

Практическая работа 2 Математические операции в Python

Цель работы: познакомиться с основными математическими операциями в Python

Язык Python, благодаря наличию огромного количества библиотек для решения разного рода вычислительных задач, сегодня является конкурентом таким пакетам как Matlab и Octave. Запущенный в интерактивном режиме, он, фактически, превращается в мощный калькулятор. В этом уроке речь пойдет об арифметических операциях, доступных в данном языке. Арифметические операции изучим применительно к числам.

Если в качестве операндов некоторого арифметического выражения используются только целые числа, то результат тоже будет целое число. Исключением является операция деления, результатом которой является вещественное число. При совместном использовании целочисленных и вещественных переменных, результат будет вещественным.

В этом уроке речь пойдет об арифметических операциях, доступных в данном языке.

Если в качестве операндов некоторого арифметического выражения используются только целые числа, то результат тоже будет целое число. Исключением является операция деления, результатом которой является вещественное число. При совместном использовании целочисленных и вещественных переменных, результат будет вещественным.

Целые числа (int)

Числа в Python 3 поддерживают набор самых обычных математических операций:

| | |
|---------|-----------|
| $x + y$ | Сложение |
| $x - y$ | Вычитание |
| $x * y$ | Умножение |
| x / y | Деление |

| | |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x // y</code> | Получение целой части от деления |
| <code>x % y</code> | Остаток от деления |
| <code>-x</code> | Смена знака числа |
| <code>abs(x)</code> | Модуль числа |
| <code>divmod(x, y)</code> | Пара (<code>x // y</code> , <code>x % y</code>) |
| <code>x ** y</code> | Возведение в степень |
| <code>pow(x, y[, z])</code> | <p><code>x</code> : Число, которое требуется возвести в степень.</p> <p><code>y</code> : Число, являющееся степенью, в которую нужно возвести первый аргумент. Если число отрицательное или одно из чисел "<code>x</code>" или "<code>y</code>" не целые, то аргумент "<code>z</code>" не принимается.</p> <p><code>z</code> : Число, на которое требуется произвести деление по модулю. Если число указано, ожидается, что "<code>x</code>" и "<code>y</code>" положительны и имеют тип <code>int</code>.</p> |

Пример применения вышеописанных операций над целыми числами

```

x = 5
y = 2
z = 3
x+y = 7
x-y = 3
x*y = 10
x/y = 2.5
x//y = 2
x%y = 1
-x = -5
abs(-x) = 5
divmod(x, y) = (2, 1)
x**y = 25
pow(x, y, z) = 1

```

Вещественные числа (float)

Вещественные числа поддерживают те же операции, что и целые. Однако (из-за представления чисел в компьютере) вещественные числа неточны, и это может привести к ошибкам.

Пример применения вышеописанных операций над вещественными числами

```

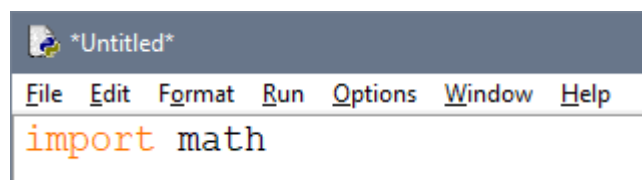
x = 5.5
y = 2.3
x+y = 7.8
x-y = 3.2
x*y = 12.649999999999999
x/y = 2.3913043478260874
x//y = 2.0
x%y = 0.90000000000000004
-x = -5.5
abs(-x) = 5.5
divmod(x,y) = (2.0, 0.90000000000000004)
x**y = 50.44686540422945

```

Библиотека (модуль) math

В стандартную поставку Python входит библиотека math, в которой содержится большое количество часто используемых математических функций.

Для работы с данным модулем его предварительно нужно импортировать.



Рассмотрим наиболее часто используемые функции модуля math

| | |
|----------------------------|-------------------------------------------------------------------------------------------------|
| math.ceil(x) | Возвращает ближайшее целое число большее, чем x |
| math.fabs(x) | Возвращает абсолютное значение числа x |
| math.factorial(x) | Вычисляет факториал x |
| math.floor(x) | Возвращает ближайшее целое число меньшее, чем x |
| math.exp(x) | Вычисляет e^x |
| math.log2(x) | Логарифм по основанию 2 |
| math.log10(x) | Логарифм по основанию 10 |
| math.log(x[, base]) | По умолчанию вычисляет логарифм по основанию e, дополнительно можно указать основание логарифма |
| math.pow(x, y) | Вычисляет значение x в степени y |
| math.sqrt(x) | Корень квадратный от x |

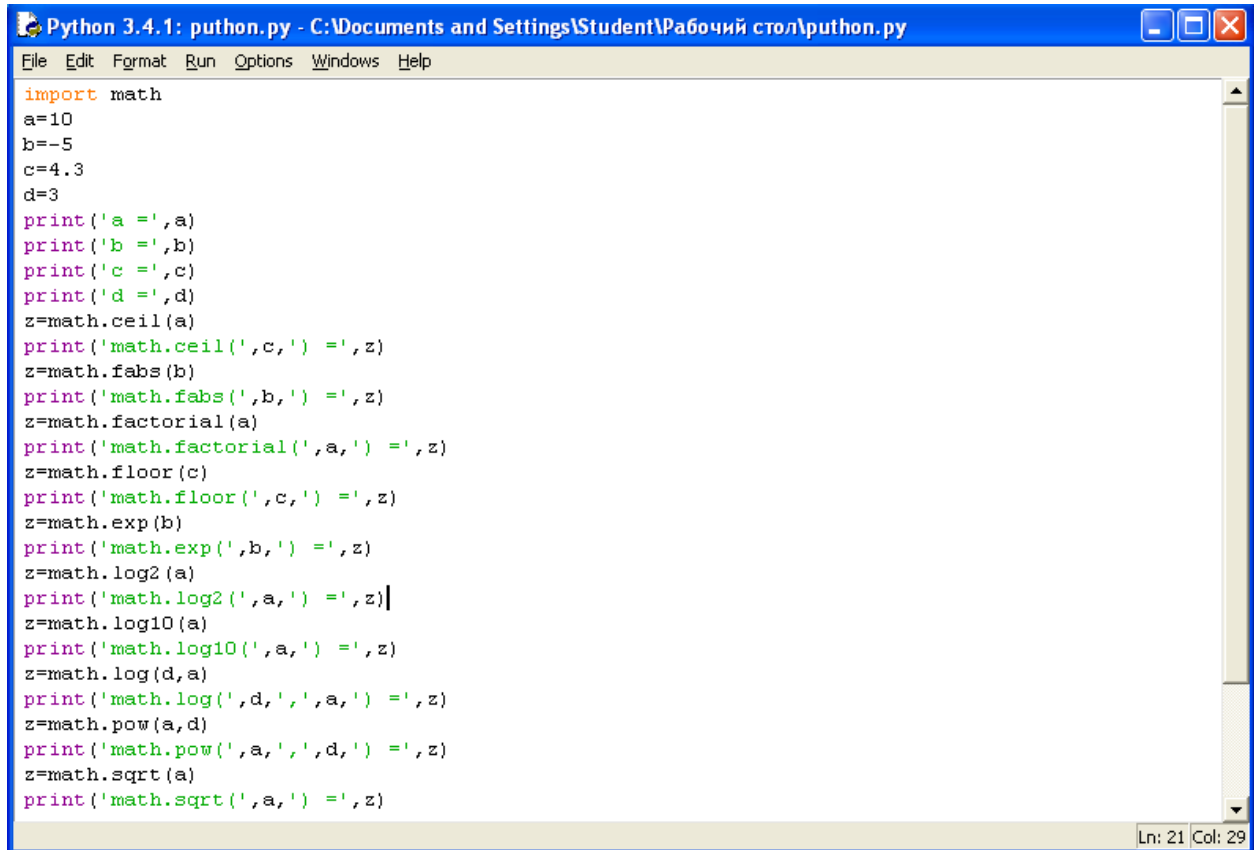
Пример применения вышеописанных функций над числами

В программе определены 4 переменные - a, b, c, d, каждая из которых является либо целым числом, либо вещественным, либо отрицательным.

Командой `print()` выводится значение каждой переменной на экран при выполнении программы.

В переменную `z` помещается результат выполнения функции модуля `math`.

Затем командой `print()` выводится сообщение в виде используемой функции и её аргумента и результат её выполнения.

