In [43]:
```
# Простые вычисления
# символом "решётка" обозначаются комментарии

x = 10
y = 23

z = x + y
z = x - y
z = x * y
z = y / x

# деление нацело
z = y // x
# z = 2

# остаток от деления
z = y % x
# z = 3

# возведение в степень
z = 2 ** 3
```

In [31]:
```
# если в отдельной яйчейке написать математическое выражение
# и не записывать его в переменную
# то оно будет выведено на экран

1 + ( 2 + 3 ) * 4
```

Out[31]: 21

Вывод значений на экран

In [17]:
```
print ( 123 )

# выводить можно не только отдельные значения
# но и результаты выражений
# сначала будет вычислен результат, а потом напечатн на экран результат
print ( 22 + 3 )

# вывод значений переменных
print( x )


# вывод строк
# строки должны быть в кавычках!
print( "Hello, World!" )


# Вывод одновременно и строк и переменных
print( "z =", z )


# вещественные числа могут выглядеть некрасиво (и не практично)
z = 3.14159265
print( z )
```

```
123
25
10
Hello, World!
z = 3.14159265
3.14159265
```

In [19]:
```
# ограничений до 3 знаков после запятой
# формат выводимого числа описывается в фигурных скобках
# чтобы ограничить число символов (чисел) используется двоетеочее
# после двоеточия указаывается формат числа:
# 5 символов на число (включая точку и возможный занк минус)
# 2 знака после запятой
# f означает что будет выводится вещественное число в обычном формате
print( "{:5.2f}".format(z) )


# общее число символов можно не указывать
print( "{:.2f}".format(z) )


# можно комбинировать строку и описание формата в скобках
print("z = {:.2f}".format(z))

# можно выводить несколько форматированых значений на экран
# каждое будет подставлено вместо соответствующей по порядку паре фигкрных с
кобок
print("z = {:.2f}; x = {}; y = {}".format(z, x, z))
```

```
 3.14
3.14
z = 3.14
z = 3.14; x = 10; y = 3.14159265
```

математические вычисления

In [24]:
```
# математические функции вроде sin или ln изначально недоступны
# чтобы ими воспользоватся нужно подключить математический модуль


# следующий код означает:
# ИЗ модуля math ПОДКЛЮЧИТЬ * (всё)
from math import *

# теперь можно пользоватся содердимым этого модуля!
y = sin ( 0 )
# y = 0

# программисты любят чтобы вместо градусов были радианы
y = sin( 90 )
# y = 0.8939966636005579


# если использовать градусы, то их нужно переводить в радианы
y = sin( radians(90) )
# y = 1

# Кроме многиз математический функций модуль содрежит некоторые константы
# в частности число Пи и E
print( pi )
print( e )
```

```
1.0
3.141592653589793
2.718281828459045
```

In [27]:
```
# математический модуль содержит много всего.
# ВСЁ знать не нужно
# потому что легко посмотреть справку по модулю

# подключим математический модуль (без непостредственно содержимого)
import math

# вызов справки
help( math )
# справка на английском :)
```

```
Help on built-in module math:

NAME
    math

DESCRIPTION
    This module is always available.  It provides access to the
    mathematical functions defined by the C standard.

FUNCTIONS
    acos(...)
        acos(x)

        Return the arc cosine (measured in radians) of x.

    acosh(...)
        acosh(x)

        Return the inverse hyperbolic cosine of x.

    asin(...)
        asin(x)

        Return the arc sine (measured in radians) of x.

    asinh(...)
        asinh(x)

        Return the inverse hyperbolic sine of x.

    atan(...)
        atan(x)

        Return the arc tangent (measured in radians) of x.

    atan2(...)
        atan2(y, x)

        Return the arc tangent (measured in radians) of y/x.
        Unlike atan(y/x), the signs of both x and y are considered.

    atanh(...)
        atanh(x)

        Return the inverse hyperbolic tangent of x.

    ceil(...)
        ceil(x)

        Return the ceiling of x as an Integral.
        This is the smallest integer >= x.

    copysign(...)
        copysign(x, y)

        Return a float with the magnitude (absolute value) of x but the sign
        of y. On platforms that support signed zeros, copysign(1.0, -0.0)
        returns -1.0.

    cos(...)
        cos(x)

        Return the cosine of x (measured in radians).

    cosh(...)
        cosh(x)

        Return the hyperbolic cosine of x.
```

In [36]:
```
# Примеры вычислений

# число сочетаний из 5 по 3
N1 = factorial(5) / ( factorial(5-3) * factorial(3) )
print( N )


# гораздо удобнее использовать переменные, чтобы потом повторять вычисления
n = 5
k = 3
N2 = factorial(n) / ( factorial(n-k) * factorial(k) )
print("n2 = ", N2)

n,m = 4,2
N3 = factorial(n) / ( factorial(n-k) * factorial(k) )
print("n3 = ", N3)
```

```
10.0
n2 =  10.0
n3 =  4.0
```

In [37]:
```
# спики (почти как масивы только лучше)
# список - набор значений
# чтобы создать список нужно указать набор значений в квадратных скобкахчере
з запятую

X = [1,2,3]

Y = [10.8, 30.8, -8.2, 0, 2]

# Z - это пустой список
Z = []
```

In [42]:
```
# Операции со списками

# сложение
X = [11, 22 ,33] + [42]
#печать списков
print(X)

# вычисление количества элементов (длинны списка)
n = len(X)
# n = 4

# получение конкретного элемента из списка по порядковому номеру
# нумерация начинается с НУЛЯ! (так тоже любят программисты)
print(X[1])  # 20
print(X[0])  # 10

# последний элемент
print(X[-1]) # 42


# сумма элементов списка
s = sum( X )
print( s )  # 108
```

```
[11, 22, 33, 42]
22
11
42
108
```