Python 面向对象(封装继承多态)

```
"""
实现继承 和  多继承
"""

class animal(object):

    def run(self):
        print('动物可以跑')
```

```
"""
使用更高级的方式定义类属性

"""

class peopel():

    @property
    def name(self):
        return self._name

    @name.setter
    def name(self,name):
        self._name  = name
```

```
from PythonOOP.Animal import animal

"""
多继承就直接在括号中以逗号的方式隔开

"""
class person(animal):

    # 如果是多态，子类重写父类的方法
    def run(self):
        print('人可以跑的更快')

```

```
"""
1.一个基本的类
2.涵盖私有,私有的语法格式是在名字前面加上__  对于私有方法可以设置set和get
"""
class Student(object):
    def __init__(self,name,age,score):
        self.name = name
        self.age = age
        self.__score = score
        print(self.__score)


```

```
13
14      def outstr(self):
15          print('%s: %s' % (self.name, self.age))
16
17      def getScore(self):
18          return self.__score
19
20
21  if __name__ == '__main__':
22      pass
```

```
 1
 2  """
 3  调用基本的类
 4  """
 5  from PythonOOP.Student import Student
 6  from PythonOOP.Person import person
 7  from PythonOOP.Animal import animal
 8  from PythonOOP.People import peopel
 9  s  =  Student('wwj',19,100)
10  s.outstr()
11  print(s.getScore())
12
13  p = person()
14  p.run()
15
16  """
17  多态的含义
18  """
19  def duotai(animal):
20      animal.run()
21
22
23  duotai(p)
24
25
26  """
27  获得对象的信息，甚至判断是否  可以考虑用type() instance(实例,类名)
28
29  dir(实例)  获取当前类里面的所有方法
30  """
31
32  """
33  如果申明类变量 就直接在class中加入类变量即可
34  """
35
36  """
37  测试类中的高级语法
38  """
39
40  peo = peopel()
41  peo.name = 18
42  print(peo.name)
43
44
45  """
46  关于 if __name__ = '_main' 的说明
```

```
参考博客:https://blog.csdn.net/qq_27017791/article/details/80212016
"""
```