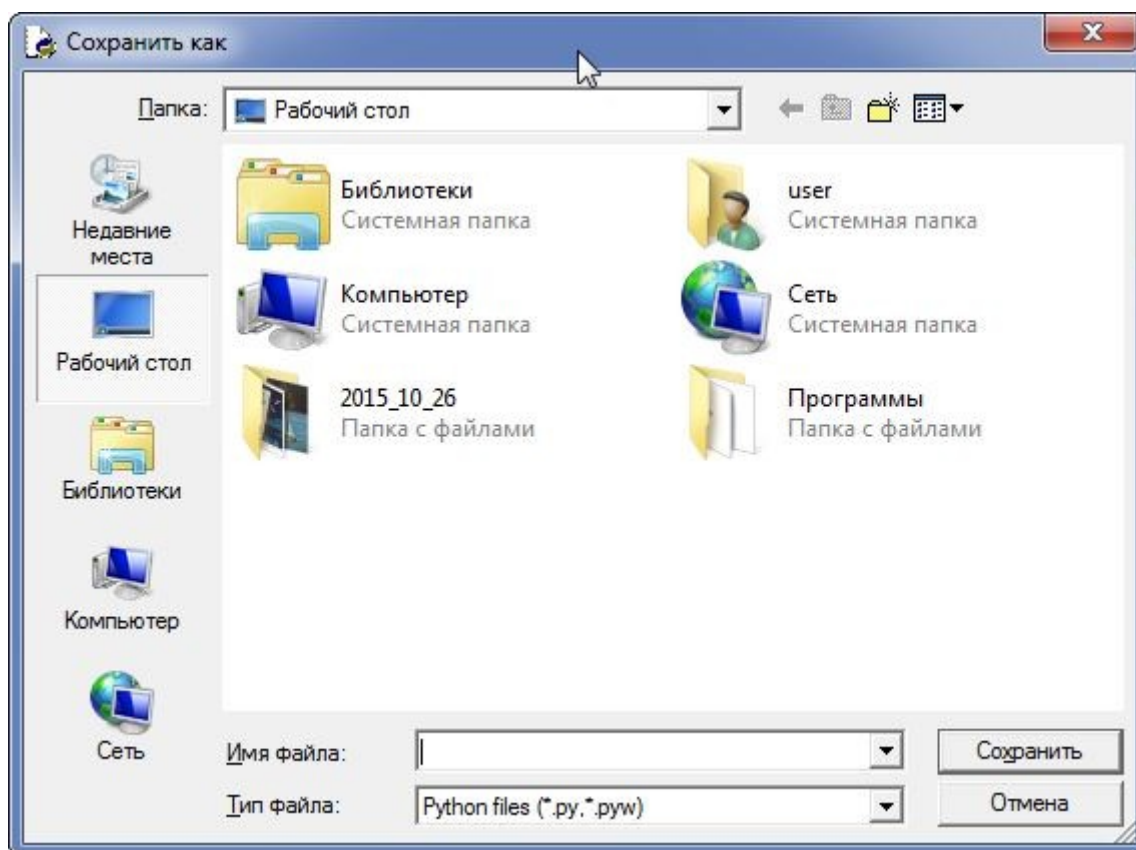
```
x = y = [1, 2, 3]
z = [1, 2, 3]
print (x == y)
print (x == z)
print (x is y)
print (x is z)
print (x is not y)
print (x is not z)|
```

Ln: 8 Col: 18

Для запуска программы используйте клавишу F5. При запуске несохраненной программы будет выведено соответствующее сообщение. Программу можно выполнить только после того, как её текст будет сохранен:

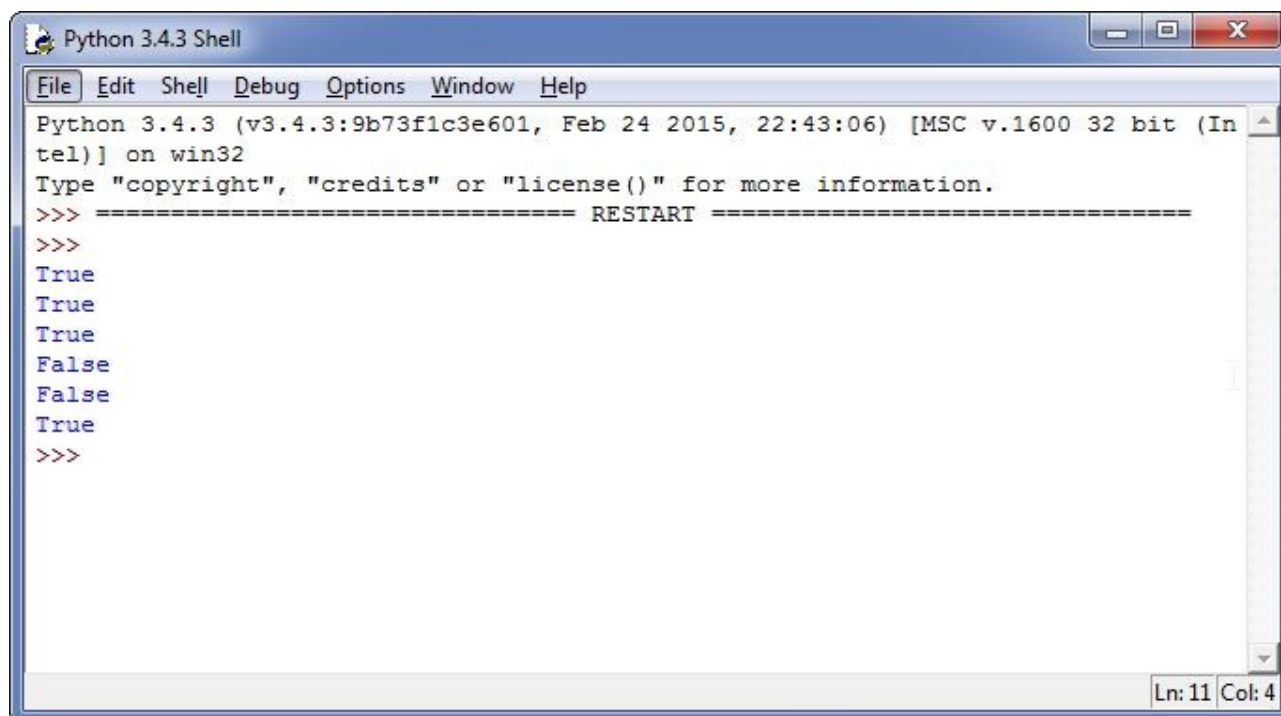


Нажимаем ОК для сохранения. Т.к. ранее мы файл не сохраняли, откроется диалоговое окно сохранения:

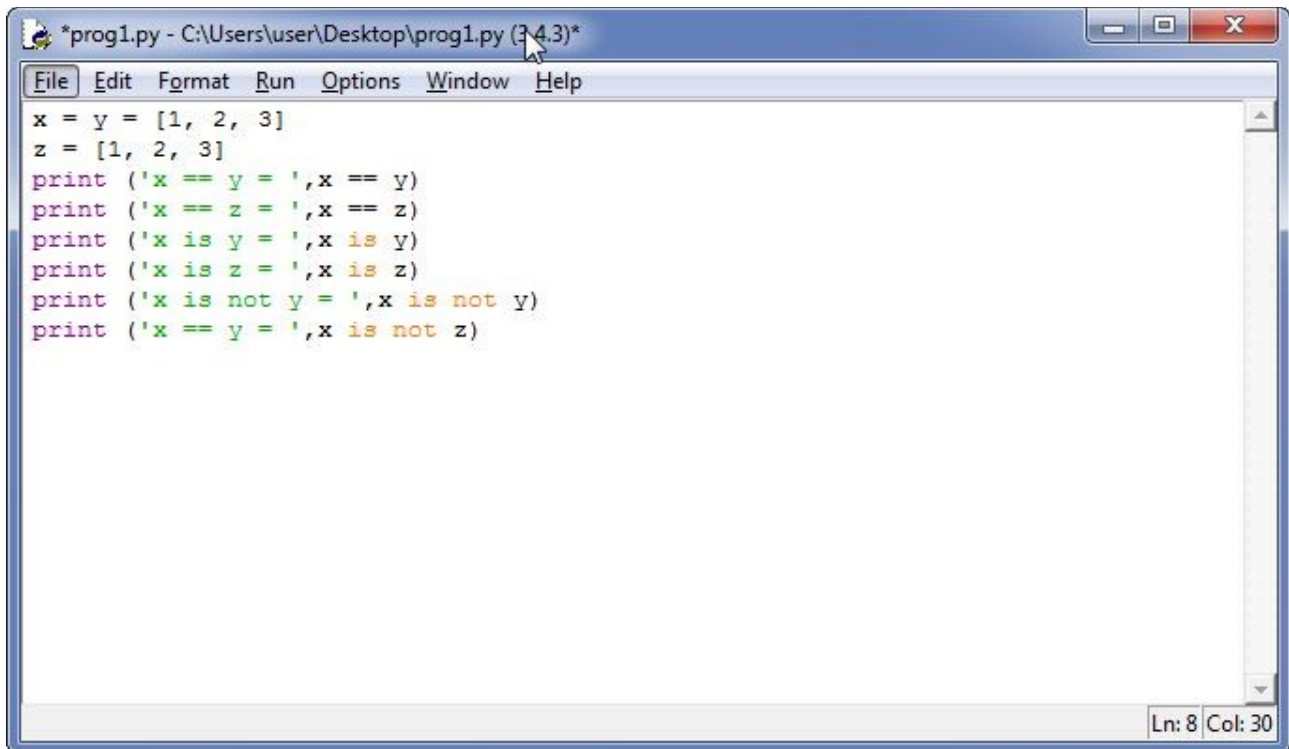


Необходимо выбрать каталог, указать имя файла и сохранить программу.

После этого в командном окне выводится результат выполнения программы:

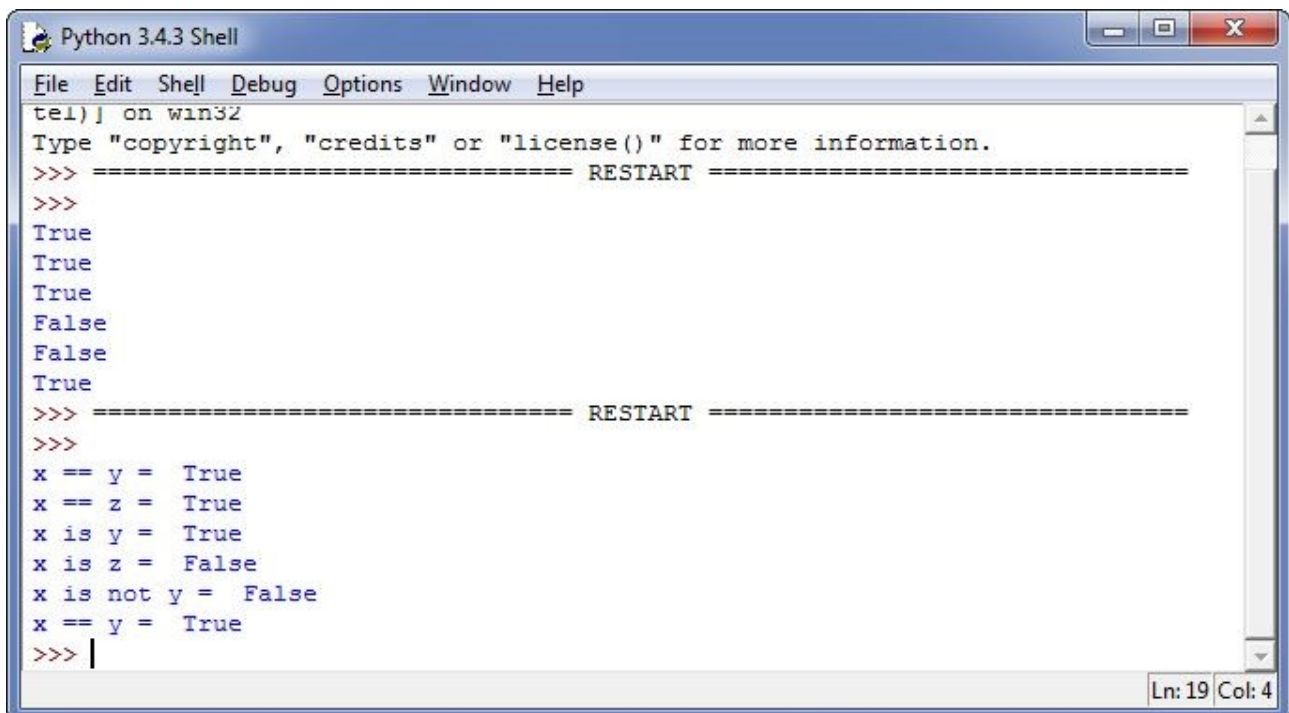


Мы написали и сохранили нашу программу. Она работает. Но пользователь не поймет, что выведено на экран. Добавим в текст программы пояснения для пользователя. Для этого выведем вместе с результатами логических операций сами операции в виде строк, заключенных в кавычки (сочетания клавиш **Ctrl + C**, **Ctrl + V** могут не работать при использовании русской раскладки клавиатуры, также для копирования можно использовать команды **Ctrl + Ins**, для вставки **Shift + Ins**):



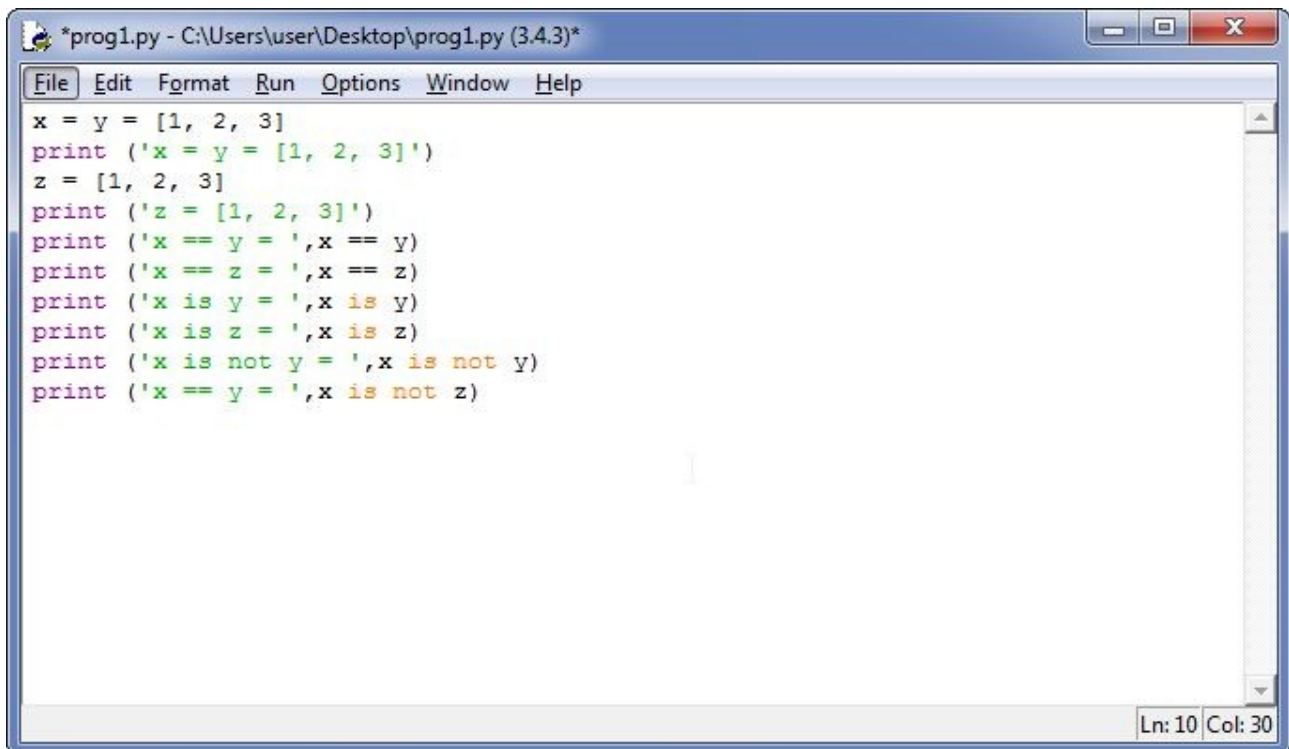
```
*prog1.py - C:\Users\user\Desktop\prog1.py (3.4.3)*
File Edit Format Run Options Window Help
x = y = [1, 2, 3]
z = [1, 2, 3]
print ('x == y = ', x == y)
print ('x == z = ', x == z)
print ('x is y = ', x is y)
print ('x is z = ', x is z)
print ('x is not y = ', x is not y)
print ('x == y = ', x is not z)
Ln: 8 Col: 30
```