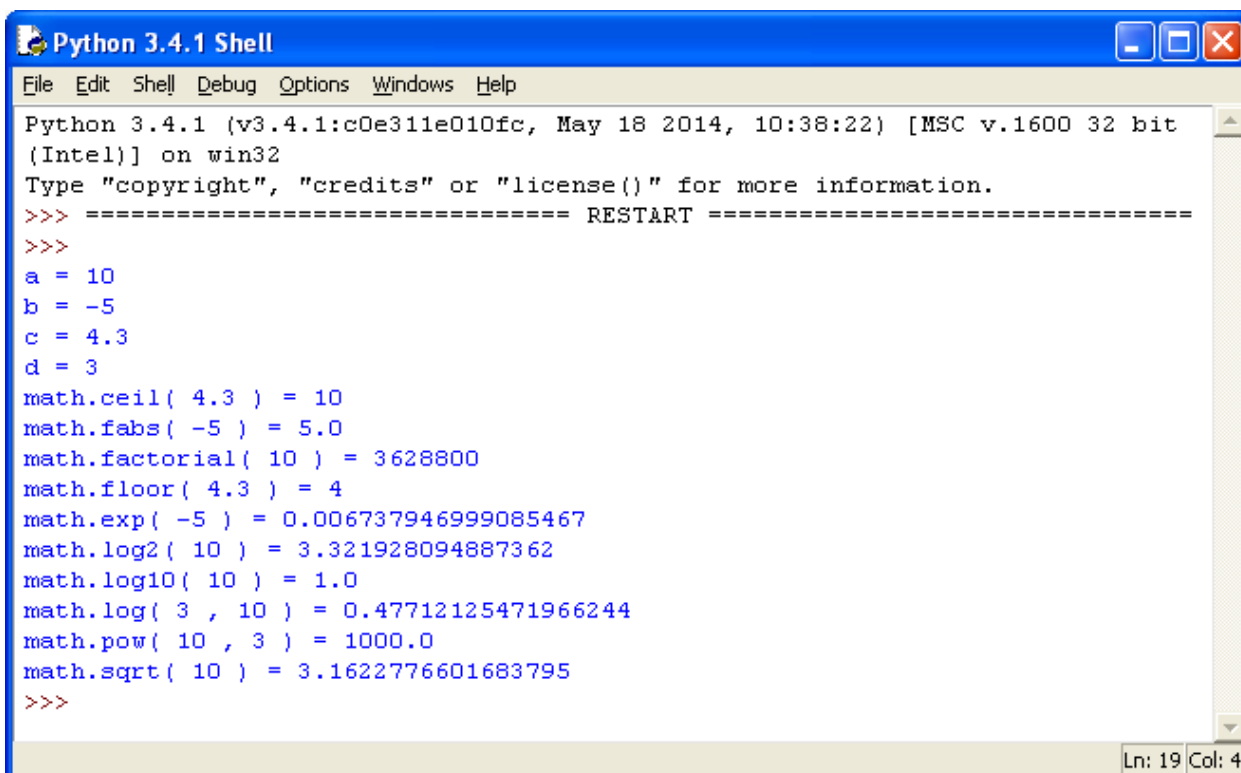


```
Python 3.4.1: puthon.py - C:\Documents and Settings\Student\Рабочий стол\puthon.py
File Edit Format Run Options Windows Help

import math
a=10
b=-5
c=4.3
d=3
print('a =',a)
print('b =',b)
print('c =',c)
print('d =',d)
z=math.ceil(a)
print('math.ceil(' ,c, ' ) =', z)
z=math.fabs(b)
print('math.fabs(' ,b, ' ) =', z)
z=math.factorial(a)
print('math.factorial(' ,a, ' ) =', z)
z=math.floor(c)
print('math.floor(' ,c, ' ) =', z)
z=math.exp(b)
print('math.exp(' ,b, ' ) =', z)
z=math.log2(a)
print('math.log2(' ,a, ' ) =', z)
z=math.log10(a)
print('math.log10(' ,a, ' ) =', z)
z=math.log(d,a)
print('math.log(' ,d, ' ,',a, ' ) =', z)
z=math.pow(a,d)
print('math.pow(' ,a, ' ,',d, ' ) =', z)
z=math.sqrt(a)
print('math.sqrt(' ,a, ' ) =', z
```

Ln: 21 Col: 29

Пример программы на Python



```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
a = 10
b = -5
c = 4.3
d = 3
math.ceil( 4.3 ) = 5.0
math.fabs( -5 ) = 5.0
math.factorial( 10 ) = 3628800
math.floor( 4.3 ) = 4.0
math.exp( -5 ) = 0.006737946999085467
math.log2( 10 ) = 3.321928094887362
math.log10( 10 ) = 1.0
math.log( 3 , 10 ) = 0.47712125471966244
math.pow( 10 , 3 ) = 1000.0
math.sqrt( 10 ) = 3.1622776601683795
>>>
```

