一 搬家具

1.1 需求

将小于房子面积的家具搬到房间中

分析: 涉及到 房子和家具

- 房子类
- 实例属性
  - 房子的大小
  - 房子的剩余面积
  - 房子里面的家具
- 实例方法
  - 将家具搬到房子里面去
- 显示房子的状态和信息

- 家具类
- 实例属性
  - 家具的名字
  - 家具的面积

```
class Furniture():
    """家具类"""
    def __init__(self, name, area):  # 在超市买来就有的数据
        # 名字
        self.name = name
        # 大小
        self.area = area

class Home():
    """房子类"""
    def __init__(self, area):
        # 大小
        self.area = area  # 本来房子就是100平
        # 剩余大小
        self.free_area = area
        # 家具
        self.furniture = []  # 搬来的家具 我们也要知道有哪些家具


    def add_furniture(self, item):
        """
        搬家具
        :param item:    形参,    用来承接家具对象
        :return:
        """
        if item.area < self.free_area:
            self.furniture.append(item.name)
            # 家具搬入之后,  更新房子的剩余面积
```

```
            self.free_area -= item.area
        else:
            print('家具太大，放不下')

    def __str__(self):
        return f'房子的大小{self.area},剩余面积为{self.free_area},搬进的家具有
{self.furniture}'


# 床
bed = Furniture('床',10)
bed1 = Furniture('床1',20)

# 房子
home1 = Home(100)
home1.add_furniture(bed)
home1.add_furniture(bed1)
print(home1)
```

面向对象-继承

目标

- 继承的概念
- 单继承
- 多继承
- 子类重写父类的同名方法和属性
- 子类调用父类的同名方法和属性
- 多层继承
- super()
- 私有属性和私有方法

思考： 类里面还能不能写类

二 继承的概念

生活中，后辈继承父辈的财产

python中面向对象中的继承指的是多个类之间的从属关系，即子类默认继承父类的所有方法和属性

- 经典类

  不有任何内置类型派生的类，称之为经典类（旧事类）

```
class 类名:
    代码
    。。。。。
```

- 新式类

```
class 类名(object):
    代码
```

在python中，所有类默认继承自object类，object类是顶级类或者是基类，其他子类就叫派生类

三 单继承

雪中悍刀行　剑神李淳罡 剑术高超 想找徒弟传承剑术 找到徐凤年收为徒弟

```
# 定义师父类
class Master(object):
    def __init__(self):
        self.kongfu = '一剑仙人跪'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')

# 定义徒弟类
class Prentice(Master):
    pass


# 用徒弟类创建对象，调用师父的属性和方法
xiaoxu = Prentice()
print(xiaoxu.kongfu)
xiaoxu.battle()
```

总结:

- 子类在继承的时候，在定义类的时候，小括号中写的就是父类的名字
- 父类的方法和属性都会被子类所继承

四 多继承

思考:　子类可不可以继承多个父类

侍女青鸟　霸王卸甲　现在徐凤年也想学霸王卸甲

也就意味着有两个师父

所谓的多继承就是指一个类同时继承多个父类

```
# 定义师父类
class Master(object):
    def __init__(self):
        self.kongfu = '一剑仙人跪'
        self.name = '李淳罡'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')

# 定义侍女类
class Maid(object):
    def __init__(self):
```

```
        self.kongfu = '霸王卸甲'
        self.name = '青鸟'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')


    def fn(self):
        print('aaaaaaaaaa')

# 定义徒弟类
class Prentice(Master, Maid):
    pass


# 用徒弟类创建对象，调用师父的属性和方法
xiaoxu = Prentice()
print(xiaoxu.kongfu)
xiaoxu.battle()
xiaoxu.fn()


print(Prentice.__mro__)
```

注意：

- 多继承可以继承多个父类，也继承了多个父类的属性和方法
- 注意：如果多个父类中有同名的方法和属性，则默认使用第一个父类的属性和方法(根据类中的魔法属性mro的顺序来查找)
- 多个父类中，不同的属性和方法，不会有任何影响

五 子类重写父类的同名方法和属性

当小徐学会了师父的招式之后，自己钻研除了新功夫，剑气滚龙壁

```
# 定义师父类
class Master(object):
    def __init__(self):
        self.kongfu = '一剑仙人跪'
        self.name = '李淳罡'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')

# 定义侍女类
class Maid(object):
    def __init__(self):
        self.kongfu = '霸王卸甲'
        self.name = '青鸟'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')
```

```python
# 定义徒弟类
class Prentice(Master, Maid):
    def __init__(self):
        self.kongfu = '剑气滚龙壁'

    def battle(self):
        print(f'运用了{self.kongfu}和敌人battle')


# 用徒弟类创建对象，调用师父的属性和方法
xiaoxu = Prentice()
print(xiaoxu.kongfu)
# print(xiaoxu.name)
xiaoxu.battle()

print(Prentice.__mro__)
```

六 子类调用父类的同名方法和属性

```
```