Выполним программу и посмотрим на результат:



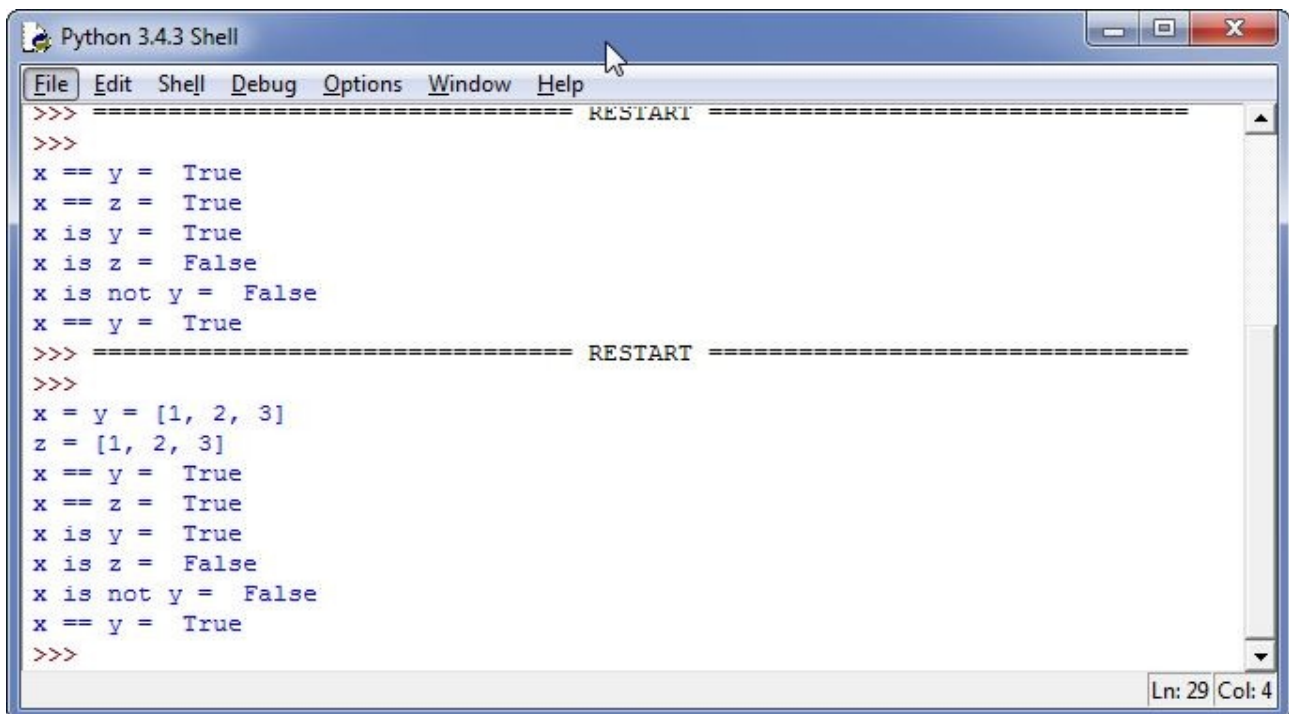
```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
tel) on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
True
True
True
False
False
True
>>> ===== RESTART =====
>>>
x == y = True
x == z = True
x is y = True
x is z = False
x is not y = False
x == y = True
>>> |
Ln: 19 Col: 4
```

Сейчас пользователь видит результаты логических операций, но не видит использованных исходных данных. Вывод программы по-прежнему остался малоинформативным. Исправим это — добавим в программу вывод на экран порядка присваивания значений переменным, который важен для нашего примера:



```
*prog1.py - C:\Users\user\Desktop\prog1.py (3.4.3)*
File Edit Format Run Options Window Help
x = y = [1, 2, 3]
print ('x = y = [1, 2, 3]')
z = [1, 2, 3]
print ('z = [1, 2, 3]')
print ('x == y = ', x == y)
print ('x == z = ', x == z)
print ('x is y = ', x is y)
print ('x is z = ', x is z)
print ('x is not y = ', x is not y)
print ('x == y = ', x is not z)
Ln: 10 Col: 30
```

