day20：继承，派生，封装

笔记本：	Python基础		
创建时间：	2018/7/5 13:26	**更新时间：**	2018/7/7 15:11
作者：	liuchang_0412@163.com		
URL：	https://blog.csdn.net/u010745324/article/details/53048910		

1.继承(inheritance)和派生(derive)：

①为什么要继承和派生：

继承的目的是延续旧的功能

派生的目的是在旧类的基础上添加新的功能

②作用：

用继承派生机制，可以将一些共有功能加在基类中，实现代码共享，在不改变超类的代码的基础上改变原有功能。

③名词：

基类(base class), 超类(super class), 父类(father class)

派生类(derived class), 子类(child class)

2.单继承：

①语法：

```
class 类名(超类名):
    pass
```

②示例：

human.py

③继承说明：

任何类都直接或间接的继承自object类

object类是一切类的超类

④__base__属性

用来记录此类的基类

3.覆盖 override（也叫重写 overwrite）

①什么是覆盖：

覆盖式指有继承关系的类中，子类中实现了与基类（超类）同名的方法，在子类实例调用该方法时，实际调用的是子类中的覆盖版本，这种现象叫做覆盖。

②示例：

human_override.py

③子类对象显示调用基类方法的方式：

基类名.方法名(实例, 参数)

4.super函数

定义：

super(type, obj)　返回绑定超类的实例(要求obj必须为type类型的实例)

super( )　返回绑定的超类的实例，等同于(class, 实例方法的第一个参数)，
此方法必须用在方法内部

作用：

返回绑定超类的实例，用超类的实例来调用其自身的方法

示例：

见super.py

5.用于类的函数：

issubclass(cls, class 或 类元组)：判断一个类是否继承自其他的类

示例：

```
class A:
    pass
class B(A):
    pass
class C(B):
    pass
class D(B):
    pass

issubclass(C, A)    #True
issubclass(C, B)    #True
issubclass(A, C)    #False
issubclass(C, D)    #False
```

6.显示调用基类的构造方法：

```
class Human:
    def __init__(self, name, age):
        self.name = name
        self.age = age
    ...

class Student(Human):
    def __init__(self, name, age, score):
        super().__init__(name, age)
        self.score = score
```

7.封装 enclosure

定义：

封装是指隐藏类的实现细节，让使用者不用关心这些细节

封装的目的是让使用者尽可能少的使用实例变量（属性）进行操作

私有属性：
python 类中，以双下划线'__'开头，不以双下划綫结尾的标识符为私有成员
在类的外部无法直接访问
示例：
enclosure.py

8.多态 polymorphic
定义：字面意思："多种状态"
多态是指在集成/派生关系的类中，调用基类对象的方法
实际能调用子类的覆盖版本方法的现象较多态
说明：
多态调用的方法与对象相关，不予类型相关
Python的全部对象都只有"运行时状态（动态）"，没有"C++/Java"里的编译时状态(静态)
示例：
poly.py

9.面向对象的编程语言的特征：
（继承，封装，多态)
C++ / Java / Python / Swift / C#

10.多继承 multiple inheritance
定义：
多继承是指一个子类继承自两个或两个以上的基类
语法：
class 类名(基类名1, 基类名2, ...):
pass
说明：
1.一个子类同时继承自多个父类，父类中的方法可以同时被继承下来
2.如果两个父类中有同名的方法，而在子类中有没有覆盖此方法时，调用结果难以确定
多继承的缺陷：
标识符（名字空间冲突的问题）
要谨慎使用多继承
示例见：
multi_inherit_bug.py

11.继承的MRO(Method Resolution Order)问题：
类内的__mro__属性用来记录集成方法的查询顺序

#练习：

已知list 列表类中没有inser_head方法，写一个自定义的类MyList,继承自list类，
在MyList类中添加

```
class MyList(list):
    def inset_head(self, value):
        """以下自己实现，将value插入到列表的开始处"""
```

如：

```
L = MyList(range(1,5))
print(L)    #[1, 2, 3, 4]
L.inset_head(0)
print(L)    #[0, 1, 2, 3, 4]
```

12.PEP8编码规范：

①代码编排：

使用4空格缩进，不使用Tab，更不允许用Tab和空格混合缩进

每行最大长度最大79字节，超过部分使用反斜杠折行

类和全局函数定义间隔两个空行，类内方法定义间隔一个空行，其他地方可以不加空行

②文档编排：

其中import部分，又按标准，三方和自己编写的顺序依次排放，之间空一行

不要在一句import中导入多个模块，比如不推荐import os, sys

尽可能用import XX 而不采用 from XX import YY引用库，因为可能出现名字冲突

③空格的使用：

各种右括号前不加空格

逗号，冒号，分号前不加空格

函数的左括号前不加空格

序列的左括号前不加空格

操作符左右各加一个空格，不要为了对齐增加空格

函数默认参数使用的赋值符的左右省略空格

不要将多条语句写在一行，即便有;

if/for/while语句中，即使执行语句只有一条，也必须另起一行

##################################

##项目练习:2048小游戏开发

##################################

1.项目说明：

①2048游戏的地图是4*4的方块矩阵，开始是4*4的零矩阵。游戏开始在任意地方出现2或4，以后每次出现的数字都是2或者4。然后我们可以上下左右移动，移动的规则是例如向左动，某一行的数比如是[2,4,0,2]向左移动,移动后变成[2,4,2,0],移动后不允许两个非0数字之间有0的存在。移动前相邻两个数相同的话会合并，例如[2,2,4,4]会合并成[4,8,0,0]。

②移动合并完后，会在所有为0的位置随机挑选出一个位置填上2或者4，先移动合并完后才会随机填上2或者4。

③有些时候在某个方向无法移动，在无法移动(即移动后还是老样子的情况下)不会在随机0位置处添加随机数2或4。只有移动后改变了矩阵的原来样子且矩阵最小值为0才会在随机0位置出添加随机数2或4。

④有两种情况移动后不会添加随机数2或4，第一种情况是上面这种情况，往一个方向移动没有效果。另外一种是矩阵都为非0数，没有位置添加随机数2或4。

⑤游戏结束的情况，当矩阵没有0而且每行每列任意两个相邻数无法合并。

问题：

1.如何添加try/except

2.如何把程序里临时变量L删除

3.如何用继承的方法改写程序

4.tkinter的使用