### 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 Person:  
# \_\_age = 0  
#  
# def setAge(self, age):  
#  
# self.\_\_age = age  
#  
# def getAge(self):  
# return self.\_\_age  
# def AgeException(\_\_age, type):  
# while True:  
# print("请输入", \_\_age, ":", end="")  
# data = input("")  
# if type == "int":  
# try:  
# return int(data)  
# except Exception:  
# print("输入非法！请重新输入！")  
# elif type == "str":  
# return data  
#  
# name = AgeException("年龄", "int")  
# print(name)*

## 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。

*# def inputa(a):  
#  
# print("请输入",a,":",end="")  
# data = int(input(""))  
# if data <= 300:  
# data = 300 - data  
# print()  
# print("余额为： ")  
# return data  
# elif data > 300:  
# try:  
# raise KeyError("余额不足！！！！")  
# except Exception:  
# print("余额不足！！！")  
#  
#  
#  
#  
# money = inputa("金额")  
# print(money)*