python之父类的init方法

写代码的时候忽然要实现继承父类所有的方法并且修改父类的__init_参数，这样就叫方法重载。

```
方法重载(overloading method)：子类修改父类的一些参数就叫重载。
方法重写(overiding  method)：子类不想原封不动的继承父类的方法，需要进行覆盖。#
修改父类的方法
#重写父类的方法的目的是为了给他扩展功能，父类的方法已经不能满足需求

1.子类继承父类时，如果 init  方法被重写，可以使用super函数调用父类的 init  方法，否则被改写的
__init__方法会覆盖原有的自动继承的_init__方法。
2.类的继承，就是子类继承除了构造函数之外的所有的东西， init  方法不是构造方法（函数）
3.子类继承父类时， init  方法被自动继承，并在创建实例时，自动调用。

核心思想就一句话，先调用一下你要重写的父类方法
```

方法重载

```
class A():
  def __init__(self,name=None,models=None):
      self.name=name
      self.models=models
  def function(self):
      print(self.name)
      print(self.models)
class B(A):
  def __init__(self):

      A.__init__(self,name=None,models=None)

      self.models='modeify'
#主函数，代码执行的开始
if __name__ == '__main__':
  a=A()
  a.function()

  b=B()
  b.function()
```

执行结果：

```
None
None
None
modeify
```

方法重写

```
class A():
  def __init__(self,name=None,models=None):
      self.name=name
      self.models=models
  def function(self):
      print(self.name)
      print(self.models)
class B(A):
  def __init__(self):

        A.__init__(self,name=None,models=None)
        self.models='modeify'
  def function(self):

      print(1)
 #主函数，代码执行的开始
if __name__ == '__main__':
  a=A()
  a.function()

  b=B()
  b.function()
```

执行结果

```
None
None
1
```