实验环境	仅供参考 严禁抄袭 Blog:zhangshier.vip

一、 实验题目

实验四：面向对象程序设计

二、 实验目的

（1） 理解类与对象的概念，掌握类的定义和使用方法。
（2） 熟悉类的成员和方法的类型，掌握其定义和使用方法。
（3） 掌握类的属性及其使用方法。
（4） 掌握派生类的创建和使用方法。
（5） 理解类的多态含义，掌握类的多态实现方法。
（6） 掌握抽象类和抽象方法的使用方法。

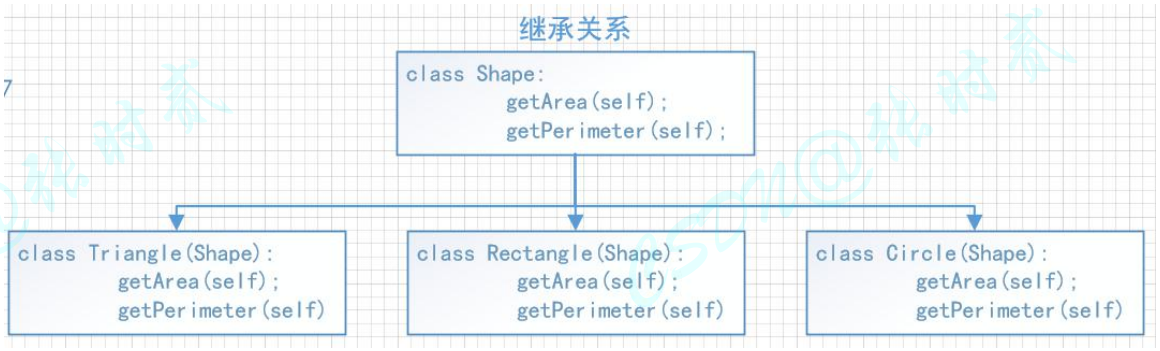
三、 实验内容

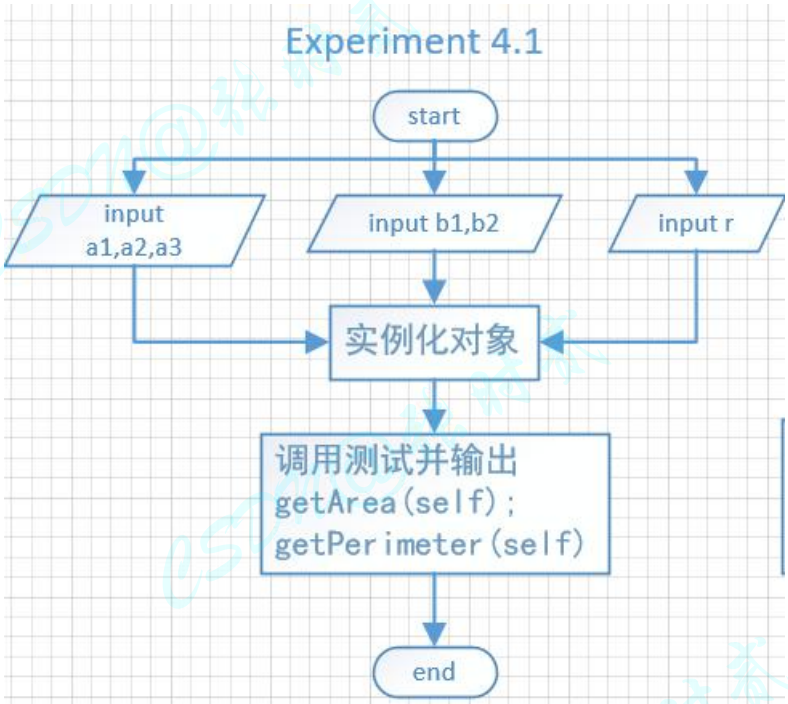
1、 编程实现如下功能：
（1） 定义一个抽象类 Shape，在抽象类 Shape 中定义求面积 getArea()和周长 getPerimeter()的抽象方法。
（2） 分别定义继承抽象类 Shape 的 3 个子类即 Triangle、Rectangle 和 Circle，在这 3 个子类中重写 Shape 中的方法 getArea()和 getPerimeter()。
（3） 创建类 Triangle、Rectangle、Circle 的对象，对 3 个类中的方法进行调用测试。

