Python中除了内置的函数以外，还可以使用关键字def和lambda来定义

4.1 自定义函数def

自定义一个函数，返回一个序列。序列中每个数字都是前两个数字之和（斐波那契数列）。

In [22]:

```
def fibs(num): # 位置参数
    result = [0,1] # 新建列表存储数列的值
    for i in range(2,num): # 循环num-2次
        a = result[i-1] + result[i-2]
        result.append(a) # 将值追加至列表
    return result # 返回列表
fibs(5)
```

Out[22]:

```
[0, 1, 1, 2, 3]
```

自定义函数，打印不同方式传入的参数。

=必备参数：必备参数须以正确的顺序传入函数，调用时的数量必须和声明时的一样。

=关键字参数：函数调用时使用等号赋值的形式传入参数。

=默认参数：调用函数时，缺省参数的值如果没有传入，则被认为是默认值。

=不定长参数：有时可能需要一个函数能处理比当初声明时更多的参数，这些参数叫做不定长参数，声明时不会命名。*args，kwargs：加了星号（）的变量args会存放所有未命名的变量参数，args为元组，而加**的变量kwargs会存放命名参数，即形如key=value的参数， kwargs为字典。**

In [2]:

```
def func(a=1, b=2, *args):
    print(a+b)
    print(args)

# 必备参数
func(0,4)

# 关键字参数
func(b=4,a=0)

# 默认参数
func()

# 不定长参数
func(1,2,3,4,5)
```

```
4
()
4
()
3
()
3
(3, 4, 5)
```

4.2 匿名函数

lambda来创建匿名函数。相比于普通函数而言，匿名函数只是一个表达式，函数体比def简单很多仅仅能在lambda表达式中封装有限的逻辑进去

In [1]:

```
a = lambda x,y:x+y # lambda 参数: 表达式
a(1,2) # 调用函数
```

Out[1]:

```
3
```

5.面向对象

面向对象：把数据和对数据的操作方法放在一起，作为一个相互依存的整体——对象。把计算机程序视为一组对象的集合，而每个对象都可以接收其他对象发过来的消息，并处理这些消息，计算机程序的执行就是一系列消息在各个对象之间传递。

面向对象的优势：

-提高代码复用性。

-使程序的编码更加灵活，提高了代码的可维护性。

-提高程序的可扩展性。

-提高开发效率。

Python中创建类使用关键字class，并且具备封装、多态和继承的特点。

-封装：封装是一个概念，它的含义是把方法、属性、事件集中到一个统一的类中，并对使用者屏蔽其中的细节。

-继承：是一种创建类的方法，在现有类（被继承的类）基础上，进行扩展生成新的类，被称为子类。被继承的类称为父类、基类、超类。

-多态：一个同样的函数对于不同的对象可以具有不同的实现。

In [28]:

```
class Person(object): # class 类名(父类), 在py3中所有的类默认继承object
    def __init__(self, age, name): # self:指实例化的对象
        # __init__方法对象初始化时调用的方法
        self.__age = age # 类属性
        self.name= name
    def func(self): # 方法
        print("my name is %s"%(self.name))
```

In [33]:

```
# 实例化对象
p1 = Person(age="18",name="张三")
print(p1.age)
p1.func()
```

```
18
my name is 张三
```

Python是一门动态语言

In [8]:

```
p1.ID = 2020001
```

```
p1.ID = 2020001
```

In [10]:

```
print(p1.ID)
print(Person.ID)
```

```
2020001


---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-10-b1d2b85a9b1d> in <module>
      1 print(p1.ID)
----> 2 print(Person.ID)

AttributeError: type object 'Person' has no attribute 'ID'
```

In [14]:

```
# 继承
class Man(Person):
    pass
```

In [15]:

```
m = Man(age="19", name="李四")
m.func()
```

```
my name is 李四
```

默认情况下，属性在Python中都是“public”，类所在模块和导入了类所在模块的其他模块都可以访问到。如果类中的某些属性不想被外界访问或者继承可以对其私有化。

-在属性或方法前加上一个下划线。可以防止模块的属性用“from mymodule import *”来加载，它只可以在本模块中使用。

-在方法或属性前加双下划线。可以实现完全私有化。