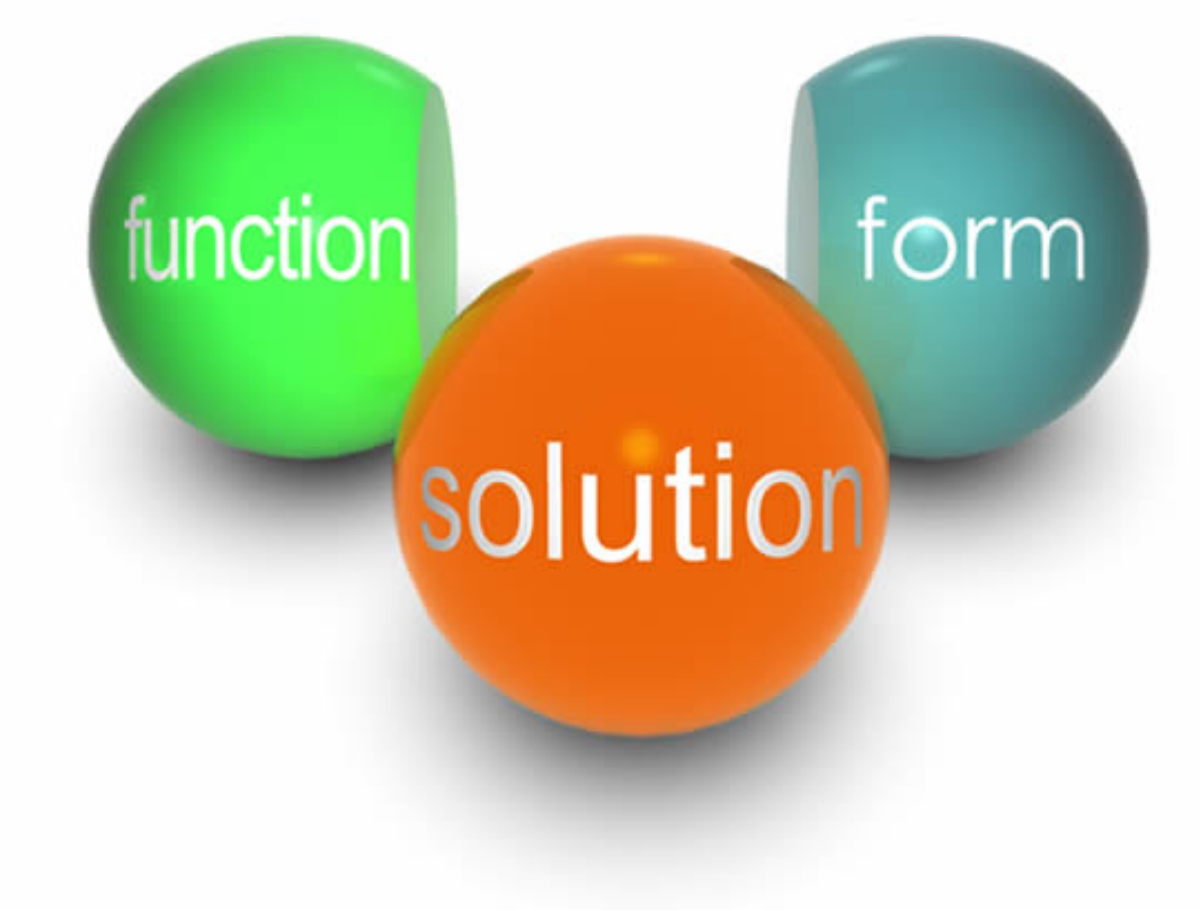
py4kids (https://github.com/wgong/py4kids)

Functions and Modules

In this lesson, we learn the building blocks of python:

- Function - named collection of python statements
- Module - a named file with functions, variables (you may call it package, library)

The purpose to write modular codes and to reuse (recycle).