И результат запуска программы:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
>>> ===== RESTART =====
>>>
x == y = True
x == z = True
x is y = True
x is z = False
x is not y = False
x == y = True
>>> ===== RESTART =====
>>>
x = y = [1, 2, 3]
z = [1, 2, 3]
x == y = True
x == z = True
x is y = True
x is z = False
x is not y = False
x == y = True
>>>
Ln: 29 Col: 4
```

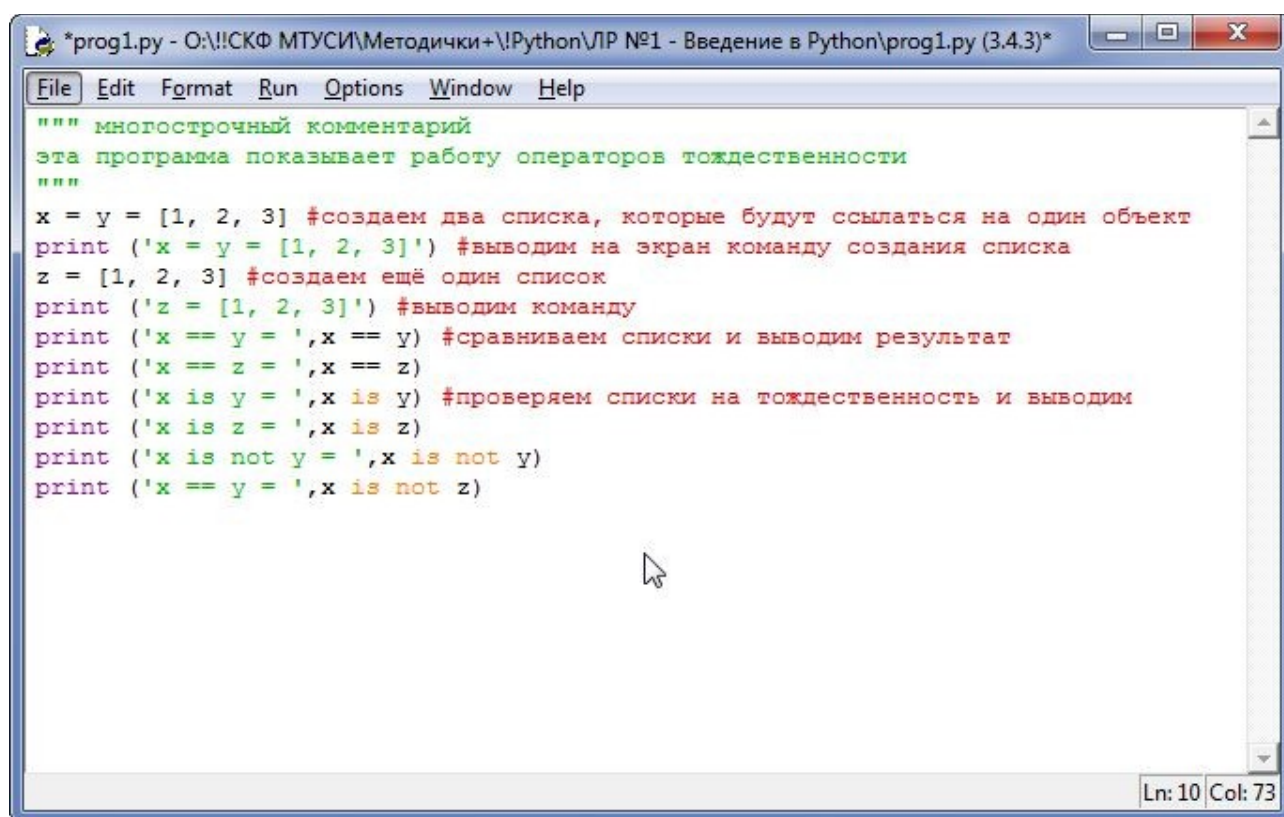
Теперь пользователь, запустивший программу, поймет, что именно выводится на экран. Но допустим, над файлом программы необходимо работать другому программисту. При наличии достаточной квалификации он сможет понять, что происходит в программе. Но чем сложнее и больше текст программы, тем сложнее понять, что происходит в отдельных её

фрагментах. Поэтому хорошим тоном при написании программ считается комментирование текста программы. Комментарии будут полезны в двух случаях: если над вашей программой будет работать кто-то другой, или если вы вернетесь к своему проекту через какое-то время, чтобы что-то в нем изменить или дополнить.

В языке Python можно использовать однострочные и многострочные комментарии. Однострочный комментарий начинается со знака «#» и может занимать как отдельную строку, так и находится на строке с фрагментом программы. Многострочный комментарий — это все, что заключено между тройными кавычками.

Ещё один случай, в котором используются комментарии — это отключение части кода во время отладки. Или, напротив, отключения кода, используемого при отладке, в итоговой версии программы.

Расставим в нашей программе комментарии, например:

A screenshot of a Python IDE window titled '*prog1.py - O:\!\СКФ МТУСИ\Методички+\Python\ЛР №1 - Введение в Python\prog1.py (3.4.3)*'. The window contains a Python script with several lines of code and comments. The comments are in Russian and explain the purpose of each line of code. The code uses list creation, assignment, and identity/equality checks. The IDE has a menu bar with File, Edit, Format, Run, Options, Window, and Help. The status bar at the bottom right shows 'Ln: 10 Col: 73'.

```
""" многострочный комментарий
эта программа показывает работу операторов тождественности
"""
x = y = [1, 2, 3] #создаем два списка, которые будут ссылаться на один объект
print ('x = y = [1, 2, 3]') #выводим на экран команду создания списка
z = [1, 2, 3] #создаем ещё один список
print ('z = [1, 2, 3]') #выводим команду
print ('x == y = ', x == y) #сравниваем списки и выводим результат
print ('x == z = ', x == z)
print ('x is y = ', x is y) #проверяем списки на тождественность и выводим
print ('x is z = ', x is z)
print ('x is not y = ', x is not y)
print ('x == y = ', x is not z)
```

Выполнение программы не поменяется, но работать с программой будет удобнее.

6.7 Пример программы

Написать программу, которая выполняет следующие действия:

- обеспечивает ввод переменных x и y с клавиатуры,
- производит вычисления по формуле:

$$c = \frac{(y - \pi) \cdot \cos \frac{\pi}{4}}{1 + \ln |1 - y|} + \sqrt[3]{\frac{xy}{1 + 0,3x}}$$

- выводит на экран полученный результат.

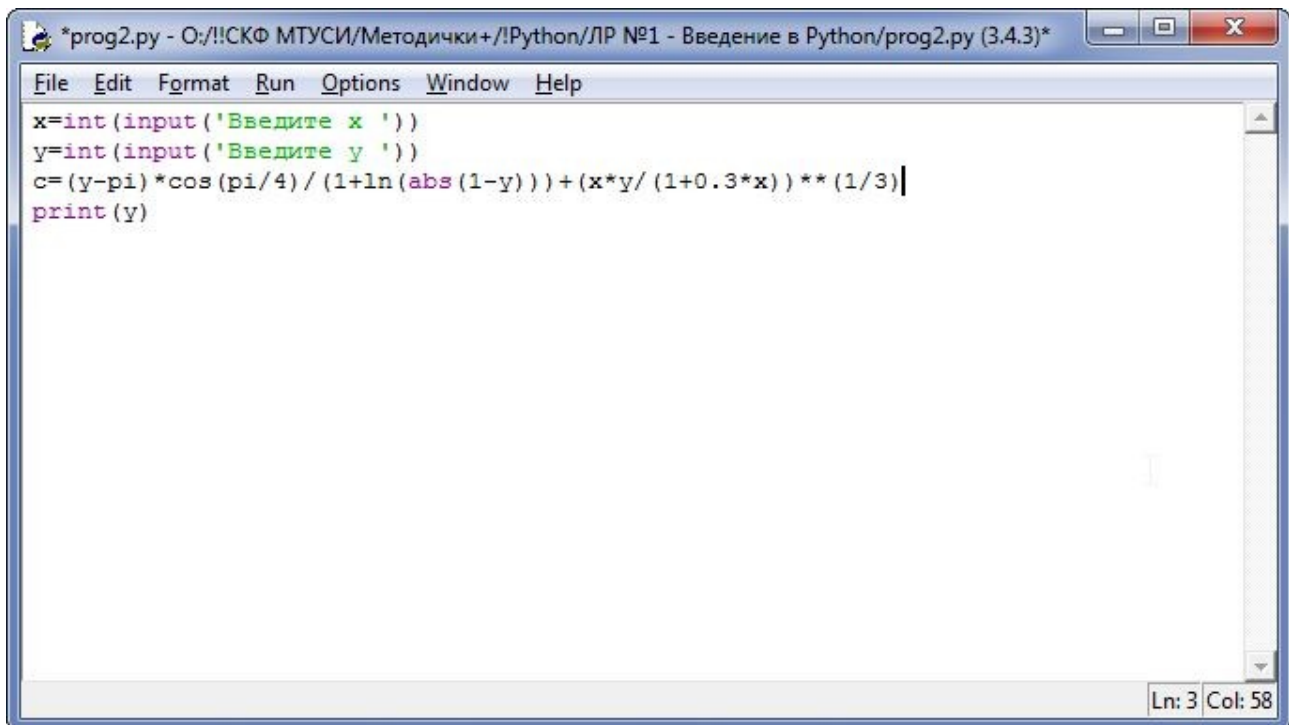
Набор данных для проверки:

x	y	c
12,737	10,56	4,6432

В программе необходимо использовать:

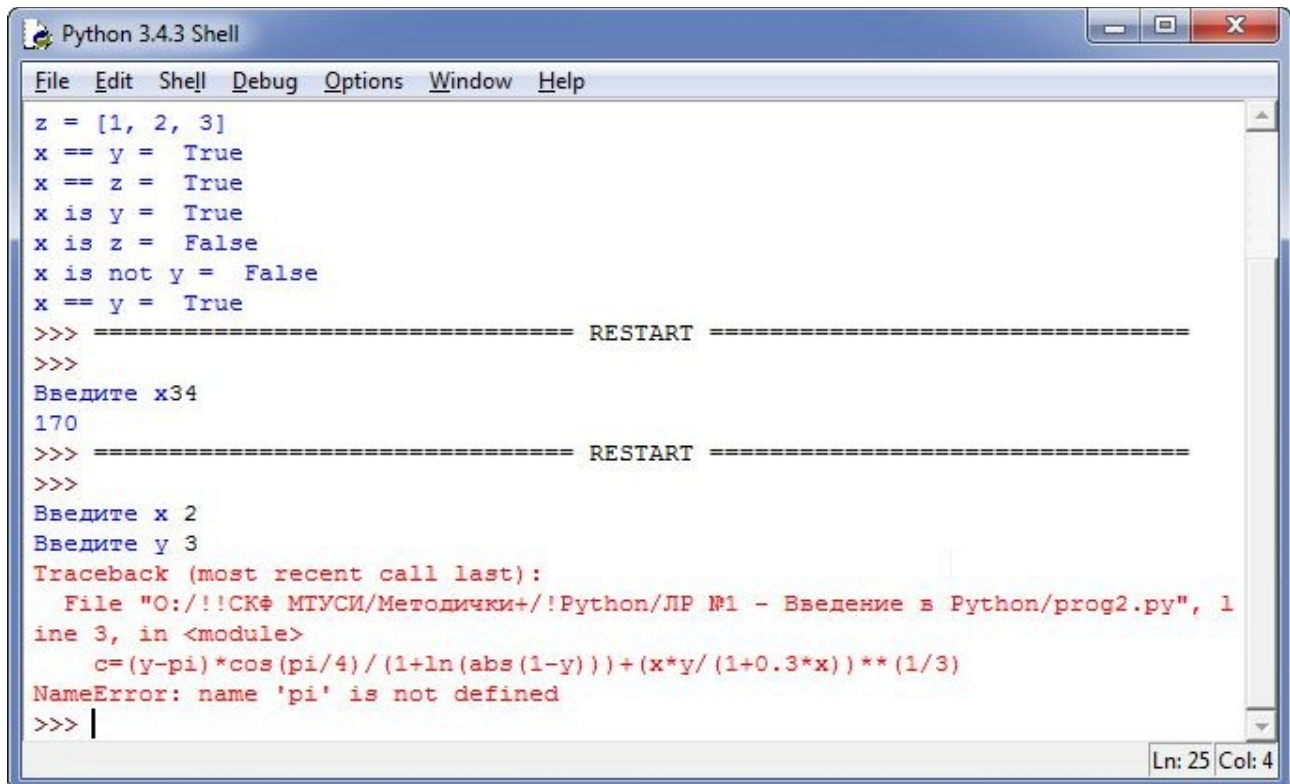
- оператор присваивания,
- арифметические операторы,
- встроенные функции ввода и вывода,
- встроенную функцию преобразования типа данных.

Напишем программу для выполнения необходимых действий. Пока что без комментариев:



```
*prog2.py - O:/!!СКФ МТУСИ/Методички+!Python/ЛР №1 - Введение в Python/prog2.py (3.4.3)*
File Edit Format Run Options Window Help
x=int(input('Введите x '))
y=int(input('Введите y '))
c=(y-pi)*cos(pi/4)/(1+ln(abs(1-y)))+(x*y/(1+0.3*x))**(1/3)
print(y)
Ln: 3 Col: 58
```

Запустим написанную программу, введем произвольные значения переменных:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
z = [1, 2, 3]
x == y = True
x == z = True
x is y = True
x is z = False
x is not y = False
x == y = True
>>> ===== RESTART =====
>>>
Введите x34
170
>>> ===== RESTART =====
>>>
Введите x 2
Введите y 3
Traceback (most recent call last):
  File "O::\!\СК\МТУСИ/Методички+!\Python/ЛР №1 - Введение в Python/prog2.py", line 3, in <module>
    c=(y-pi)*cos(pi/4)/(1+ln(abs(1-y)))+(x*y/(1+0.3*x))**(1/3)
NameError: name 'pi' is not defined
>>> |
```

Ln: 25 Col: 4

В результате выполнения программы было получено сообщение об ошибке. Не удивительно, ведь Python не знает, что такое «pi», которое мы написали. Так же он отреагирует и на слова cos, ln и abs, которые были взяты по аналогии с языком программирования Pascal. Для выполнения этих и других операций в Python необходимо подключить стандартный модуль math.

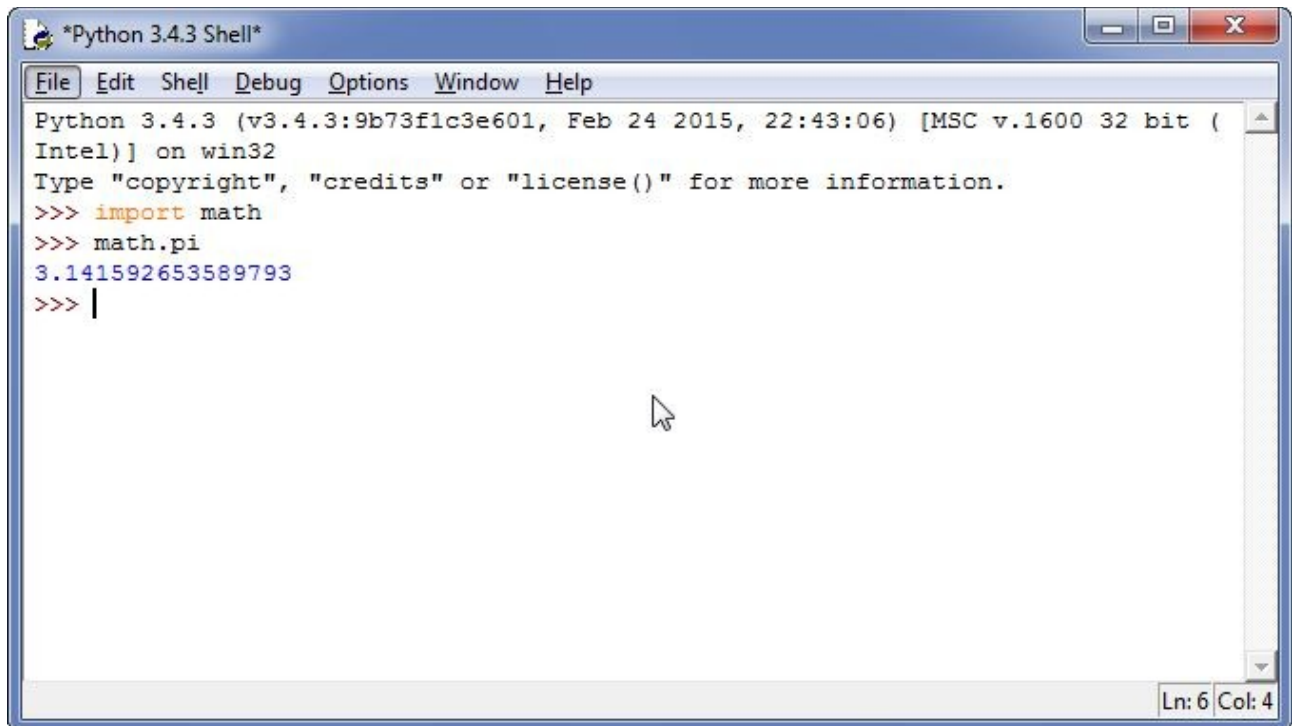
6.8 Подключение модулей. Модуль *math*

Возможности языка Python можно расширить с помощью подключения модулей. Модули могут быть встроенными и внешними. В Python существует более 200 встроенных модулей, каждый из которых позволяет решать группу задач, например, работать с датой и временем или работать с zip-архивами.

Подключение модуля выполняется с помощью инструкции `import`. Для решения примера нам необходимо подключить модуль `math`. Код подключения будет выглядеть следующим образом:

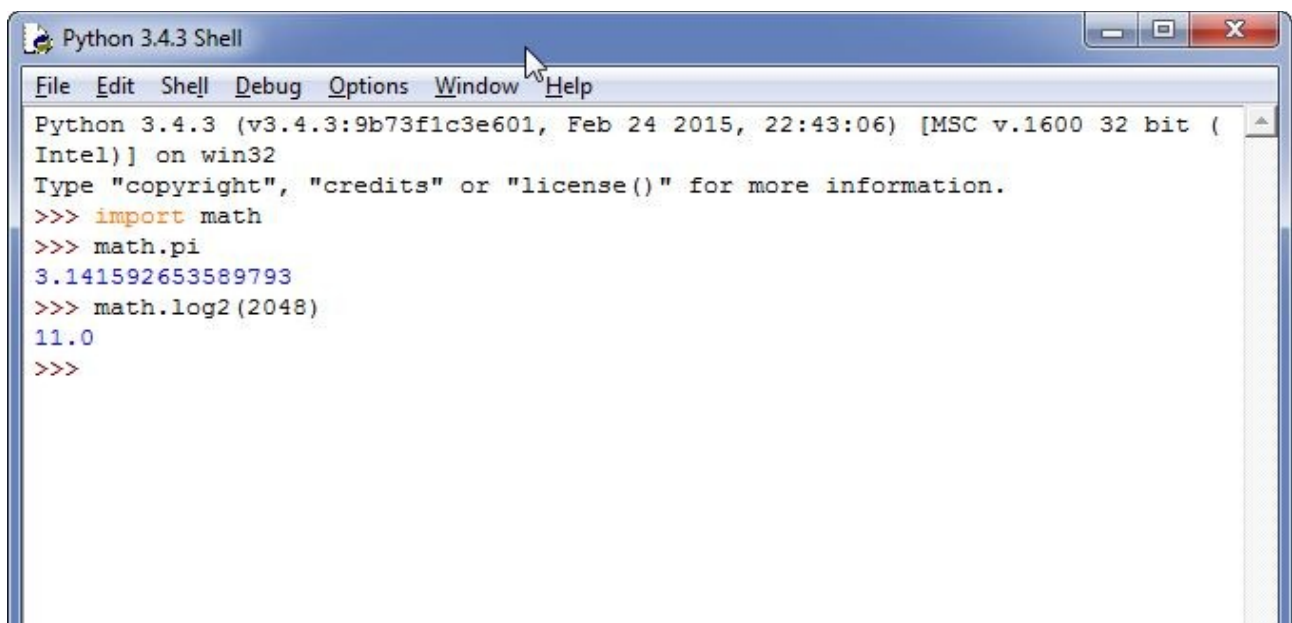
```
import math
```

После подключения модуля его название становится переменной, через которую можно получить доступ к атрибутам модуля, например:



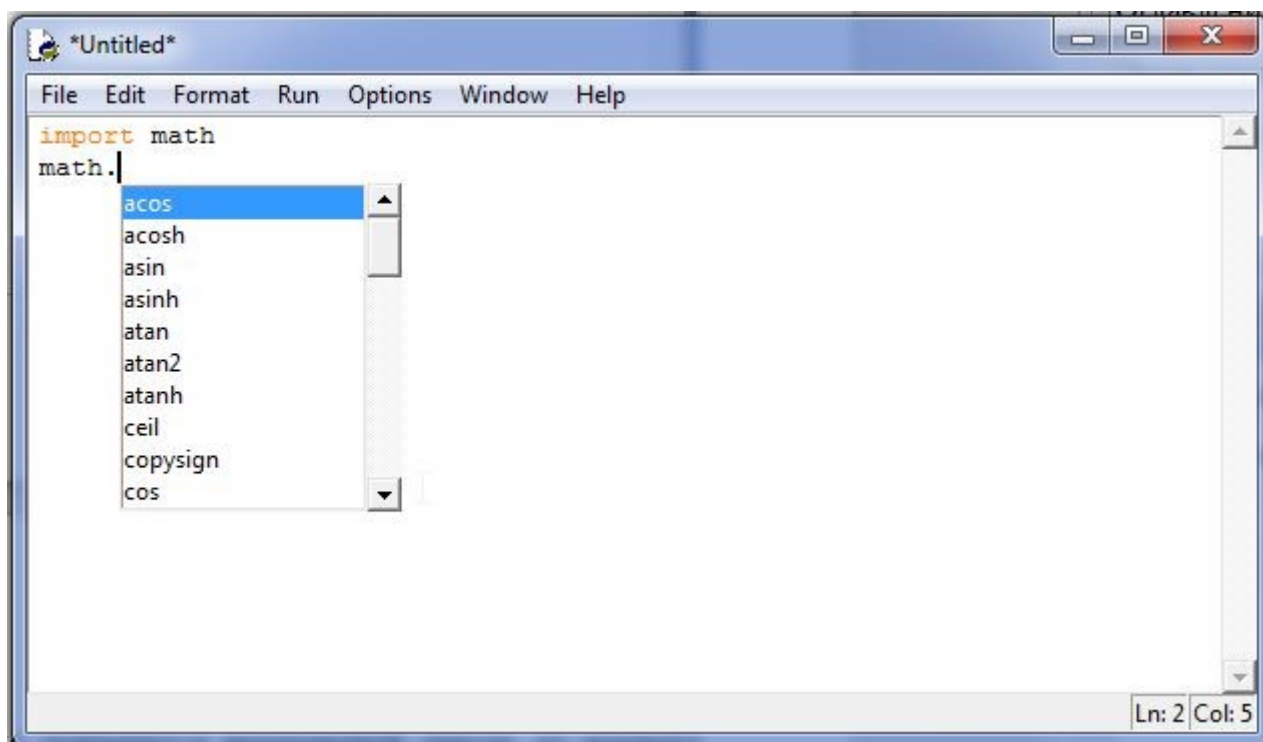
```
*Python 3.4.3 Shell*
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import math
>>> math.pi
3.141592653589793
>>> |
```

Также можно использовать встроенные в модуль функции:

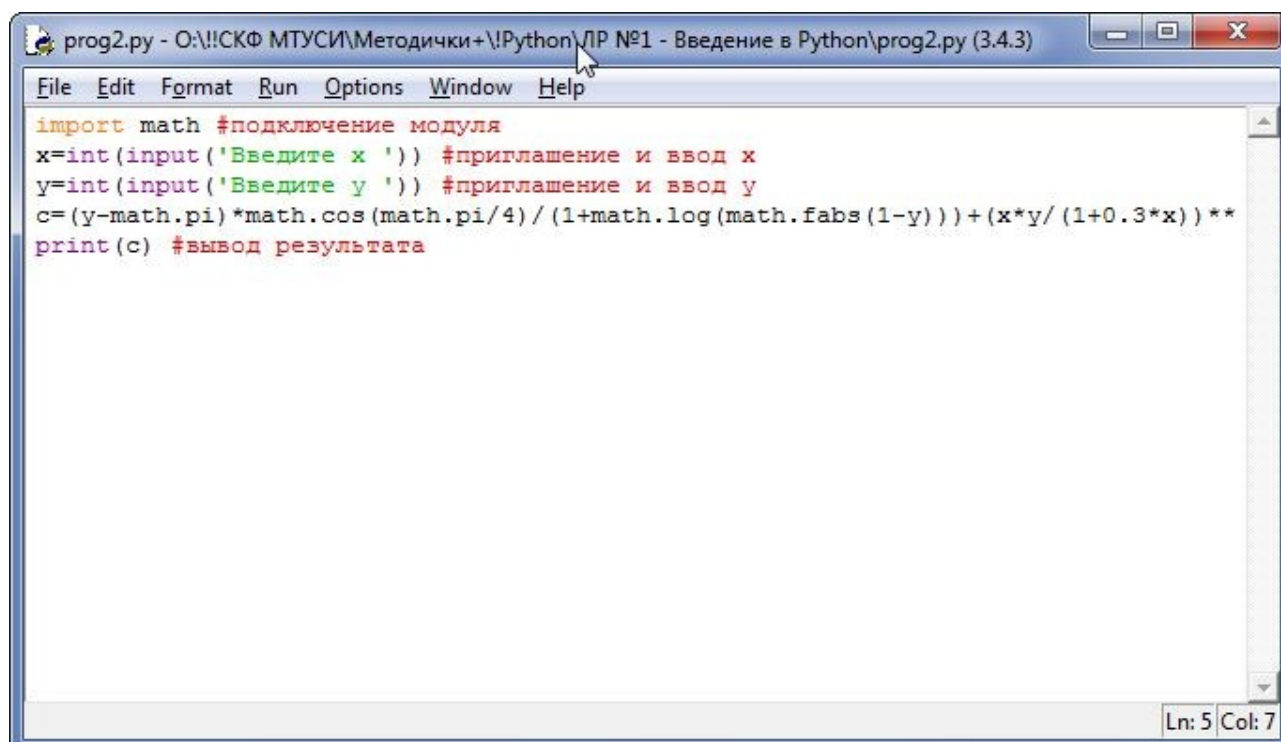


```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import math
>>> math.pi
3.141592653589793
>>> math.log2(2048)
11.0
>>>
```

