# 继承

## 课程目标

* 能够说出为什么需要继承
* 能够定义两个类并建立两个类间的继承关系
* 能够完成对父类中已有方法进行重写的操作
* 能够完成反恐精英基础版案例

## 课程内容

* 继承
* 重写
* 多继承
* 多态
* 鸭子类型

### 1.继承

#### 1.1继承的概念

继承描述的是一种类间关系，一个类A从另一个类B获取成员信息，称类A继承自类B。

提供成员信息的类称父类（基类），获取成员信息的类称子类（派生类）。

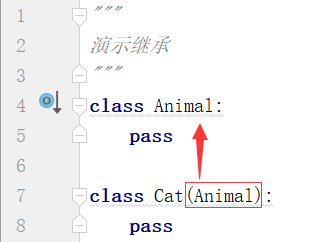
#### 1.2继承的作用

子类可以使用父类的成员（成员变量，成员方法）

#### 1.3继承语法格式

class 类名（父类名）：

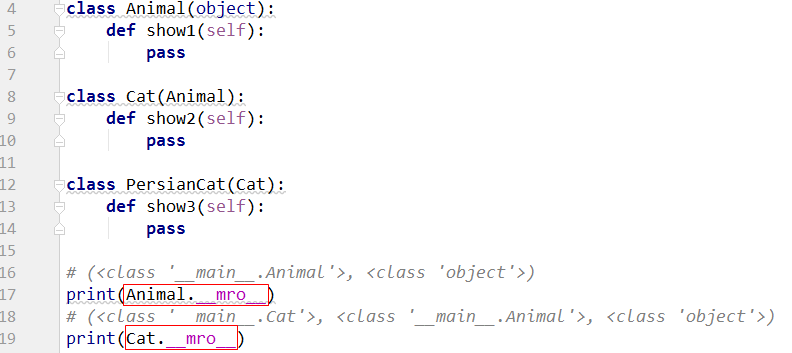
pass



#### 1.4获取继承关系图谱

使用Python内置属性\_\_mro\_\_可以查看继承关系

语法格式：类名.\_\_mro\_\_



说明：mro即Method Resolution Order方法解析顺序

#### 1.5．注意事项

* + - 子类可以添加父类没有的成员
    - 父类私有成员不可被继承

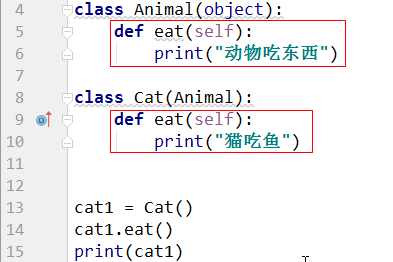
### 2.重写

#### 2.1重写的概念

重写指在子类中定义与父类相同名称的成员方法。

#### 2.2重写的作用

子类对父类的成员方法进行重写后，使用子类对象调用该方法时，将执行子类中重写后的方法



#### 2.3在子类中访问父类被重写的方法

格式一：

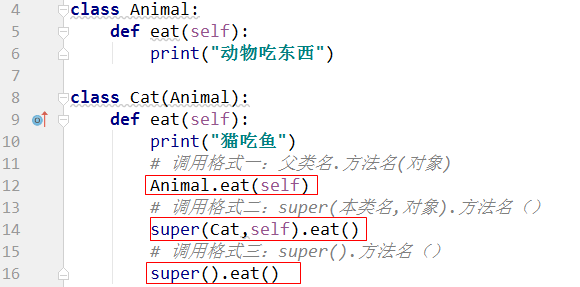
父类名.方法名(对象)

格式二：

super(本类名,对象).方法名（）

格式三：(推荐)