In [1]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

Out[1]:

Function

Function is useful, it has specific functionality. It has the following parts:

- keyword **def** defines a function;
- function has a name: create_a_number_list, it should be meanful;
- function may have input parameters, e.g. start,stop, stride
- function has a body, which implements functionality of the function

In [2]:
```
# this function does nothing
def dummy_function():
    pass
```

In [3]:
```
print(dummy_function())
```

```
None
```

In [4]:
```
dummy_function()
```

In [5]:
```
def greeting(name):
    print('Hi, %s' % name)
```

In [6]:
```
greeting('Allen')
```

```
Hi, Allen
```

In [7]:
```
greeting('Teacher')
```

```
Hi, Teacher
```

In [8]:
```
# this function creates a list of integer numbers
def create_integer_list(start, stop, stride=1):
    return list(range(start, stop, stride))
```

In [9]:
```
print(create_integer_list(0,10,3))
```

```
[0, 3, 6, 9]
```

In [10]:
```
import numpy as np
```

In [11]:
```
np.arange(0.2, 1, 0.1)
```

Out[11]:
```
array([ 0.2,  0.3,  0.4,  0.5,  0.6,  0.7,  0.8,  0.9])
```

In [12]:
```
def create_real_list(start, stop, stride):
    import numpy as np
    return np.arange(start, stop, stride)
```

In [14]:
```
# this function has error handling to avoid crash
def create_real_list2(start, stop, stride):
    import numpy as np
    if stride != 0.0:
        return np.arange(start, stop, stride)
    else:
        print('[ERROR] stride is zero')
        return None
```

In [15]:
```
create_real_list2(-0.1,2.0,0.2)
```

Out[15]:
```
array([-0.1,  0.1,  0.3,  0.5,  0.7,  0.9,  1.1,  1.3,  1.5,  1.7,  1.9])
```

In [16]:
```
# this function has error handling to avoid crash
def create_real_list2(start, stop, stride):
    import numpy as np
    if stride != 0.0:
        return np.arange(start, stop, stride)
    else:
        print('[ERROR] stride is zero')
        return None
```

In [17]:
```
create_real_list2(-0.1,2.0,0)
```

```
[ERROR] stride is zero
```

variable scope

In [18]:
```
eng_name = "Mr Wang"
chn_name = u"老王"
```

In [19]:
```
all(u'\u4e00' <= c <= u'\u9fff' for c in eng_name)
```

Out[19]:
```
False
```

In [20]:
```
all(u'\u4e00' <= c <= u'\u9fff' for c in chn_name)
```

Out[20]:
```
True
```

In [21]:
```
# this function is useful to detect if a string is written in Chinese
def is_chinese(s):
    return all(u'\u4e00' <= c <= u'\u9fff' for c in s)
```

In [22]:
```
is_chinese(chn_name)
```

Out[22]:
```
True
```

In [23]:
```
def greeting2(name):
    lang_dic = {'chn':'你好', 'eng':'Hello'}
    if is_chinese(name):
        print(lang_dic['chn'], ', ', name)
    else:
        print(lang_dic['eng'], ', ', name)
```

In [24]:
```
greeting2(chn_name)
```

```
你好 ,  老王
```

In [25]:
```
greeting2(eng_name)
```

```
Hello ,  Mr Wang
```

In [26]:
```
lang_dic
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-26-5bb30bc4276a> in <module>()
----> 1 lang_dic

NameError: name 'lang_dic' is not defined
```

any variable defined inside a function is called local variable

In [27]:
```
g_lang_dic = {'chn':'您好', 'eng':'Hello'}
def greeting3(name):
    if is_chinese(name):
        print(g_lang_dic['chn'], ', ', name)
    else:
        print(g_lang_dic['eng'], ', ', name)
```

In [28]:
```
g_lang_dic
```

Out[28]:
```
{'chn': '您好', 'eng': 'Hello'}
```

In [29]:
```
greeting3(chn_name)
```

```
您好 ,  老王
```

g_lang_dic is a global variable

math formula

quadratic equation (https://www.wikiwand.com/en/Quadratic_equation)

