### 自定义一个Person类，其中有字段age，并提供相应的set get 方法，age业务逻辑上不可以使用<= 0 的数字

需求：如果使用setAge 为age赋值时，如果设置的是非法的数据，那么抛出一个自定义的编译期异常（AgeException），然后测试，调用setAge方法，并进行try except 处理。

Testperson

p1 = Person()  
try:  
 p1.setAge(-3)  
except AgeException:  
 print(**"输入非法！"**)

Person

class Person:  
 \_\_age = None  
 def setAge(self,age):  
 if age <= 0:  
 raise AgeException(**"输入非法！"**)  
 else:  
 self.\_\_age = age  
 def getAge(self):  
 return self.\_\_age

AgeException

class AgeException(Exception):  
 def \_\_init\_\_(self,msg):  
 self.msg = msg

## 使用键盘录入模拟从银行取钱，假设余额是3000元，如果录入的数据大于余额，通过给出提示：金额不足异常。

Test

money = int(input(**"请输入金额："**))  
try:  
 if money > 3000:  
 raise MoneyException(**"余额不足！"**)  
 else:  
 print(**"取钱成功！"**)  
except MoneyException:  
 print(**"金额不足"**)

MoneyException

class MoneyException(Exception):  
 def \_\_init\_\_(self,msg):  
 self.msg = msg

## 下列程序的运行结果？

def getNum(a,b):

try:

c = a + b

return c

except Exception:

return 4

finally:

Return 6

num = getNum(6,9)

print(num)

结果：6