super().方法名（）



### 3.多继承

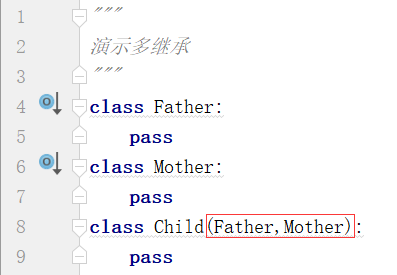
#### 3.1多继承的概念

一个类同时继承多个类，称为多继承

#### 3.2定义多继承语法格式

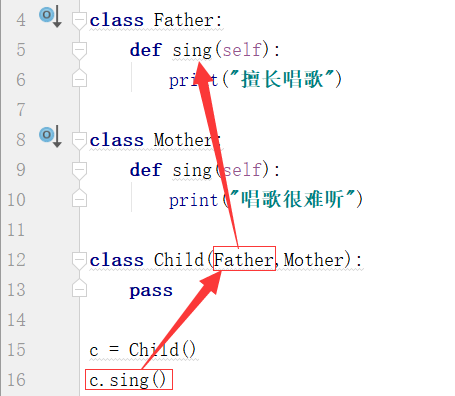
class 类名（父类名1，父类名2…）：

pass



#### 3.3 多继承关系中访问父类成员冲突

多继承关系中，当多个父类具有同名的成员，子类调时该成员时先调用继承关系中的第一个声明的类的成员。



### 4.多态

#### 4.1多态的概念

一个对象具有多种形态，在不同的使用环境中以不同的形态展示其功能，称该对象具有多态特征。

#### 4.2鸭子类型

鸭子类型是一种特殊的调用现象，当对象在语法层面能够满足调用关系，但对象并不具有对应的对象形态，称该对象此时具备鸭子类型

鸭子类型是一种特殊的多态表现形式

#### 4.3注意事项

多态通常发生在继承关系的基础之上。

### 案例：反恐精英基础版

"""  
演示反恐精英案例  
对一个匪徒  
分析：  
1.定义人类，描述公共属性 life:100 name:姓名要传参  
2.定义出英雄与恐怖分子类  
3.定义主函数描述枪战过程 main，创建两个对象  
4.定义开枪方法，分成两个方法，Hero Is都有  
 定义的方法要传入被射击的对象  
 被射击对象的生命值要进行减少  
5.主程序中调用开枪操作  
6.开枪操作后，要在主程序中显示每个人的状态信息  
7.定义Person类的\_\_str\_\_方法，用于显示每个人的状态  
8.设置开枪操作为反复操作  
 再设置停止条件：一方生命值<=0  
 停止循环使用break  
"""  
class Person:  
 def \_\_init\_\_(self,name):  
 self.name = name  
 self.life = 100  
 def \_\_str\_\_(self):  
 return "%s当前的生命值为：%d" %(self.name,self.life)  
class Hero(Person):  
 def fire(self ,p):  
 damage = 40  
 print("%s向%s开枪，造成了%d伤害" % (self.name,p.name,damage))  
 p.life = p.life - damage  
class Is(Person):  
 def fire(self, p):  
 damage = 10  
 print("%s向%s开枪，造成了%d伤害" % (self.name, p.name, damage))  
 p.life = p.life - damage  
  
def main():  
 h = Hero("【英雄】")  
 is1 = Is("【不要命】")  
 while True:  
 h.fire(is1)  
 is1.fire(h)  
 print(h)  
 print(is1)  
 #设置结束条件  
 if h.life <= 0:  
 print("%s死亡，枪战结束" % h.name)  
 break  
 if is1.life <= 0:  
 print("所有恐怖份子全部死亡，枪战结束")  
 break  
main()

### 案例：反恐精英修复版

"""  
演示反恐精英案例——修复版  
对一个匪徒  
分析：  
-----------------------基础版-----------------------  
1.定义人类，描述公共属性 life:100 name:姓名要传参  
2.定义出英雄与恐怖分子类  
3.定义主函数描述枪战过程 main，创建两个对象  
4.定义开枪方法，分成两个方法，Hero Is都有  
 定义的方法要传入被射击的对象  
 被射击对象的生命值要进行减少  
5.主程序中调用开枪操作  
6.开枪操作后，要在主程序中显示每个人的状态信息  
7.定义Person类的\_\_str\_\_方法，用于显示每个人的状态  
8.设置开枪操作为反复操作  
 再设置停止条件：一方生命值<=0  
 停止循环使用break  
-----------------------修复版-----------------------  
9.修复英雄的信息显示模式  
 状态描述 0 - 1- 70 - 99- 100  
 if..elif.. and组合条件  
10.修复生命值为负的问题  
 射击时如果生命值<伤害值，生命值 = 0，否则正常减生命  
"""

class Person:  
 def \_\_init\_\_(self,name):  
 self.name = name  
 self.life = 100  
 def \_\_str\_\_(self):  
 return "%s当前的生命值为：%d" %(self.name,self.life)  
class Hero(Person):  
 def fire(self ,p):  
 damage = 40  
 print("%s向%s开枪，造成了%d伤害" % (self.name,p.name,damage))  
 if p.life < damage:  
 p.life = 0  
 else:  
 p.life = p.life - damage  
 def \_\_str\_\_(self):  
 state = ""  
 if self.life == 100:  
 state = "无伤"  
 elif self.life >= 70 and self.life < 100:  
 state = "轻伤"  
 elif self.life >= 1 and self.life < 70:  
 state = "重伤"  
 elif self.life <= 0:  
 state = "挂了"  
 return "%s当前的状态为：%s" %(self.name,state)  
class Is(Person):  
 def fire(self, p):  
 damage = 10  
 print("%s向%s开枪，造成了%d伤害" % (self.name, p.name, damage))  
 # if p.life < damage:  
 # p.life = 0  
 # else:  
 # p.life = p.life - damage  
  
def main():  
 h = Hero("【英雄】")  
 is1 = Is("【不要命】")  
 while True:  
 h.fire(is1)  
 is1.fire(h)  
 print(h)  
 print(is1)  
 print()  
 #设置结束条件  
 if h.life <= 0:  
 print("%s死亡，枪战结束" % h.name)  
 break  
 if is1.life <= 0:  
 print("所有恐怖份子全部死亡，枪战结束")  
 break  
main()

### 案例：反恐精英加强版

"""  
演示反恐精英案例——加强版  
对三个匪徒  
分析：  
-----------------------基础版-----------------------  
1.定义人类，描述公共属性 life:100 name:姓名要传参  
2.定义出英雄与恐怖分子类  
3.定义主函数描述枪战过程 main，创建两个对象  
4.定义开枪方法，分成两个方法，Hero Is都有  
 定义的方法要传入被射击的对象  
 被射击对象的生命值要进行减少  
5.主程序中调用开枪操作  
6.开枪操作后，要在主程序中显示每个人的状态信息  
7.定义Person类的\_\_str\_\_方法，用于显示每个人的状态  
8.设置开枪操作为反复操作  
 再设置停止条件：一方生命值<=0  
 停止循环使用break  
-----------------------修复版-----------------------  
9.修复英雄的信息显示模式  
 状态描述 0 - 1- 70 - 99- 100  
 if..elif.. and组合条件  
10.修复生命值为负的问题  
 射击时如果生命值<伤害值，生命值 = 0，否则正常减生命  
-----------------------加强版-----------------------  
11.创建三个恐怖份子对象  
 三个对象都要开枪，三个对象都要打印状态  
12.修复结束条件为三个恐怖份子都死亡  
 三个满足同时死亡 and  
13.解决向三个恐怖份子开枪的问题  
 随机数：random  
 步骤1:使用random import random 必须写在所有程序的前面  
 步骤2：使用random.randint(1,3) 可以产生1到3的随机数  
 产生一个随机数，判断是几就向几号敌人开枪  
  
"""  
  
import random  
class Person:  
 def \_\_init\_\_(self,name):  
 self.name = name  
 self.life = 100  
 def \_\_str\_\_(self):  
 return "%s当前的生命值为：%d" %(self.name,self.life)  
class Hero(Person):  
 def fire(self ,p):  
 damage = 40  
 print("%s向%s开枪，造成了%d伤害" % (self.name,p.name,damage))  
 if p.life < damage:  
 p.life = 0  
 else:  
 p.life = p.life - damage  
 def \_\_str\_\_(self):  
 state = ""  
 if self.life == 100:  
 state = "无伤"  
 elif self.life >= 70 and self.life < 100:  
 state = "轻伤"  
 elif self.life >= 1 and self.life < 70:  
 state = "重伤"  
 elif self.life <= 0:  
 state = "挂了"  
 return "%s当前的状态为：%s" %(self.name,state)  
class Is(Person):  
 def fire(self, p):  
 damage = 10  
 print("%s向%s开枪，造成了%d伤害" % (self.name, p.name, damage))  
 if p.life < damage:  
 p.life = 0  
 else:  
 p.life = p.life - damage  
  
def main():  
 h = Hero("【英雄】")  
 is1 = Is("【不要命】")  
 is2 = Is("【不怕死】")  
 is3 = Is("【还有谁】")  
 while True:  
 # 产生1到3的随机数  
 x = random.randint(1,3)  
 if x == 1:  
 h.fire(is1)  
 elif x == 2:  
 h.fire(is2)  
 else:  
 h.fire(is3)  
 is1.fire(h)  
 is2.fire(h)  
 is3.fire(h)  
 print(h)  
 print(is1)  
 print(is2)  
 print(is3)  
 print()  
 #设置结束条件  
 if h.life <= 0:  
 print("%s死亡，枪战结束" % h.name)  
 break  
 if is1.life <= 0 and is2.life <= 0 and is3.life <= 0:  
 print("所有恐怖份子全部死亡，枪战结束")  
 break  
main()