$a \cdot x^2 + b \cdot x + c = 0$

$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

In [30]:
```
def quad_eq(a, b, c):
    """
    Solving a quadratic equation:
        https://www.wikiwand.com/en/Quadratic_equation
    """
    if a == 0.0:
        if b == 0.0:
            print('Both a and b are zero, no solution')
            return None
        else:
            x = - c/b
            return x
    else:
        import math
        x1 = (-b + math.sqrt(b**2 - 4*a*c)) / (2.0*a)
        x2 = (-b - math.sqrt(b**2 - 4*a*c)) / (2.0*a)
        return [x1, x2]
```

We need to test it for various cases

In [31]:
```
print(quad_eq(1,-2,1))
```

```
[1.0, 1.0]
```

In [32]:
```
print(quad_eq(1,2,3))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-32-cefefd2b68e1> in <module>()
----> 1 print(quad_eq(1,2,3))

<ipython-input-30-ed5713c9ed9b> in quad_eq(a, b, c)
     13     else:
     14         import math
---> 15         x1 = (-b + math.sqrt(b**2 - 4*a*c)) / (2.0*a)
     16         x2 = (-b - math.sqrt(b**2 - 4*a*c)) / (2.0*a)
     17         return [x1, x2]

ValueError: math domain error
```

In [33]:
```
print(quad_eq(1,5,5))
```

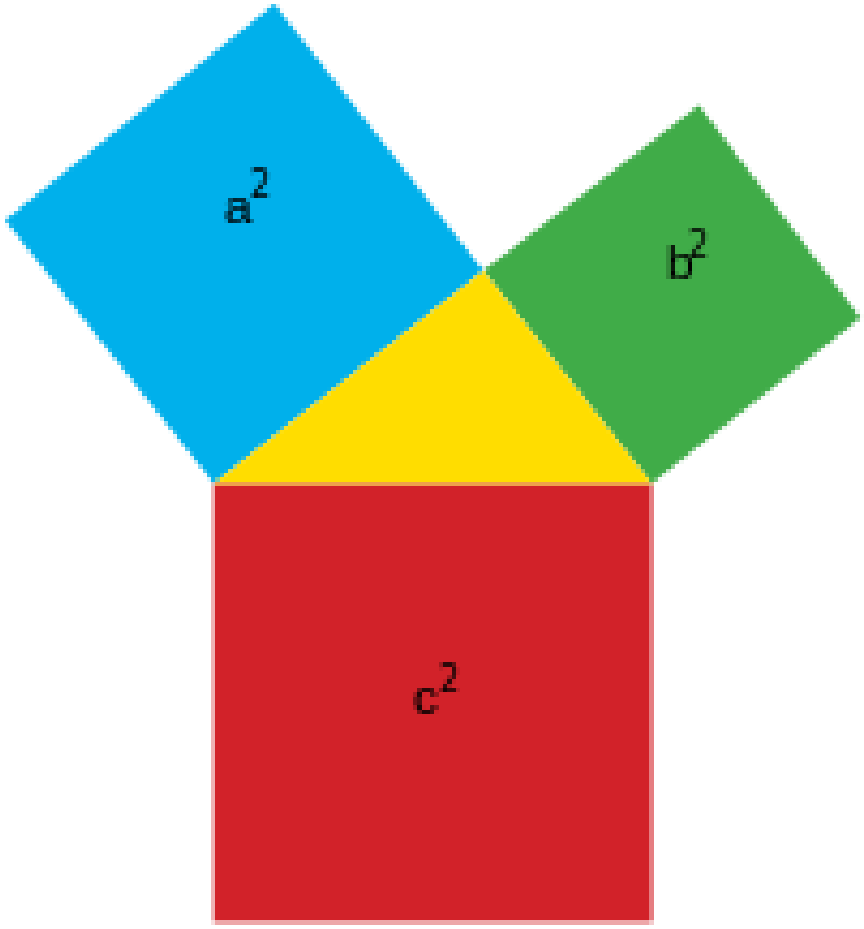
```
[-1.381966011250105, -3.618033988749895]
```