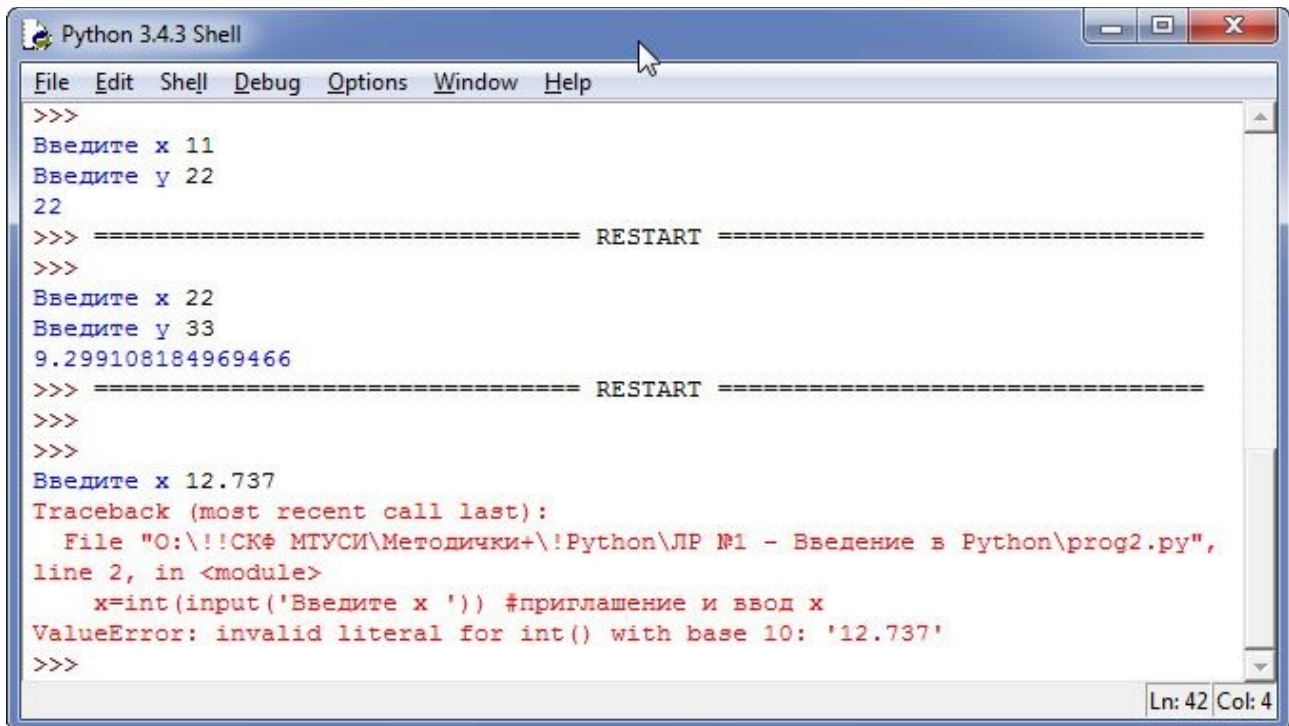
Для просмотра все доступных атрибутов модуля необходимо после импорта ввести его имя, поставить точку и нажать клавишу **Tab**. Эта функция работает как в интерактивном режиме, так и в режиме редактора:



Итак, добавим в нашу программу использование модуля math:

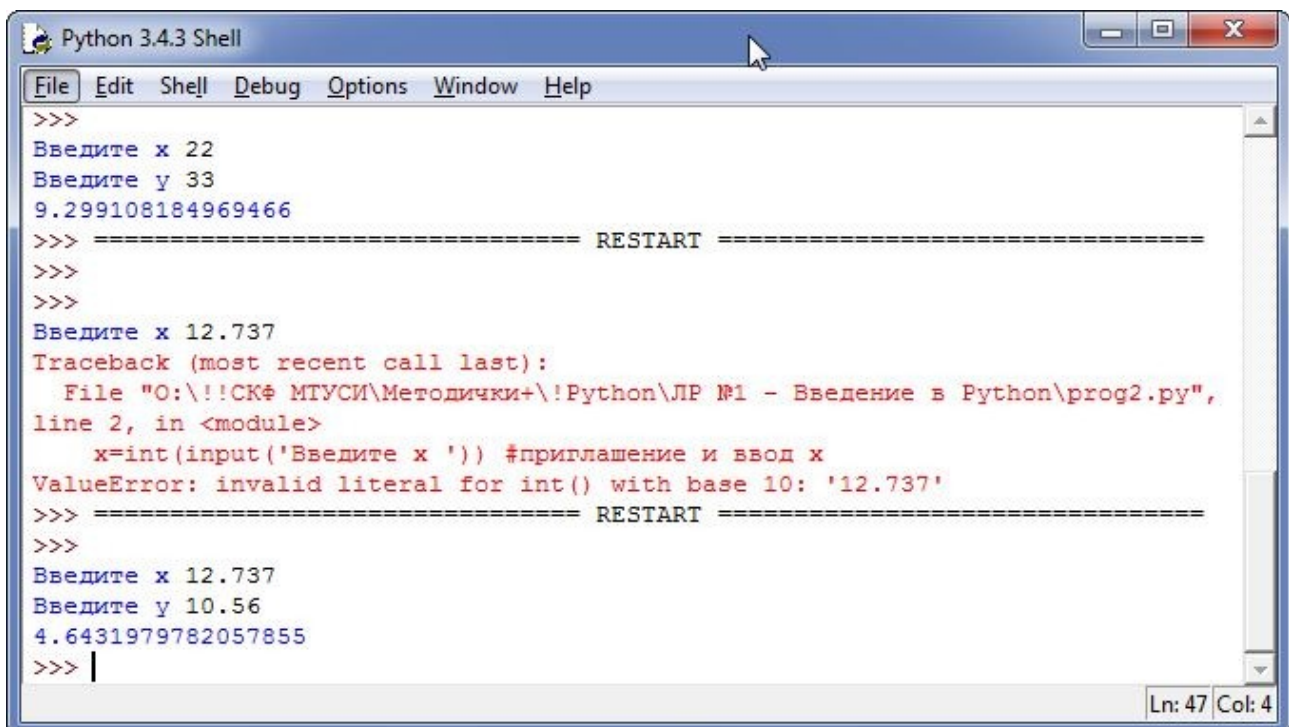


Выполним нашу программу, используя данные для проверки:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
>>>
Введите x 11
Введите y 22
22
>>> ===== RESTART =====
>>>
Введите x 22
Введите y 33
9.299108184969466
>>> ===== RESTART =====
>>>
>>>
Введите x 12.737
Traceback (most recent call last):
  File "O:\!!СКФ МТУСИ\Методички+\\Python\ЛР №1 - Введение в Python\prog2.py",
    line 2, in <module>
      x=int(input('Введите x ')) #приглашение и ввод x
ValueError: invalid literal for int() with base 10: '12.737'
>>>
Ln: 42 Col: 4
```

Мы снова столкнулись с ошибкой! Как и в прошлый раз, Python указывает нам на ошибку. Мы пытаемся использовать встроенную функцию `int` для преобразования числа с плавающей точкой. Заменяем её на `float` и запустим программу ещё раз:



```
Python 3.4.3 Shell
File Edit Shell Debug Options Window Help
>>>
Введите x 22
Введите y 33
9.299108184969466
>>> ===== RESTART =====
>>>
>>>
Введите x 12.737
Traceback (most recent call last):
  File "O:\!!СКФ МТУСИ\Методички+\\Python\ЛР №1 - Введение в Python\prog2.py",
    line 2, in <module>
      x=int(input('Введите x ')) #приглашение и ввод x
ValueError: invalid literal for int() with base 10: '12.737'
>>> ===== RESTART =====
>>>
>>>
Введите x 12.737
Введите y 10.56
4.6431979782057855
>>> |
Ln: 47 Col: 4
```

На этот раз мы получили результат, совпадающий с данными, предложенными для проверки. Задача решена.

6.9 Самостоятельная работа

1. Используя функцию `random.randint(0,5)` сгенерировать случайное число t от 0 до 5. Сделать скриншот запуска программы.

2. Написать на языке Python программы для вычисления функции. Предусмотреть ввод и вывод данных.

Необходимо решить 5 примеров. Номера вариантов заданий n выбираются по правилу:

$$n_i = 6 \cdot (i-1) + t,$$

где t – результат работы генератора в предыдущем задании,

i – порядковый номер выбираемого задания (1,2,3,4,5).

Т.е. выполняется условие $n \bmod 6 = t$.

В отчет по каждой программе включить:

- текст программы
- скриншот окна редактора с готовой программой,
- скриншот командного окна сразу после выполнения программы.

Варианты заданий:

Вариант	Формулы для вычислений	Данные для проверки		
		х	у	вывод
00	$c = \frac{(y - \pi) \cdot \cos \frac{\pi}{4}}{1 + \ln 1 - y } + \sqrt[3]{\frac{xy}{1 + 0,3x}}$	12,737	10,56	4,6432
01	$t = \cos \frac{\pi}{7} \cdot \frac{\sin^2(x - 8y)}{2,7(x - \pi)}$	3,59	17,53	0,7441
02	$d = \frac{(1 - e^{xy})^2}{0,7 \lg 1 - x^2 }$	1,674	-0,533	1,9456
03	$h = \frac{xy + \sin x}{ 1 - y \cdot \ln x}$	32,01	-0,4917	-2,9359
04	$c = \frac{(xy^2 - 1)^2}{2} \cdot (\cos^2 y - \sin x^2)$	2,123	-1,89	23,3509
05	$b = \sqrt[3]{\frac{x+y}{0,2x}} \cdot \sin(\operatorname{tg}^2 x)$	8,402	-0,2226	0,7486
06	$d = \frac{xe^{xy} + 8 \sin^2 x}{x(x - y)(3x + y)}$	1,002	-0,5321	1,6480
07	$z = \frac{\pi}{2} - \sqrt{2x} - \frac{x + y^2}{0,75 \operatorname{tg} x + y }$	12,003	-5,408	-173,9594

08	$d = \frac{xy^2 - \sqrt{x^2 - 2,5 \cdot 10^{-3} y}}{2 \sin xy}$	1,3802	-1,9	-3,6300
09	$f = 120 \cdot \frac{\lg(x + y)}{x - \frac{1}{0,45 \sin(x - 8 y)}}$	12,678	6,9	14,8449
10	$a = 0,8 \cdot 10^{-5} (e x^{-(y-1,2)} - yx)^3$	82,578	1,4517	-13,7820
11	$R = \sqrt[4]{\frac{2x + y}{0,1 x}} \cdot \cos(\sin^2 x)$	5,422	-0,1537	1,7678
12	$s = \frac{x^2 y + \ln x}{ 1 - y \cdot \cos x}$	25,01	-0,5736	-227,6709
13	$t = \cos \frac{\pi}{5} \cdot \frac{\sin^2(3x - 5y)}{3,5(x + \pi)}$	4,72	21,35	0,0292
14	$u = \frac{x e^{xy} + 4 \cos^2 x}{x(x - y)(2x + y)}$	1,472	-0,4342	0,1158
15	$a = 3 \cdot \lg(x + y) + x - \frac{1}{0,68 \sin(x - 7 y)}$	12,345	7,5	18,5604
16	$b = \frac{(3 - e^{xy})^2}{0,5 \cdot \lg(1 - x)}$	-1,834	-0,533	0,5175
17	$c = \frac{xy + \sqrt{x^2 - 3,4 \cdot 10^{-3} y}}{2 \cos xy^2}$	1,385	-1,2	0,3352
18	$d = 0,75 \cdot 10^{-6} (x e^{-x(y-1,5)} - yx)^2$	75,412	1,1745	8,93E+18
19	$z = \frac{\pi}{3} - \sqrt{3} x + \frac{x^2 + y}{0,55 \cdot \operatorname{tg} x + y }$	15,207	-4,415	79,4329
20	$f = \sin(\operatorname{tg}^2 x) \cdot \sqrt{\frac{2x + y}{0,3 x}}$	10,315	-0,2114	2,5661
21	$z = \frac{2x + y^2}{0,82 \operatorname{tg} x + y } + \frac{\pi}{4} - \ln x + y $	11,485	-3,231	-18,5521
22	$h = \frac{\sin^2(x - 7 y)}{2,5(x - \pi)} \cdot \sin \frac{\pi}{4}$	4,58	16,34	0,0047
23	$P = \frac{(1 - y) \cdot \ln x}{ xy + \sin x } + \sqrt[3]{\frac{x + y}{0,1 x}}$	7,428	-0,543	3,0914

24	$R = x - \frac{1}{0,27 \sin(x - 4y)} + 21 \cdot \lg (x + y) $	8,804	-0,4915	31,9183
25	$t = \frac{x(x - y)(4x + y)}{2,7(x - \pi)} + \sqrt[3]{\frac{0,2x}{\sin y}}$	20,356	0,523	713,8126
26	$S = (y \cdot e^{-x(y - 0,8)} + xy^2)^3 \cdot 0,9 \cdot 10^{-4}$	75,447	1,2328	135,6826
27	$a = \frac{\sqrt{(x^3 - 3,4 \cdot 10^{-3}y)} + x^2y}{3 \cos xy}$	12,615	7,5	441,8435
28	$b = \frac{0,72 \cdot \operatorname{tg} x + y }{x^2 + y} - \sqrt{2x} + \frac{\pi}{2}$	11,872	1,4517	-3,2972
29	$d = \frac{(1 - e^{2xy^2})^2}{0,65 \sin x \cdot \ln 1 - y^2 }$	-32,01	1,457	-23,7209

7 Контрольные вопросы

1. Что такое язык программирования высокого уровня?
2. Что такое программа?
3. Что такое алгоритм?
4. Что такое оператор?

8 Информационные источники

1. Википедия. <http://ru.wikipedia.org/>
2. ГОСТ 19781-90 Обеспечение систем обработки информации программное. Термины и определения.
3. Гражданский кодекс РФ. Глава 70. Статья 1261.
4. <https://docs.python.org/3/library/functions.html>