Результат выполнения программы с применением функций модуля math

Тригонометрические функции модуля math

| | |
|---------------------|-------------------------|
| math.cos(x) | Возвращает cos числа X |
| math.sin(x) | Возвращает sin числа X |
| math.tan(x) | Возвращает tan числа X |
| math.acos(x) | Возвращает acos числа X |
| math.asin(x) | Возвращает asin числа X |
| math.atan(x) | Возвращает atan числа X |

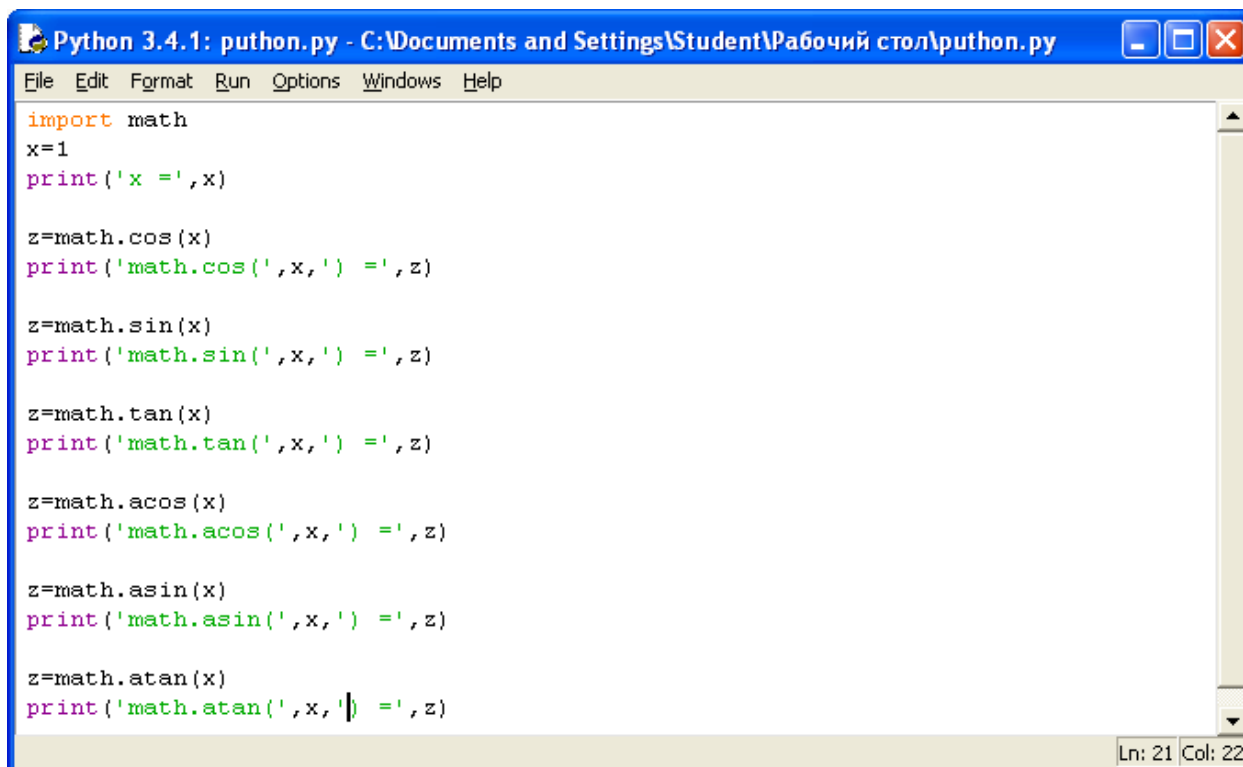
Пример применения вышеописанных функций над числами

В программе определена переменная x, содержащая целое число.

Значение переменной выводится командой print() на экран.

В переменную z помещается результат выполнения тригонометрической функции модуля math.

