### 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。

class Nianling(Exception):  
 def \_\_init\_\_(self,mag):  
 self.mag = mag  
  
class Person:  
 \_\_age = None  
  
 def setAge(self,age):  
 self.\_\_age = age  
 def getAge(self):  
 return self.\_\_age  
  
 def Nan(self):  
 if self.\_\_age <=0:  
 raise Nianling(**"年龄非法"**)  
 else:  
 print(self.\_\_age)  
  
u=Person()  
u.setAge(0)  
u.Nan()

## 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。

class Yue(Exception):  
 def \_\_init\_\_(self,smg):  
 self.\_\_smg = smg  
  
  
def Qvkuan():  
 p = int(input(**"请输入取款金额"**))  
 i = 3000  
 if p<=i:  
 print(**"取款成功"**,p,**"元"**)  
 else:  
 raise Yue(**"金额不足"**)  
Qvkuan()