2、 设计一个“超市进销存管理系统”，要求如下：
（1） 系统包括 7 中操作，分别是：1.查询所有商品；2.添加商品；3.修改商品；4.删除商品；5.卖出商品；6.汇总；-1.退出系统。
（2） 选择操作序号“1”，显示所有商品。
（3） 选择操作序号“2”，添加新的商品（包括商品名称、数量和进货价格）。
（4） 选择操作序号“3”，修改商品。
（5） 选择操作序号“4”，删除商品。
（6） 选择操作序号“5”，卖出商品（包括商品名称、数量和售出价格）。
（7） 选个操作序号“6”，汇总当天卖出商品，包括每种销售商品名称、数量、进货总价、销售总价等。
（8） 选择操作序号“-1”，退出系统。

四、 实验步骤

1、 题目一
(1) 问题分析（含解决思路、使用的数据结构、程序流程图等）
设计抽象类 Shape，在 Shape 中定义抽象方法 getArea()和周长 getPerimeter()，之后让 Triangle、Rectangle、Circle 分别继承 Shape 类并重写函数，其中 Triangle 类输入的三边需要额外判断是否满足两边之和永远大于第三边。设计好四个类之后，在主函数中新建对象并调用计算周长和面积





(2) 算法/代码描述（基本要求源代码）

```
import abc
import math


# 抽象类，用 abc 库，只需要声明函数，不需要写具体功能，不能够实例化
class Shape ( metaclass=abc.ABCMeta ):
    # 面积
    @abc.abstractmethod
    def getArea(self):
        pass

    # 周长
    @abc.abstractmethod
    def getPerimeter(self):
        pass
```

```
# 三角形
class Triangle ( Shape ):
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c



    def getArea(self):
        return 0.25*math.sqrt ( (self.a + self.b + self.c)*(self.a + self.b -
self.c)*(self.b + self.c - self.a)*(
                    self.a + self.c - self.b) )

    def getPerimeter(self):
        return self.a + self.b + self.c

    # 判断三边关系
    def judgeInput(self):
        a1 = self.a + self.b - self.c
        a2 = (self.a - self.b) - self.c
        b1 = self.b + self.c - self.a
        b2 = (self.b - self.c) - self.a
        c1 = self.a + self.c - self.b
        c2 = (self.a - self.c) - self.b
        if (a1 > 0 and a2 < 0) and (b1 > 0 and b2 < 0) and (c1 > 0 and c2 < 0):
            return True
        else:
            return False


# 矩形
class Rectangle ( Shape ):
    def __init__(self, a, b):
        self.a = a
        self.b = b

    def getArea(self):
        return self.a*self.b

    def getPerimeter(self):
        return (self.a + self.b)*2
```

```
# 圆
class Circle ( Shape ):
    def __init__(self, r):
        self.r = r

    def getArea(self):
        return 3.14*self.r ** 2

    def getPerimeter(self):
        return 2*math.pi*self.r


if __name__ == '__main__':
    a1, a2, a3 = map ( int, input ( "请输入三角形三边:" ).split ( " " ) )
    t1 = Triangle ( a1, a2, a3 )
    if t1.judgeInput():
        print ( f"三角形面积={t1.getArea ()}" )
        print ( f"三角形周长={t1.getPerimeter ()}" )
    else:
        print ( "输入的三边长不能构成三角形" )
    b1, b2 = map ( int, input ( "请输入矩形两边:" ).split ( " " ) )
    r1 = Rectangle ( b1, b2 )
    print ( f"矩形面积={r1.getArea ()}" )
    print ( f"矩形周长={r1.getPerimeter ()}" )
    r1 = int(input ( "请输入圆的半径 r1:" ))
    c1 = Circle ( r1 )
    print ( "圆面积%.2f"%c1.getArea () )
    print ( "圆周长%.2f"%c1.getPerimeter () )
```

(3) 运行结果（含执行结果验证、输出显示信息）

输入错误提前判断

```
"D:\太工\大三第二学期\Python\192062116 张帆\
请输入三角形三边:1 2 3
输入的三边长不能构成三角形
```

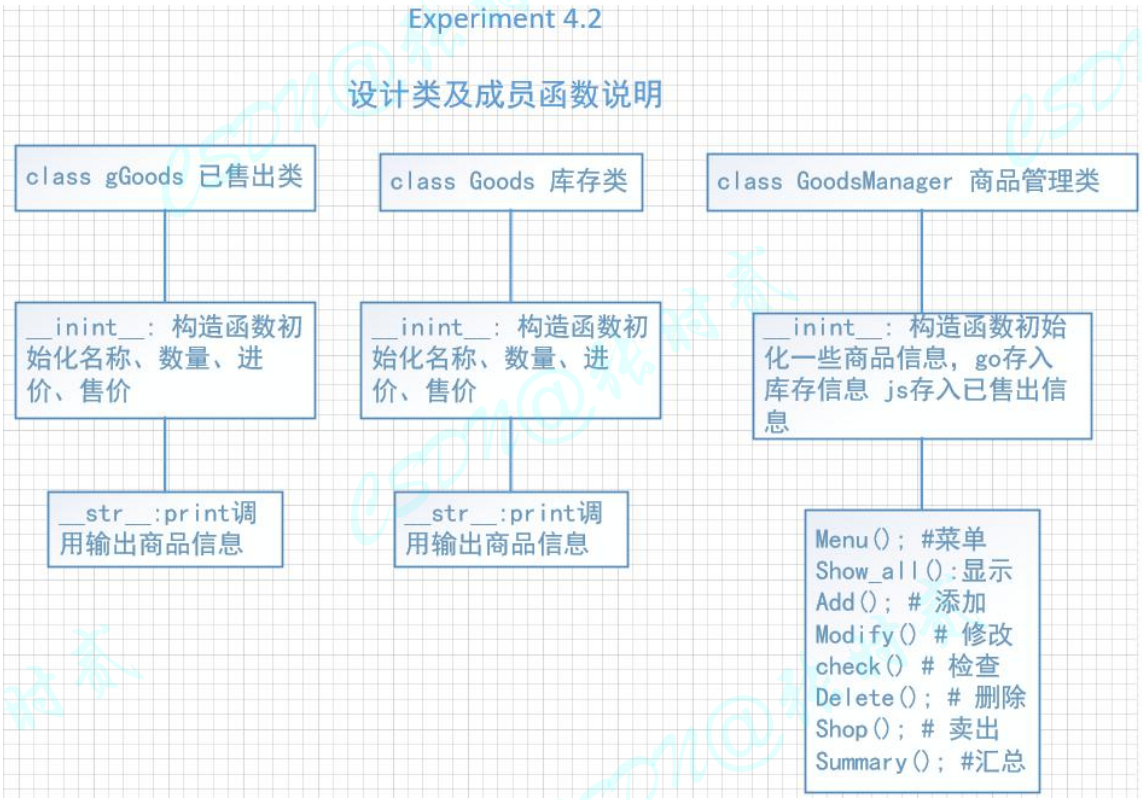
输入正确输出

```
"D:\太工\大三第二学期\Python\192062116 张帆\
请输入三角形三边:3 4 5
三角形面积=6.0
三角形周长=12
请输入矩形两边:2 2
矩形面积=4
矩形周长=8
请输入圆的半径r1:2
圆面积12.56
圆周长12.57
```

2、 题目二

(1) 问题分析（含解决思路、使用的数据结构、程序流程图等）

分别设计三个类，已售出类，库存类，管理商品类。在管理商品类中通过两个列表分别保存已售出和库存信息。之后通过 Menu 菜单调用要求的 7 个功能，其中修改、删除、卖出功能调用时，需要一个先调用 check()函数判断是否存在该商品



(2) 算法/代码描述（基本要求源代码）

```
# 定义库存类
class Goods:
```

```
        # 有参构造初始化
        def __init__(self, name, num, cin, cout):
            self.name = name
            self.num = num
            self.cin = cin
            self.cout = cout

        def __str__(self):
            state = "已售罄"
            if self.num == 0:
                return '名称：%s , 数量：%d %s, 进货价格：%.2f ,售出价格：%.2f
'%(self.name, self.num, state, self.cin, self.cout)
            else:
                return '名称：%s , 数量：%d , 进货价格：%.2f ,售出价格：%.2f
'%(self.name, self.num, self.cin, self.cout)


    # 已售出类
    class gGoods:
        def __init__(self, name, gnum, gcin, gcout):
            self.name = name
            self.gnum = gnum
            self.gcin = gcin
            self.gcout = gcout

        def __str__(self):
            return '名称：%s , 卖出数量：%d , 进货价格：%.2f ,卖出价格：%.2f
'%(self.name, self.gnum, self.gcin, self.gcout)


    # 定义管理商品类
    class GoodsManager:
        go = [ ]   # 库存
        js = [ ]   # 售出

        # 构造方法，初始化加三个商品
        def init(self):
            self.go.append ( Goods ( '牛奶', 5, 40, 60 ) )
            self.go.append ( Goods ( '盒饭', 5, 10, 60 ) )
            self.js.append ( gGoods ( '面条', 1, 30, 60 ) )

        # 菜单
        def Menu(self):
            self.init ()
```

```
        print ( '\"超市进销存管理系统\"菜单：' )
        print ( "1.显示所有商品" )
        print ( "2.添加新的商品" )
        print ( "3.修改商品信息" )
        print ( "4.删除商品" )
        print ( "5.卖出商品" )
        print ( "6.汇总" )
        print ( "-1.退出" )
        print ( "**********************************" )
        while True:
            SN = int ( input ( "===请输入操作序号：" ) )
            if SN in [ -1, 1, 2, 3, 4, 5, 6 ]:
                if SN == -1:
                    print ( "已经退出" )
                    break;
                if SN == 1:
                    self.Show_all ()
                elif SN == 2:
                    self.Add ()
                elif SN == 3:
                    self.Modify ()
                elif SN == 4:
                    self.Delete ()
                elif SN == 5:
                    self.Shop ()
                elif SN == 6:
                    self.Summary ()
            else:
                print ( "输入有误！" )

    # 显示
    def Show_all(self):
        for goods in self.go:
            print ( str ( goods ) )

    # 添加
    def Add(self):
        goods_name = input ( "请输入商品名称：" )
        ret = self.check ( goods_name )
        if ret != None:
            print ( '商品已经存在' )
            print ( '是否增加商品数量：（y/n）' )
            while True:
                pd = input ()
```

```
                    if pd == 'y':
                        goods_num = int ( input ( "请输入商品的数量：" ) )
                        old_goods = Goods ( goods_name, goods_num +
ret.num, ret.cin, ret.cout )
                        self.go.remove ( ret )
                        self.go.append ( old_goods )
                        print ( "增加成功" )
                        break
                    elif pd == 'n':
                        print ( "已经返回" )
                        break
                    else:
                        print ( "输入有误，重新输入：" )

            else:
                goods_num = int ( input ( "请输入商品的数量：" ) )
                goods_cin = float ( input ( "请输入商品进货价格：" ) )
                goods_cout = float ( input ( "请输入商品出货价格：" ) )
                if goods_num > 0 and goods_cin > 0 and goods_cout > 0:
                    new_goods = Goods ( goods_name, goods_num, goods_cin,
goods_cout )
                    self.go.append ( new_goods )
                    print ( "添加成功" )
                else:
                    print ( "输入错误！" )

        # 修改
        def Modify(self):
            goods_name = input ( "请输入需要修改的商品名称：" )
            ret = self.check ( goods_name )
            if ret != None:
                print ( ret )
                goods_name1 = input ( "请输入修改后商品的名称：" )
                goods_num = int ( input ( "请输入修改后商品的数量：" ) )
                goods_cin = float ( input ( "请输入修改后商品进货价格：" ) )
                goods_cout = float ( input ( "请输入修改后商品出货价格：" ) )
                old_goods = Goods ( goods_name1, goods_num, goods_cin,
goods_cout )
                self.go.remove ( ret )
                self.go.append ( old_goods )
                print ( "修改成功" )
            else:
                print ( "没有此商品!" )
```

```
    # 检查，修改删除卖出之前先调用检查是否存在商品
    def check(self, goods_name):
        for goods in self.go:
            if goods.name == goods_name:
                return goods
        else:
            return None

    def checkjs(self, goods_name):
        for goods in self.js:
            if goods.name == goods_name:
                return goods
        else:
            return None

    # 删除
    def Delete(self):
        goods_name = input ( "请输入需要删除的商品名称：" )
        ret = self.check ( goods_name )
        if ret != None:
            print ( ret )
            print ( '是否删除商品：（y/n）' )
            while True:

                pd = input ()
                if pd == 'y':
                    self.go.remove ( ret )
                    print ( "删除成功" )
                    break
                elif pd == 'n':
                    print ( "已经返回" )
                    break
                else:
                    print ( "输入有误，重新输入：" )
        else:
            print ( "没有此商品！" )

    # 卖出
    def Shop(self):
        goods_name = input ( "请输入需要卖出的商品名称：" )
        ret = self.check ( goods_name )
        if ret != None:
            g_num = int ( input ( "卖出个数:" ) )
            if ret.num - g_num < 0:
```

```
                    print ( "该商品数量不足！请补充" )
                else:
                    old_goods = Goods ( ret.name, ret.num - g_num, ret.cin,
ret.cout )
                    self.go.remove ( ret )
                    self.go.append ( old_goods )
                    gret = self.checkjs ( goods_name )
                    if gret == None:
                        shop_goods = gGoods ( ret.name, g_num,
ret.cin*g_num, ret.cout*g_num )
                        self.js.append ( shop_goods )
                    else:
                        shop_goods = gGoods ( gret.name, g_num + gret.gnum,
gret.gcin + ret.cin*g_num,
                                              gret.gcout + ret.cout*g_num )
                        self.js.remove ( gret )
                        self.js.append ( shop_goods )
                    print ( "卖出后：", end=' ' )
                    old_goods = Goods ( ret.name, ret.num - g_num,
ret.cin*g_num, ret.cout*g_num )
                    print ( old_goods )
            else:
                print ( "没有此商品！" )

    # 汇总当天卖出商品，包括每种销售商品名称、数量、进货总价、销售
总价等。
    def Summary(self):

        for goods in self.js:
            print (   goods   )
        print ( "售出的物品进货总价：", end="" )
        x = 0
        for goods in self.js:
            x += float ( goods.gcin )

        print ( "售出的物品销售总价：", end="" )
        y = 0
        for goods in self.js:
            y += float ( goods.gcout )
        print ( y )
        print ( "利润：", y - x )


if __name__ == '__main__':
```

```
manager = GoodsManager ()
manager.Menu ()
```

(3) 运行结果（含执行结果验证、输出显示信息）

菜单

```
"D:\太工\大三第二学期\Python\192062116 张帆\venv\Scripts
"超市进销存管理系统"菜单：
1.显示所有商品
2.添加新的商品
3.修改商品信息
4.删除商品
5.卖出商品
6.汇总
-1.退出
```

修改、删除、卖出操作时会先检查是否有当前商品

```
请输入操作序号：3
请输入需要修改的商品名称：面包
没有此商品！
```

显示所有商品

```
请输入操作序号：1
名称：牛奶 ，数量：5 ， 进货价格：40.00 ,售出价格：60.00
名称：盒饭 ，数量：5 ， 进货价格：10.00 ,售出价格：60.00
```

新增商品

```
请输入操作序号：2
请输入商品名称：牛奶面包
请输入商品的数量：10
请输入商品进货价格：2
请输入商品出货价格：20
添加成功
```

删除商品

```
请输入需要删除的商品名称：面包
名称：面包 ，数量：10 ， 进货价格：10.00 ,售出价格：10.00
是否删除商品：（y/n）
y
```

卖出商品，数量不足

```
请输入操作序号：5
请输入需要卖出的商品名称：牛奶
卖出个数:50
该商品数量不足！请补充
```

卖出商品，数量满足

```
请输入操作序号：5
请输入需要卖出的商品名称：牛奶
卖出个数:1
卖出后：  名称：牛奶 ， 数量：4 ， 进货价格：40.00 ,售出价格：60.00
```

汇总已售商品

```
请输入操作序号：6
名称：面条 ， 卖出数量：1 ， 进货价格：30.00 ,卖出价格：60.00
名称：牛奶 ， 卖出数量：1 ， 进货价格：40.00 ,卖出价格：60.00
售出的物品进货总价：售出的物品销售总价：120.0
利润：  50.0
```

五、 出现的问题及解决的方法

学到了一个新的库，abc 库声明父类 Shape 为抽象类，不需要实现具体功能，之后通过子类对函数重写