Затем командой `print()` выводится сообщение в виде используемой функции и ее аргумента и результат ее выполнения



```
Python 3.4.1: puthon.py - C:\Documents and Settings\Student\Рабочий стол\puthon.py
File Edit Format Run Options Windows Help

import math
x=1
print('x =',x)

z=math.cos(x)
print('math.cos(' ,x, ' ) =', z)

z=math.sin(x)
print('math.sin(' ,x, ' ) =', z)

z=math.tan(x)
print('math.tan(' ,x, ' ) =', z)

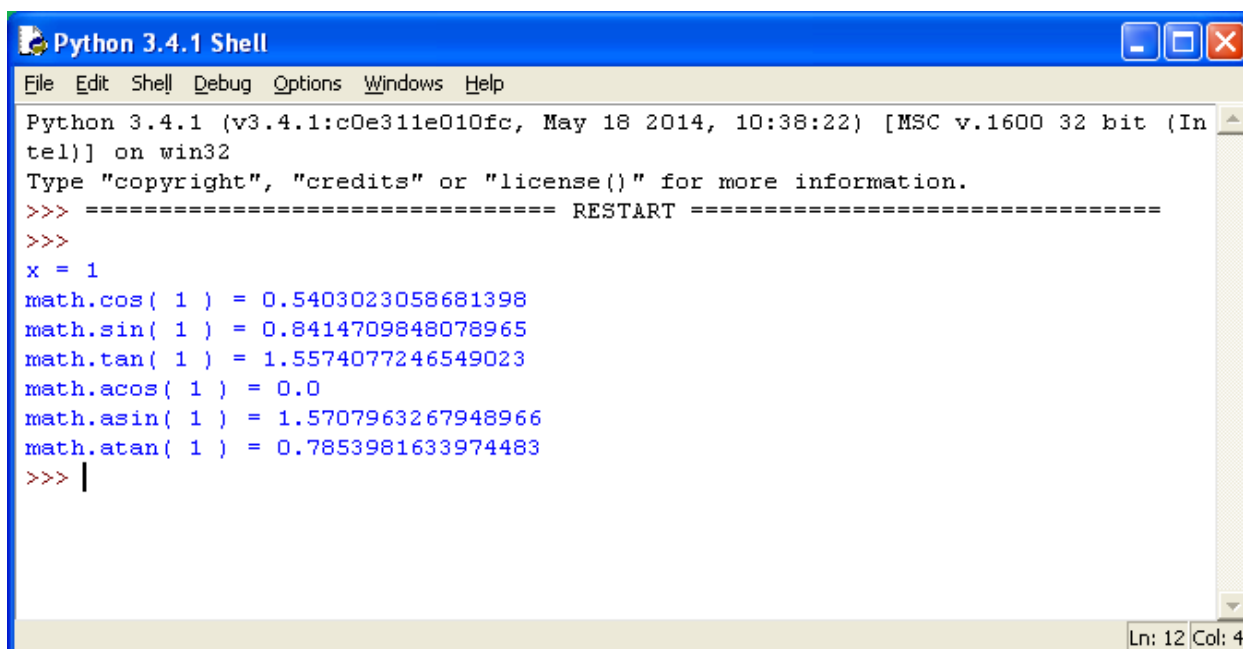
z=math.acos(x)
print('math.acos(' ,x, ' ) =', z)

z=math.asin(x)
print('math.asin(' ,x, ' ) =', z)

z=math.atan(x)
print('math.atan(' ,x, ' ) =', z)

Ln: 21 Col: 22
```

Пример программы с использованием тригонометрических функций модуля `math`



```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help

Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
x = 1
math.cos( 1 ) = 0.5403023058681398
math.sin( 1 ) = 0.8414709848078965
math.tan( 1 ) = 1.5574077246549023
math.acos( 1 ) = 0.0
math.asin( 1 ) = 1.5707963267948966
math.atan( 1 ) = 0.7853981633974483
>>> |

Ln: 12 Col: 4
```

Результат выполнения программы с применением тригонометрических функций модуля `math`

Константы:

- **math.pi** - число Pi.
- **math.e** - число e (экспонента).

Пример

Напишите программу, которая бы вычисляла заданное арифметическое выражение при заданных переменных. Ввод переменных осуществляется с клавиатуры. Вывести результат с 2-мя знаками после запятой.

Задание

$$Z = \frac{9\pi t + 10 \cos(x)}{\sqrt{t} - |\sin(t)|} * e^x$$

x=10; t=1

Решение

Сначала импортируем модуль math. Для этого воспользуемся командой `import math`.

Затем следует ввести значения двух переменных целого типа x и t.

Для ввода данных используется команда `input`, но так как в условии даны целые числа, то нужно сначала определить тип переменных: `x=int(), t=int()`.