In [34]:
```
print(quad_eq(0,0,1))
```

```
Both a and b are zero, no solution
None
```

The Pythagorean Theorem (http://www.montereyinstitute.org/courses/DevelopmentalMath/COURSE_TEXT2_RESOURCE/U07_L1_T4_text_final.html)

$$a^2 + b^2 = c^2$$



In [35]:
```
def pythagorean(a,b,c):
    import math
    if c is None and a is not None and b is not None:
        return math.sqrt(a**2 + b**2)
    elif a is None and b is not None and c is not None and c >= b:
        return math.sqrt(c**2 - b**2)
    elif b is None and a is not None and c is not None and c >= a:
        return math.sqrt(c**2 - a**2)
    else:
        return "No solution"
```

Test time

In [36]:
```
print(pythagorean(3,4,None))
```

```
5.0
```

In [37]:
```
print(pythagorean(None,4,5))
```

```
3.0
```

In [38]:
```
print(pythagorean(None,5,5))
```

```
0.0
```

In [39]:
```
print(pythagorean(None,7,5))
```

```
No solution
```

Module

Modules are used to group functions, variables, and other things together into larger, more powerful programs.

Module is like Lego

In [40]:
```
import py4kids as mm
```

In [41]:
```
print(mm.version)
```

```
1.0
```

In [42]:
```
print(mm.g_lang_dic)
```

```
{'chn': '您好', 'eng': 'Hello'}
```

In [43]:
```
mm.greeting('Frank')
```

```
Hi, Frank
```

In [44]:
```
mm.create_real_list(1.1, 9.1, 0.3)
```

Out[44]:
```
array([ 1.1,  1.4,  1.7,  2. ,  2.3,  2.6,  2.9,  3.2,  3.5,  3.8,  4.1,
        4.4,  4.7,  5. ,  5.3,  5.6,  5.9,  6.2,  6.5,  6.8,  7.1,  7.4,
        7.7,  8. ,  8.3,  8.6,  8.9])
```

