Python的面向对象编程

# 定义类

class Father():  
 # 构造方法  
 def \_\_init\_\_(self,name,age):  
 self.name = name  
 self.age = age  
 # 一般方法  
 def toString(self):  
 print('name:{0},age:{1}'.format(self.name,self.age))  
  
f = Father('甜甜',22)  
f.toString()

# 类的继承

**可以实现多继承，一次可以继承多个类**。

**既可以继承构造方法，也可以继承一般方法。**

class GrandFather():  
 def smoke(self):  
 print('Smoking')  
  
class Father(GrandFather):  
 # 构造方法  
 def \_\_init\_\_(self,name,age):  
 self.name = name  
 self.age = age  
 # 一般方法  
 def toString(self):  
 print('Father:name={0},age={1}'.format(self.name,self.age))  
  
# 继承只需要在括号中声明即可  
class Son(Father,GrandFather):  
 #构造方法继承Father  
 # 重载  
 def toString(self):  
 print('Son:name={0},age={1}'.format(self.name,self.age))  
 # 新方法  
 def study(self):  
 print('小甜甜在学习')  
  
f = Father('大甜甜',22)  
f.toString()#Father:name=大甜甜,age=22  
f.smoke()#Smoking  
  
s = Son('小甜甜',-4)  
s.toString()#Son:name=小甜甜,age=-4  
s.smoke()#Smoking

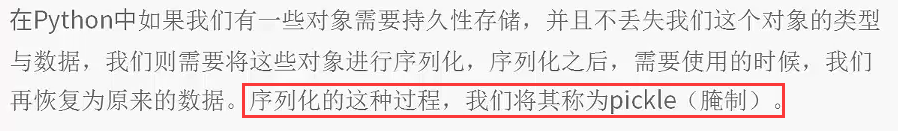
# 引入其他文件中的类

# tiantian.py  
class TianTian():  
 def \_\_init\_\_(self,name,age):  
 self.name = name  
 self.age = age  
 def sing(self):  
 print('{0} is Sing'.format(self.name))

# honghong.py  
import tiantian  
tt = tiantian.TianTian('甜甜',22)  
tt.sing()  
  
from tiantian import TianTian  
tt = TianTian('甜甜',22)  
tt.sing()

# pickle模块：对象序列化和反序列化

**Python的pickle模块**实现了基本的数据**序列化和反序列化**。通过pickle模块的序列化操作我们能够将程序中运行的对象信息保存到文件中去，永久存储；通过pickle模块的反序列化操作，我们能够从文件中创建上一次程序保存的对象。





## 序列化

利用pickle模块中的**dump或dumps方法**。



**pickle.dump(obj, file, [,protocol])**

　　注解：将对象obj保存到文件file中去。

　　　protocol为序列化使用的协议版本，0：ASCII协议，所序列化的对象使用可打印的ASCII码表示；1：老式的二进制协议；2：2.3版本引入的新二进制协议，较以前的更高效。其中协议0和1兼容老版本的python。protocol默认值为3。

**file：对象保存到的类文件对象。file必须有write()接口， file可以是一个以'w'方式打开的文件或者一个StringIO对象或者其他任何实现write()接口的对象。文件对象需要是二进制模式打开的。**

file1 = open('./student.txt','wb')#模式选择wb

反序列化

利用pickle模块的**load或loads方法**。



**pickle.load(file)**

　　注解：从file中读取一个字符串，并将它重构为原来的python对象。

**file: 类文件对象，有read()和readline()接口**。

file2 = open('./student.txt','rb')#模式选择rb

## 示例

### 保存字符串形式：dumps与loads

import pickle#导入pickle模块  
list1 = ['ming','yue','ji','shi','you']  
list\_p = pickle.dumps(list1) #序列化  
print(list\_p)#b'\x80\x03]q\x00(X\x04...  
# 反序列化  
list2 = pickle.loads(list\_p)  
print(type(list2))#<class 'list'>  
print(list2)#['ming', 'yue', 'ji', 'shi', 'you']

创建类Student：

class Student():  
 def \_\_init\_\_(self,name):  
 self.name = name  
 def printName(self):  
 print(self.name)  
  
student = Student('小甜甜')  
sp = pickle.dumps(student)  
print(sp)  
student2 = pickle.loads(sp)  
student2.printName()#小甜甜

### 保存到文件：dump与load

**先序列化的对象，在反序列的时候，先加载出来，满足先进先出的原则。FIFO**

class Student():  
 def \_\_init\_\_(self,name):  
 self.name = name  
 def printName(self):  
 print(self.name)  
  
student1 = Student('小甜甜')  
student2 = Student('小园园')  
file1 = open('./student.txt','wb')  
pickle.dump(student1,file1,True)  
pickle.dump(student2,file1,True)  
  
file1.close()#文件对象一定要及时关闭，否则报错EOFError: Ran out of input  
  
file2 = open('./student.txt','rb')  
student12 = pickle.load(file2)  
student22 = pickle.load(file2)  
file2.close()#文件对象一定要及时关闭，否则报错  
student12.printName()#小甜甜  
student22.printName()#小园园

## 注意事项

### 文件操作后，一定要close，否则会报错

EOFError: Ran out of input

### 序列化时，文件要以wb模式打开

### 反序列化时，文件以rb模式打开

### 可以将多个对象，序列化到一个文件中，反序列化时，满足FIFO原则。