Определив тип переменных, следует их ввести, для этого в скобках команды `int()` нужно написать команду `input()`.

Для переменной x это выглядит так: `x=int(input("сообщение при вводе значения"))`.

Для переменной t аналогично: `t=int(input("сообщение при вводе значения"))`.

Следующий шаг - это составление арифметического выражения, результат которого поместим в переменную z.

Сначала составим числитель. Выглядеть он будет так:

`9*math.pi*t+10*math.cos(x)`.

Затем нужно составить знаменатель, при этом обратим внимание на то, что числитель делится на знаменатель, поэтому и числитель и знаменатель нужно поместить в скобки `()`, а между ними написать знак деления `/`.

Выглядеть это будет так: `(9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-math.fabs(math.sin(t)))`.

Последним шагом является умножение дроби на экспоненту в степени x.

Так как умножается вся дробь, то следует составленное выражение поместить в скобки `()`, а уже потом написать функцию `math.pow(math.e,x)`.

В результате выражение будет иметь вид:

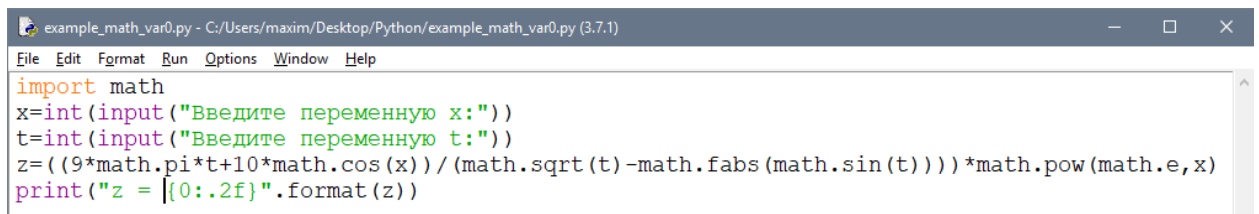
$$z = ((9 * \pi * t + 10 * \cos(x)) / (\sqrt{t} - \sin(t))) * e^x$$

При составлении данного выражения следует обратить внимание на количество открывающихся и закрывающихся скобок.

Командой `print()` выведем значение переменной, отформатировав его командой `format`.

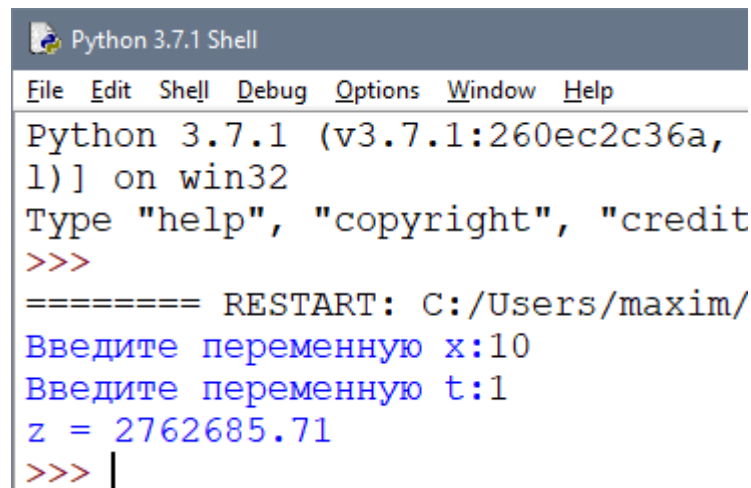
Сам формат записывается в апострофах в фигурных скобках `{}`.

В задаче требуется вывести число с двумя знаками после запятой, значит вид формата будет выглядеть следующим образом: `{0:.2f}`, где 2 - это количество знаков после запятой, а `f` указывает на то, что форматируется вещественное число. При этом перед 2 нужно поставить точку, указав тем самым на то, что форматируем именно дробную часть числа.



```
example_math_var0.py - C:/Users/maxim/Desktop/Python/example_math_var0.py (3.7.1)
File Edit Format Run Options Window Help
import math
x=int(input("Введите переменную x:"))
t=int(input("Введите переменную t:"))
z=((9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-math.fabs(math.sin(t))))*math.pow(math.e,x)
print("z = |{0:.2f}|".format(z))
```

Результат



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a,
1)] on win32
Type "help", "copyright", "credit
>>>
===== RESTART: C:/Users/maxim/
Введите переменную x:10
Введите переменную t:1
z = 2762685.71
>>> |
```