In [45]:
```
monkey = '孙悟空'
mm.greeting3(monkey)
```

```
您好 ,  孙悟空
```

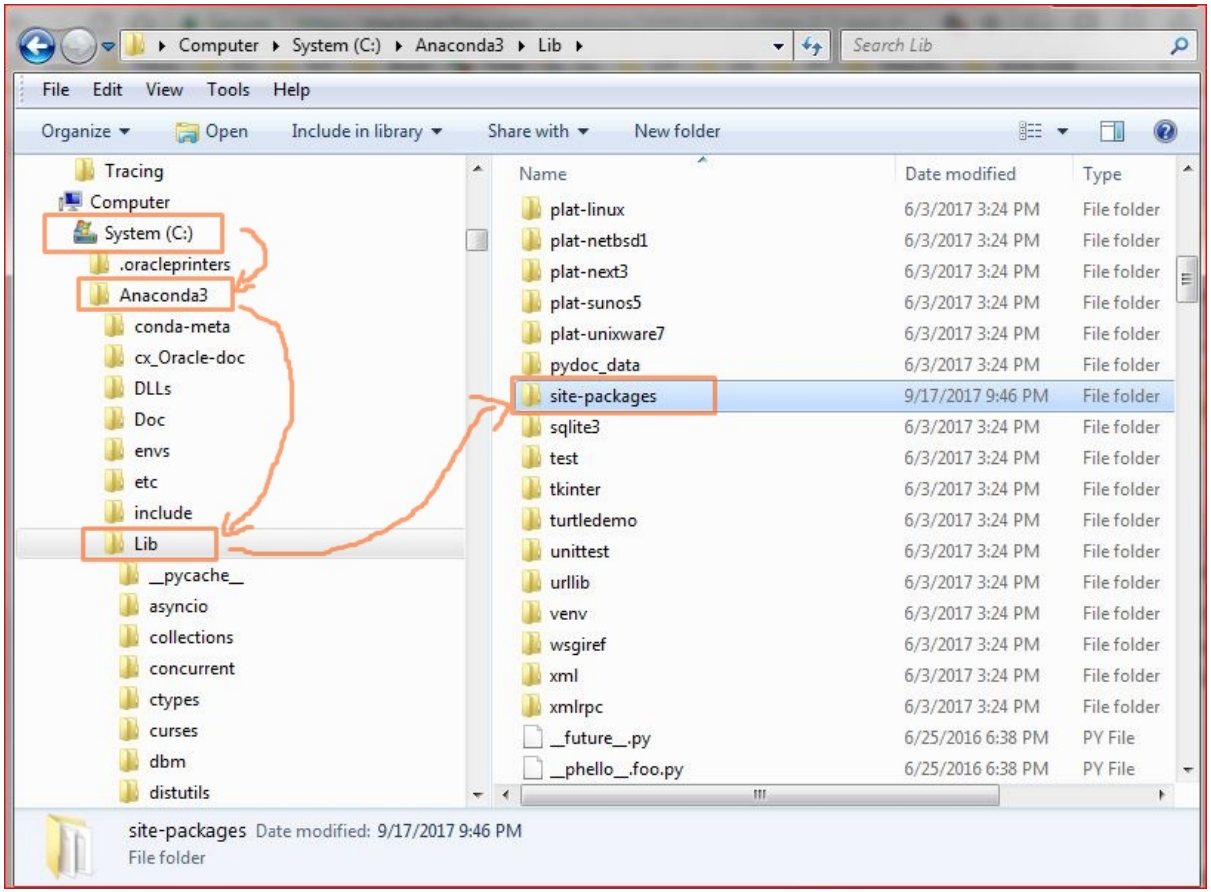
In [46]:
```
mm.quad_eq(1,-6,9)
```

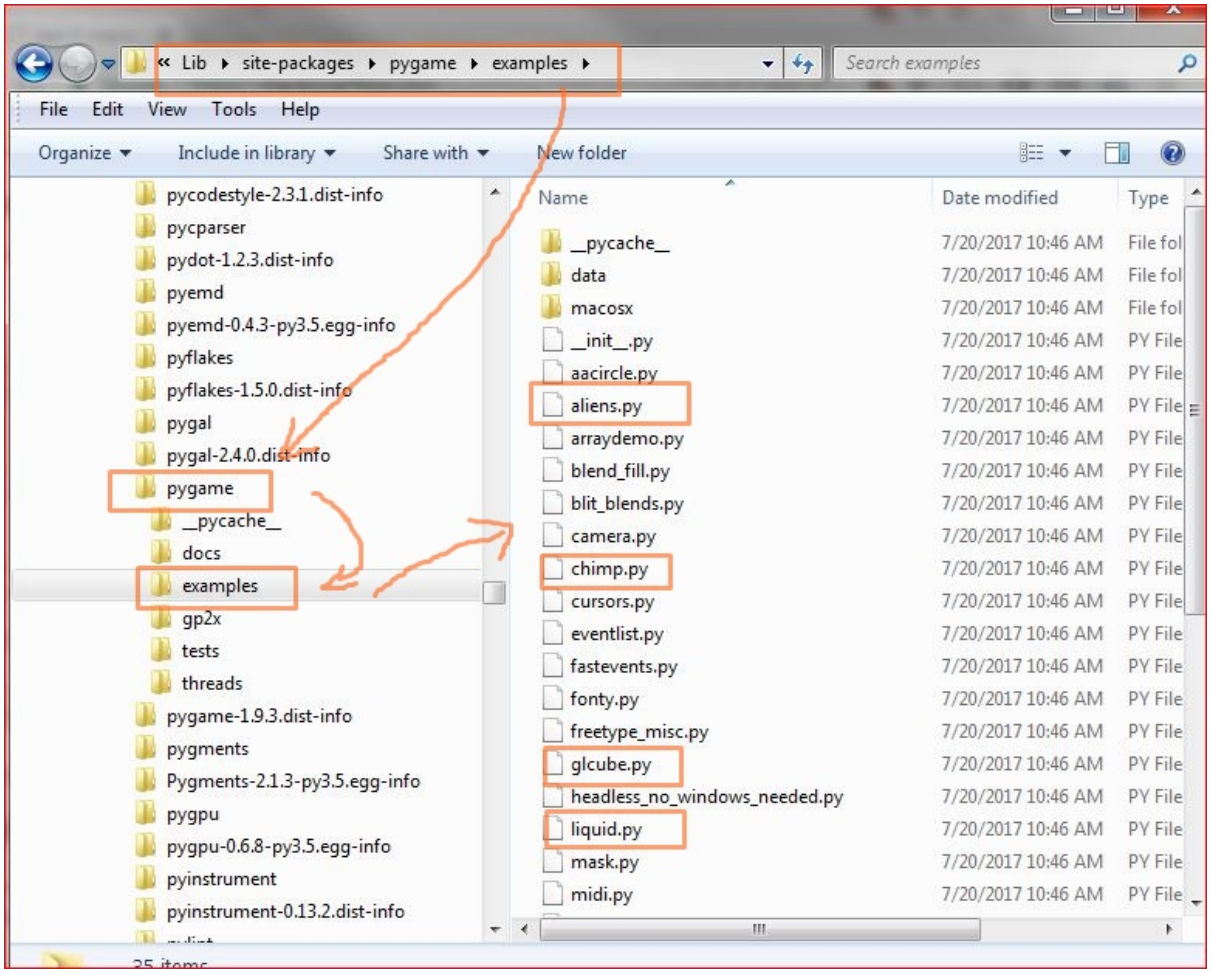
Out[46]:
```
[3.0, 3.0]
```

In [49]:
```
mm.pythagorean(5,12,None)
```

Out[49]:
```
13.0
```

pygame (https://www.pygame.org/news)


Computer ▸ System (C:) ▸ Anaconda3 ▸ Lib ▸
Search Lib
File Edit View Tools Help
Organize Open Include in library Share with New folder
Tracing
Computer
System (C:)
.oracleprinters
Anaconda3
conda-meta
cx_Oracle-doc
DLLs
Doc
envs
etc
include
Lib
__pycache__
asyncio
collections
concurrent
ctypes
curses
dbm
distutils
Name
Date modified
Type
plat-linux 6/3/2017 3:24 PM File folder
plat-netbsd1 6/3/2017 3:24 PM File folder
plat-next3 6/3/2017 3:24 PM File folder
plat-sunos5 6/3/2017 3:24 PM File folder
plat-unixware7 6/3/2017 3:24 PM File folder
pydoc_data 6/3/2017 3:24 PM File folder
site-packages 9/17/2017 9:46 PM File folder
sqlite3 6/3/2017 3:24 PM File folder
test 6/3/2017 3:24 PM File folder
tkinter 6/3/2017 3:24 PM File folder
turtledemo 6/3/2017 3:24 PM File folder
unittest 6/3/2017 3:24 PM File folder
urllib 6/3/2017 3:24 PM File folder
venv 6/3/2017 3:24 PM File folder
wsgiref 6/3/2017 3:24 PM File folder
xml 6/3/2017 3:24 PM File folder
xmlrpc 6/3/2017 3:24 PM File folder
__future__.py 6/25/2016 6:38 PM PY File
__phello__.foo.py 6/25/2016 6:38 PM PY File
site-packages Date modified: 9/17/2017 9:46 PM
File folder



open a terminal, go to pygame examples folder

On Windows:

```
CMD> c:
CMD> cd C:\Anaconda3\Lib\site-packages\pygame\examples
CMD> python chimp.py
```

On Mac or Unix:

```
CMD> cd ~\Anaconda3\Lib\site-packages\pygame\examples
CMD> python chimp.py
```

```
```

Reference

Learning Python: From Zero to Hero (https://medium.freecodecamp.org/learning-python-from-zero-to-hero-120ea540b567) is a useful review of basic python

In [ ]: