**暨南大学本科实验报告专用纸**

课程名称 算法分析与设计实验 成绩评定

实验项目名称 0-1背包问题 指导教师 李展

实验项目编号 实验十 实验项目类型 综合性 实验地点N504 学生姓名 张印祺 学号 2018051948

学院 信息科学技术 系 计算机科学 专业 网络工程

实验时间 2019 年 4 月 16 日

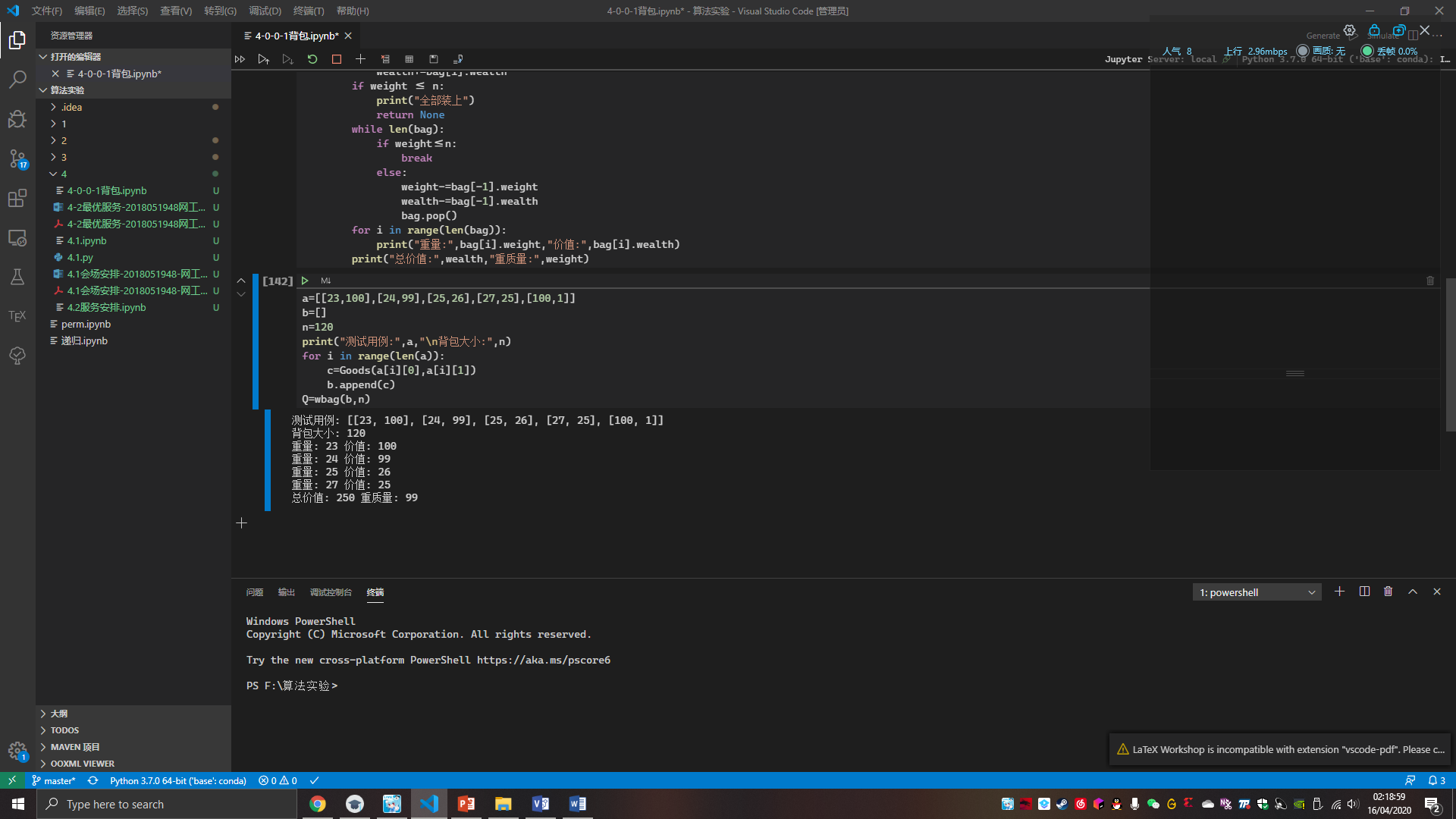
1. 问题描述

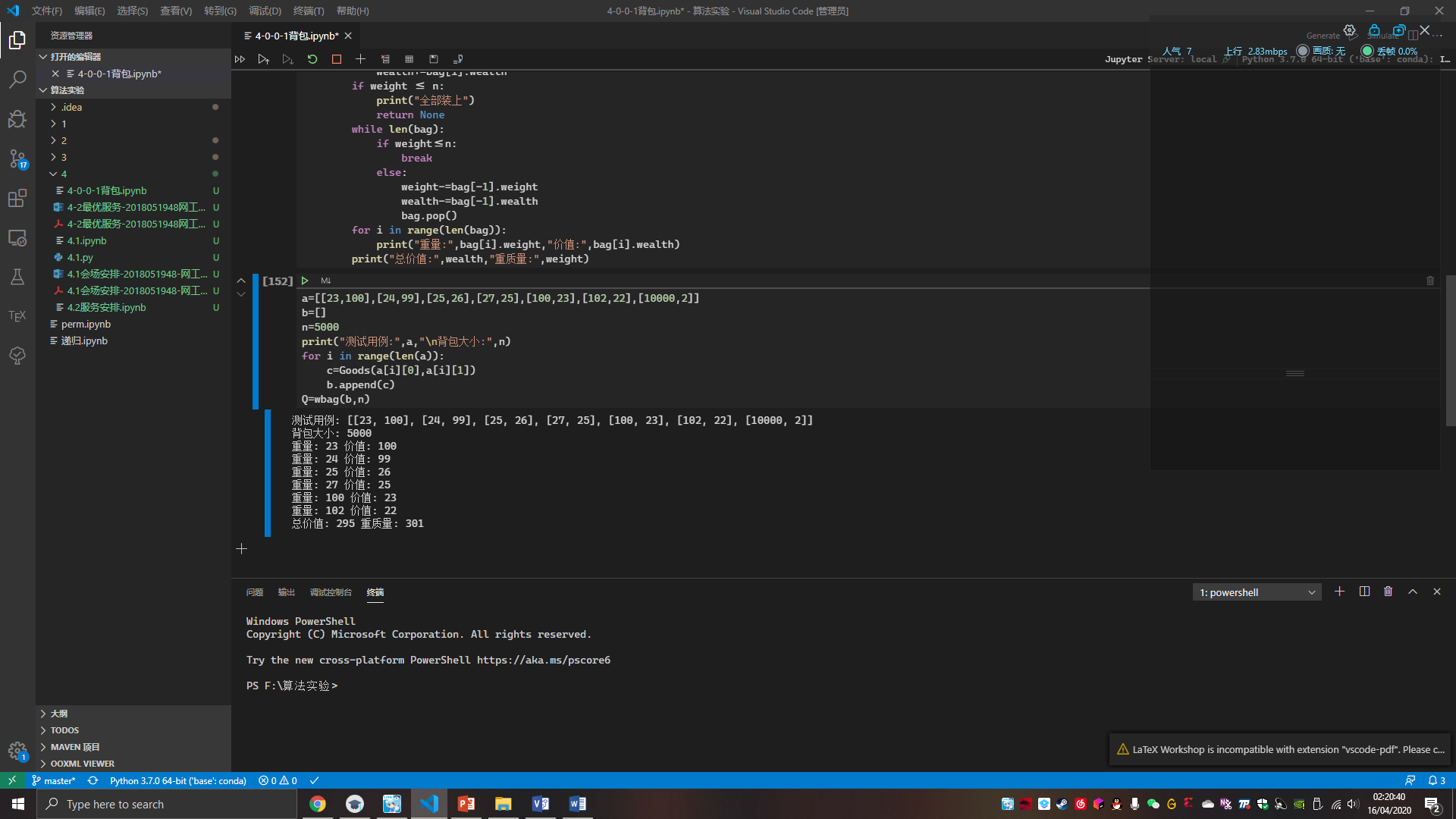
若在0-1背包问题中，各物品依重量递增排列时其价值恰好依递件排列。对这个特殊的0-1背包问题，设计一个有效算法找出最优解，并说明算法的正确性

1. 算法思路
2. 将货物按价值降序进行排序，用一个栈作为背包；
3. 将货物按价值大小装入背包了，直到背包容量为负；
4. pop装入的最后一件货物。
5. 流程图



1. 测试结果





1. 实验总结

本方法时间复杂度：

1. 进行排序操作
2. 进行循环操作将货物push到bag中，最后pop溢出件

总时间复杂度

最优解证明：

我们通过逆向思维去考虑这道题：

1. 我们将所有的货物全部装入，将问题转变为扔掉x件后总价值最优且可以装入背包
2. 我们一定是首先丢掉价值最低最重的货物
3. 我们先考虑丢去一件的问题：

如果我们首先丢掉小件的话，由题设可得，小件高价值，我们丢掉的小件一定可以找到一件价值大于它且质量大于它的物品，即：我们丢去a[1]后满足条件，那么我们丢去一个价值小于a[1]，质量大于a[1]的a[2]也一定是满足背包容量且价值更优。将此向后推导可以得知：当n>m如果丢弃a[m]满足条件那么丢弃a[n]一定满足且更优。

1. 丢去1件物品其实就是丢去n件物品的子问题

综上我们可以得知，在这类特殊的背包问题中，我们将其装满后首先丢弃最重/最低价的物品，也就是我们在装入背包时按价值降序/质量升序装入。

1. 附录 （程序代码）

class Goods:

    def \_\_init\_\_(self,weight:float,wealth:float):

        self.weight=weight

        self.wealth=wealth

class wbag:

    def \_\_init\_\_(self,a:list,n:int):

        a.sort(key =lambda x:x.wealth,reverse=True)

        wealth,weight=0,0

        self.Assign(a,n,weight,wealth)

    def Assign(self,a:list,n:float,weight:float,wealth:float):

        bag=[]

        for i in range (len(a)):

            bag.append(a[i])

            n-=a[i].weight

            if n < 0:

                n+=bag[-1].weight

                bag.pop()

                break

        for i in range(len(bag)):

            print("重量:",bag[i].weight,"价值:",bag[i].wealth)

            weight+=bag[i].weight

            wealth+=bag[i].wealth

        print("总价值:",wealth,"重质量:",weight)

x=eval(input("请输入货物总数:"))

a=[]

for i in range(x):

    weight=eval(input("请输入货物质量:"))

    wealth=eval(input("请输入货物价值:"))

    b=Goods(weight,wealth)

    a.append(b)

n=eval(input("请输入背包大小:"))

Q=wbag(a,n)

**暨南大学本科实验报告专用纸(附页)**