### 案例：反恐精英超级加强版

"""  
演示反恐精英案例——超级加强版  
对三个匪徒  
加入开枪射击命中概率  
加入开枪伤害值波动  
加入鞭尸文字显示效果  
分析：  
-----------------------基础版-----------------------  
1.定义人类，描述公共属性 life:100 name:姓名要传参  
2.定义出英雄与恐怖分子类  
3.定义主函数描述枪战过程 main，创建两个对象  
4.定义开枪方法，分成两个方法，Hero Is都有  
 定义的方法要传入被射击的对象  
 被射击对象的生命值要进行减少  
5.主程序中调用开枪操作  
6.开枪操作后，要在主程序中显示每个人的状态信息  
7.定义Person类的\_\_str\_\_方法，用于显示每个人的状态  
8.设置开枪操作为反复操作  
 再设置停止条件：一方生命值<=0  
 停止循环使用break  
-----------------------修复版-----------------------  
9.修复英雄的信息显示模式  
 状态描述 0 - 1- 70 - 99- 100  
 if..elif.. and组合条件  
10.修复生命值为负的问题  
 射击时如果生命值<伤害值，生命值 = 0，否则正常减生命  
-----------------------加强版-----------------------  
11.创建三个恐怖份子对象  
 三个对象都要开枪，三个对象都要打印状态  
12.修复结束条件为三个恐怖份子都死亡  
 三个满足同时死亡 and  
13.解决向三个恐怖份子开枪的问题  
 随机数：random  
 步骤1:使用random import random 必须写在所有程序的前面  
 步骤2：使用random.randint(1,3) 可以产生1到3的随机数  
 产生一个随机数，判断是几就向几号敌人开枪  
-----------------------超级加强版-----------------------  
14.加入开枪射击命中概率  
 产生一个随机数，如果在范围内，命中，否则不命中  
 文字效果要变化  
 两处开枪都要重新制作  
15.加入开枪伤害值波动  
 产生一个随机数，作为伤害值  
16.加入鞭尸文字显示效果  
"""  
  
import random  
class Person:  
 def \_\_init\_\_(self,name):  
 self.name = name  
 self.life = 100  
 def \_\_str\_\_(self):  
 return "%s当前的生命值为：%d" %(self.name,self.life)  
class Hero(Person):  
 def fire(self ,p):  
 # 加入命中率  
 hit = random.randint(1,100)  
 if hit > 20 : #命中率80%  
 # 判断当前射击的对象是否是尸体  
 if p.life == 0 :  
 print("%s都死了，就不要鞭尸了" % p.name)  
 else:  
 damage = random.randint(20, 50)  
 print("%s向%s开枪，造成了%d伤害" % (self.name, p.name, damage))  
 if p.life < damage:  
 p.life = 0  
 else:  
 p.life = p.life - damage  
 else:  
 print("枪法真臭，这是个臭籽，没有打到恐怖份子")  
  
 def \_\_str\_\_(self):  
 state = ""  
 if self.life == 100:  
 state = "无伤"  
 elif self.life >= 70 and self.life < 100:  
 state = "轻伤"  
 elif self.life >= 1 and self.life < 70:  
 state = "重伤"  
 elif self.life <= 0:  
 state = "挂了"  
 return "%s当前的状态为：%s" %(self.name,state)  
class Is(Person):  
 def fire(self, p):  
 damage = random.randint(5,15)  
 hit = random.randint(1,100)  
 if hit > 80 :  
 print("%s向%s开枪，造成了%d伤害" % (self.name, p.name, damage))  
 if p.life < damage:  
 p.life = 0  
 else:  
 p.life = p.life - damage  
 else :  
 print("%s枪法不行啊，回去接着练吧" % self.name)  
  
def main():  
 h = Hero("【英雄】")  
 is1 = Is("【不要命】")  
 is2 = Is("【不怕死】")  
 is3 = Is("【还有谁】")  
 while True:  
 # 产生1到3的随机数  
 x = random.randint(1,3)  
 if x == 1:  
 h.fire(is1)  
 elif x == 2:  
 h.fire(is2)  
 else:  
 h.fire(is3)  
 is1.fire(h)  
 is2.fire(h)  
 is3.fire(h)  
 print(h)  
 print(is1)  
 print(is2)  
 print(is3)  
 print()  
 #设置结束条件  
 if h.life <= 0:  
 print("%s死亡，枪战结束" % h.name)  
 break  
 if is1.life <= 0 and is2.life <= 0 and is3.life <= 0:  
 print("所有恐怖份子全部死亡，枪战结束")  
 break  
main()