python蓝桥杯基础知识总结：

1.模块方面

1.1statistics

```
import statistics
list0=[2,3,2,3,5,6,7,8]
print(statistics.mean(list0))
print(statistics.median(list0))
print(statistics.median_low(list0))
print(statistics.median_high(list0))
print(statistics.mode(list0))
print(statistics.pvariance(list0))
#1、statistics.mean()  求算术平均值
#2、statistics.median() 计算数据的中位数，如果有两个中位数，则返回其平均值
#    statistics.median_low() 数据中的低中位数
#    statistics.median_high() 数据中的高中位数
# 3、statistics.mode()  计算众数
# 4、statistics.pvariance() 计算数据的总体方差
```

1.2collections

```
import collections
q = collections.deque([1, 2, 3, 4])
q.rotate(1)
print(q)  # [4, 1, 2, 3]
q.rotate(1)
print(q)  # [3, 4, 1, 2]
##############################
from collections import deque
q=deque([1,2,3,4])
temp=[2,5,6]
q.extendleft(temp)
print(list(q))
# deque([6, 5, 2, 1, 2, 3, 4])
# [6, 5, 2, 1, 2, 3, 4]
```

1.3calendar

```
import calendar
print(calendar.isleap(2022))
print(calendar.leapdays(2000,2022))
# False
# 6
```

1.4datetime

```
import datetime
bt = datetime.date(2000,11,6)
print(bt)
# 2000-11-06
a = datetime.timedelta(days=100)
datetime.timedelta(days=100) #weeks / hours
b = a + bt
datetime.date(2001, 2, 14)

# bt.weekday()：返回weekday，如果是星期一，返回0；如果是星期2，返回1，以此类推；
# bt.isoweekday()：返回weekday，如果是星期一，返回1；如果是星期2，返回2，以此类推；
```

1.5math

```
# math.ceil(x)：返回不小于 x 的最小整数。
# math.floor(x)：返回不大于 x 的最大整数。
# math.trunc(x)：返回 x 的整数部分。
# math.modf(x)：返回 x 的小数部分和整数部分，以元组形式返回。
# math.fabs(x)：返回 x 的绝对值。
# math.factorial(x)：返回 x 的阶乘。
# math.gcd(a, b)：返回 a 和 b 的最大公约数。
```

1.6itertools

```
import itertools
   # 1)product
for i in itertools.product('ab', 'cd'):
    print(i)
# >>>('a', 'c')
#    ('a', 'd')
#    ('b', 'c')
#    ('b', 'd')

  # 2)permutations
    list(itertools.permutations('ABC'))
    # >>>[('A', 'B', 'C'), ('A', 'C', 'B'), ('B', 'A', 'C'), ('B', 'C', 'A'),
('C', 'A', 'B'), ('C', 'B', 'A')]
    list(itertools.permutations('ABC', 2))
    # >>>[('A', 'B'), ('A', 'C'), ('B', 'A'), ('B', 'C'), ('C', 'A'), ('C', 'B')]

    # 3)combinations
    list(itertools.combinations("ABC",2))
    # >>>[('A', 'B'), ('A', 'C'), ('B', 'C')]

    # 4)accumulate 可指定函数，默认为求和
    list(itertools.accumulate([0,1,0,1,1,2,3,5]))
    # >>>[0, 1, 1, 2, 3, 5, 8, 13]
    list(itertools.accumulate([0,1,0,1,1,2,3,5],max))
    # >>>[0, 1, 1, 1, 1, 2, 3, 5]
```

1.7bisect

```
# bisect_left(a, x, lo=0, hi=len(a))
# bisect_right(a, x, lo=0, hi=len(a))
# insort_left(a, x, lo=0, hi=len(a))
# insort_right(a, x, lo=0, hi=len(a))
```

2.简单知识点

2.1functions

```
# 返回单个字母的特定值，可以先返回其Unicode码
# ord("a")
# >> > 97

# 千位分隔符
# print({:,}.format(n).replace(",","."))
# n = 123456
# >>>123.456

# 求a和b最大公约数
#   gcd(a,b)

#  二进制
# bin() 返回一个整数 int 或者长整数 long int 的二进制表示。

# 二进制转十进制 int(“二进制字符串”,2)
# 十进制转二进制 bin(整数)[2:]

# 幂函数
# pow(a,b)

# enumerate() 函数
# 用于将一个可遍历的数据对象(如列表、元组或字符串)组合为一个索引序列，
# 同时列出数据和数据下标，一般用在 for 循环当中。
# max()和min()，字符串是可以比较的，
# 按照ascII值排 abb， aba，abac，最大为abb，最小为aba
```

2.2set

```
# 并：|  交：  &   差：  -  对称差集：  ^

# 返回一个新集合，该集合的元素既包含在集合 x 又包含在集合 y 中
x=set([2,5,6,3,4])
y=set([2,6,9,8,5])
print(x.intersection(y))
# {2, 5, 6}

# symmetric_difference(other_set) 或 ^：返回两个集合的对称差集（即只属于其中一个集合的元素）。
s1 = {1, 2, 3}
```

```
s2 = {3, 4, 5}
s3 = s1.symmetric_difference(s2)
print(s3)  # 输出：{1, 2, 4, 5}


# discard(element)：移除集合中的一个元素，如果元素不存在不会引发错误。
s = {1, 2, 3}
s.discard(4)
print(s)  # 输出：{1, 2, 3}

# clear()：清空集合中的所有元素。
s = {1, 2, 3}
s.clear()
print(s)  # 输出：set()

# set函数可以进行比较
a = set('qwertyuiop')
b = set("reagdagd")
print(a)
print(b)
print(a<b)
# {'t', 'r', 'p', 'i', 'w', 'o', 'e', 'y', 'u', 'q'}
# {'r', 'a', 'e', 'g', 'd'}
# True
```

2.3list

```
# zip函数
a = [1,2,3]
b = [4,5,6]
list(zip(a,b))
# >>>[(1,4),(2,5),(3,6)]
strs = ["flower","flow","flight"]
# list(zip(*strs))
print(list(zip(*strs)))
# [('f', 'f', 'f'), ('l', 'l', 'l'), ('o', 'o', 'i'), ('w', 'w', 'g')]

# 将两个列表交叉合并
from itertools import chain
a=[1,2,3,4,5,6]
ans=[8,9,10,22]
sss=list(chain.from_iterable(zip(ans, a)))
# [8, 1, 9, 2, 10, 3, 22, 4]

# 获取二维数组每列元素
matrix = [[3,7,8],[9,11,13],[15,16,17]]
colmin = [i for i in zip(*matrix)]
# >>>[(3, 9, 15), (7, 11, 16), (8, 13, 17)]

# bisect.bisect_left(list , i) 查找该数值将会插入的位置并返回
# 字符串.rfind() 查找字符最后一次出现的位置，没有则返回-1
```

```
# 按照某种规律进行排序
l = [[5,10],[2,5],[4,7],[3,9]]
l = sorted(l, key=lambda x:-x[1])
# >>>[[5, 10], [3, 9], [4, 7], [2, 5]]
l = ["adss","a","sad","sd"]
l = sorted(l,key=lambda x:len(x))
# >>>['a', 'sd', 'sad', 'adss']

# 找出给定列表的所有子集
nums=[1,2,3,4]
l = [[]]
for x in nums:
    l.extend([i + [x] for i in l])

# 对序列中的元素进行累计 eg：求所有元素的异或总和
from functools import reduce
j = [1,2,3,4,5]
reduce(lambda x,y:x ^ y,j)
```

2.4string

```
# find在字符串中查找子串，如果找到，返回字串的第一个字符的索引，找不到返回-1
a='cbacbis'
b='bi'
print(a.find(b))
# 4

# 字符串.rfind() 查找字符最后一次出现的位置，没有则返回-1

# replace将指定字符串替换为另一个字符串
   #str.replace(指定字符，目标字符，不超过的次数)
```

2.5dict

```
# 字母异位词分组
strs = ["eat", "tea", "tan", "ate", "nat", "bat"]
res = []
dic = {}
for s in strs:
    keys = "".join(sorted(s))
    if keys not in dic:
        dic[keys] = [s]
    else:
        dic[keys].append(s)
print(list(dic.values()))
# >>>[['eat', 'tea', 'ate'], ['tan', 'nat'], ['bat']]

 # 字典的键值对反过来
b = {v:k for k,v in d.items()}
for i in d:
    if ans[i] == a:
        l.append(i)
```

```python
# 字典根据值的大小进行排序
d = sorted(d.items(), key=lambda item:item[1])
```

第十一届国赛python_B组

> 参考小蓝刷题
>
> 【蓝桥真题】——2020年蓝桥python组国赛真题+解析+代码（通俗易懂版）
>
> https://blog.csdn.net/m0_55148406/article/details/122863206?spm=1001.2014.3001.5506

> 参考吧唧吧唧orz
>
> 【python语言】第十一届蓝桥杯国赛 pyb组
>
> https://blog.csdn.net/H20211009/article/details/138362478?spm=1001.2014.3001.5506

试题A：美丽的2

```
cnt=0                           #计数器初始化
for i in range(1,2021):         #遍历公元1年到2020年
    if str(i).count("2")>0:     #如果年份中至少有一个2
        cnt+=1                  #计数器+1
print(cnt)                      #输出结果：563
```

试题B：合数个数

```
#判断合数
def heshu(x):
    for i in range(2,x):
        if x%i==0:
            return True
    return False

#主函数
cnt=0
for i in range(1,2021):      #遍历1~2020
    if heshu(i)==True:       #如果判断为合数
        cnt+=1               #计时器+1
print(cnt)                   #输出结果：1713
```

试题C：阶乘约数

> 这里有一个约数定理

```
1.创建1~100的质数集
def prime(x):
    for i in range(2,int(x**0.5)+1):
        if x%i==0:
            return False
    return True
zhishu=[]
for i in range(2,101):
    if prime(i)==True:
```

```
        zhishu.append(i)
#zhishu=[2,3,5,7,11···,97]

#2.计算约数个数
p=[0]*101                #创建计数数组
for num in range(1,101): #遍历1~100
    x=num                #当前变量赋值
    for i in zhishu:     #遍历质数数组
        while x%i==0:    #判断约数
            p[i]+=1      #对应计数+1
            x//=i        #循环条件
#p=[0, 0, 97···0,0]

#3.遍历结果
ans=1
for i in range(1,101):   #遍历1~100
    if p[i]!=0:          #计数数组不为0
        ans*=(p[i]+1)    #根据公式累乘
print(ans)               #39001250856960000
```

试题D：本质上升序列

```
#本质上升序列
s='tocyjkdzcieoiodfpbgcncsrjbhmugdnojjddhllnofawllbhfiadgdcdjstemphmnjihecoapdjjr
prrqnhgccevdarufmliqijgihhfgdcmxvicfauachlifhafpdccfseflcdgjncadfclvfmadvrnaaahah
ndsikzssoywakgnfjjaihtniptwoulxbaeqkqhfwl'
dp=[1]*len(s)                #dp:递增子序列的个数初始化为1
for i in range(len(s)):      #遍历0~len(s)-1
    for j in range(i):       #遍历0~i-1
        if s[i]>s[j]:        #如果当前字符大于以前的字符
            dp[i]+=dp[j]     #把当前字符添加到前面递增子序列的末尾
        if s[i]==s[j]:       #如果当前字符等于以前的字符
            dp[i]-=dp[j]     #去掉重复字符个数
print(sum(dp))               #输出结果：3616159
```

试题E：玩具蛇

> DFS 写

```
import sys
import os
import math

ans = 0    #数目统计
direction = [[1,0], [0,1], [-1,0], [0,-1]]  #4个方向选择
flag = [[0]*4 for _ in range(4)]
def dfs(x, y, count):
    global ans
    # 终止条件
```

```
    if count==16:
        ans += 1
        return

    # 对4个方向进行搜索
    for d in direction:
        dx = x + d[0]
        dy = y + d[1]
        if 0<=dx<4 and 0<=dy<4 and flag[dx][dy]!=1:
            flag[dx][dy] = 1    #标记
            dfs(dx, dy, count+1)    #基于该点继续搜索
            flag[dx][dy] = 0    #上一条路径搜索完毕，消除痕迹

for i in range(4):
    for j in range(4):
        flag[i][j] = 1  #从盒子的不同起点开始依次搜索：A~P
        dfs(i, j, 1)
        flag[i][j] = 0  #取消上一次搜索的起点（以备下一次可选择）
print(ans)
```

试题F：天干地支

```
import sys
import os
import math

year = int(input())
tiangan = {1:"jia", 2:'yi', 3:'bing', 4:'ding', 5:'wu', 6:'ji', 7:'geng',
8:'xin', 9:'ren', 10:'gui'}
dizhi = {1:'zi', 2:'chou', 3:'yin', 4:'mao', 5:'chen', 6:'si', 7:'wu', 8:'wei',
9:'shen', 10:'you', 11:'xu', 12:'hai'}

left = 2020%12
year -= left
x = year%10+1
y = year%12+1
print(tiangan[x]+dizhi[y])
```

试题J：重复字符串

> 这个题目好像有点熟悉啊，是不是国赛题目考过一个，虽然只有四届，但是确实有类似的就是划分成k组然后怎样怎样呢

```
import sys
import os
import math

k = int(input())
s = input()
ans = 0
```

```
"""
将序列分成了k份
每一份的内容都要相同
若字符串长度不为k的倍数 则无法修改
"""
if len(s)%k!=0:
    print("-1")
else:
    duration = len(s)//k
    """
    在每列中 重复最多的字母不用修改 其余字母修改为重复最多的字母
    一共可以划分为k组 每组有duration个元素
    若按列再划分 一共有duration个组 每组有k个元素
    """
    for i in range(duration):
        ds = dict()
        for j in range(k):
            d = s[duration * j + i] #按列获取元素
            if d not in ds: #在寻找元素中构建字典并计数
                ds[d] = 1
            else:
                ds[d] += 1
        ans += k-max(ds.values())
print(ans)
```

试题H：答疑

```
import sys
import os
import math

n = int(input())
s=[0]*n;a=[0]*n;e=[0]*n
for i in range(n):
    s[i], a[i], e[i] = map(int, input().split())

# 计算每位同学的总用时
t = [0]*n
for i in range(n):
    t[i] = a[i] + s[i] + e[i]

time = 0
time_collection = []
t_new = sorted(t)
index = [t.index(x) for x in t_new]
for i in range(len(t_new)):
    time += t_new[i]
    time_collection.append(time-e[index[i]])     #发消息的时间
print(sum(time_collection))
```

试题X：总结

> 可以写的题目：填空题ABCDE

第十二届国赛python_B组

> 参考
>
> 2021第十二届蓝桥杯Python组国赛【真题+解析+代码】
>
> https://blog.csdn.net/qq_73450915/article/details/131018608?spm=1001.2014.3001.5506
>
> 【蓝桥真题】——2021年蓝桥python组国赛真题+解析+代码（通俗易懂版）
>
> https://blog.csdn.net/m0_55148406/article/details/122790101?spm=1001.2014.3001.5506

试题X：总结

可以做的题目有ABCDE这五道填空题，F是前缀和和二分，其实暴力就行，H和与乘积暴力就行，I是数位dp，不会写但可以暴力，

不用写的是G冰山，J是最后一题，不用写。

第十三届国赛python_B组

试题A：斐波那契与7

填空题基本都是有规律的，先打印出来看就OK，斐波那契基本就是60一次循环。

```
f1 = 0
f2 = 1
f3 = 1
c = 0   # 统计出现的次数
for i in range(60):
    print(f3, end=' ')  # 斐波那契数列中的数字
    f3 = (f1+f2)%10
    if f3 ==7:
        c+=1
    f1 = f2
    f2 = f3
# 1 1 2 3 5 8 3 1 4 5 9 4 3 7 0 7 7 4 1 5 6 1 7 8 5 3 8 1 9 0 9 9 8 7 5 2 7 9 6 5
1 6 7 3 0 3 3 6 9 5 4 9 3 2 5 7 2 9 1 0
# 经过验证，60个一循环 每个循环里面有8个个位为7的数字
print()
print(c)
print(202202011200//60)  # 一共3370033520次循环
print((202202011200//60)*8) # 26960268160
```

试题 B: 小蓝做实验

这个和十四届的第二题类似，只不过是去寻找素数的个数。十四届是寻找最大面积其实，预估15届也是一个txt文件，但是需要就是知道怎么读如txt文件很重要。

```
# 读如txt时候这样子：
f=open(r'nums.txt','r',encoding='utf-8')
list0=f.read().split()
```

```
# 运用埃氏筛法进行解题
# 因为只有少部分的数据大于10**8,将数据分为两部份，小于10**8的，大于10**8
f = open(r'primes.txt','r',encoding='utf-8')
txt = f.read().split()          # 讲文件内的东西转化为列表
arr1 = [int(i) for i in txt if len(i)<=8]  # 根据长度分，然后在转换为整型
arr2 = [int(i) for i in txt if len(i)>8]   # 长度为170，所以单独判断花费的时间并不长

# 先默认所有的都为质数(这部分可以看我质数筛的文章)
# 埃氏筛选法效率非常高2分42秒能够找出10*8以内的质数
nums = [True for i in range(10**8+1)]
for i in (range(3,10**8+1)):
    if nums[i]:
```

```
        for j in range(i+i,10**8,i):
            nums[j] = False
c=0 # 记录次数
for i in arr1:  # 根据列表里面的值判断是否为质数
    if nums[i]:
        c+=1
for i in arr2:   # 对大于10**8的数进行判断
    for j in range(2,int(i**0.5)+1):
            if i%j == 0:
                break
    else:
        c+=1
print(c)
```

试题 C: 取模

这个暴力就OK，哦对有一个反证法：就是这个如果是不存在的话那么对1，2，3，4，5，6，，，，n-1这些数字取模的话得到的就是0，1，2，3，4，5，，，，n-2这些肯定都是不也一样的，不然就会存在。

```
# 暴力
t = int(input())
nums = []
for i in range(t):
    n,m = input().split()
    n = int(n)
    m = int(m)
    f = False
    for j in range(1,m+1):
        if f:
            break
        for k in range(j+1,m+1):
            if n%j==n%k:
                f = True
                nums.append('Yes')
                break

    else:
        nums.append('No')
for i in nums:
    print(i)
```

```python
# 反证法代码：
from sys import stdin
T = int(input())
for _ in range(T) :
    n, m = map(int, stdin.readline().split())
    flag = True
    for i in range(1, m + 1) :
        if n % i != i - 1 :
            print('Yes')
            flag = False
            break
    if flag :
        print('No')
```

试题 D: 内存空间

第四题也就是蓝桥杯国赛大题的第二题，是一道大模拟题目，和十四届是一样的，这两道题目需要重点学习一下，虽然不考察算法，但是考验代码的基本功，考验考试的心态，这道题目放最后做感觉心态不好就做不出来。

```python
t = int(input())
zong = 0 # 总大小，单位B
for i in range(t):
    s_lst = input().split()
    if s_lst[0] == 'int':     # 根据不同的输入情况进行分类
        st1 = s_lst[1].split(',')
        zong+=len(st1)*4
    elif s_lst[0] == 'long':
        st1 = s_lst[1].split(',')
        zong+=len(st1)*8
    elif  s_lst[0] == 'String':
        st1 = s_lst[1].split(',')
        num = 0
        for item in st1:
            num+=len(item.split('=')[1])-2
        zong+=num-1
    elif s_lst[0] == 'int[]':
        num = 0
        for it in range(1,len(s_lst)):
            if 'long' in s_lst[it] and ';' not in s_lst[it]:
                num += int(s_lst[it][4:-1])
            elif 'long' in s_lst[it] and ';' in s_lst[it]:
                num += int(s_lst[it][4:-2])
        zong+=num*4
    elif  s_lst[0] == 'long[]':
        num = 0
        for it in range(1,len(s_lst)):
            if 'long' in s_lst[it] and ';' not in s_lst[it]:
                num += int(s_lst[it].split(',')[0][5:-1])
            elif 'long' in s_lst[it] and ';' in s_lst[it]:
                num += int(s_lst[it][5:-2])
```

```
        zong+=num*8

z = [0,0,0,0]   # B，KB，MB，GB 前的数值
for i in range(4):
    z[i]=zong%1024
    zong = zong//1024
    if zong <=0:
        break

result = ''
result_st = ['GB','MB','KB','B']
for i in range(1,len(z)+1):
    if z[4-i] != 0:
        result+=f'{z[4-i]}{result_st[i-1]}'
print(result)
```

试题 E: 近似 GCD

这个可以把能整除的看成0，不能的看成1，然后求一段连续的子序列至少长度为2的，有多少个满足最多只有一个1。

```
import math
n,g=map(int,input().split())
a=[0]+list(map(int,input().split()))

re=0
left=1
right=1
temp=0#记录的上一个不是g的数的位置
for right in range(1,n+1):
    t=math.gcd(g,a[right])
    if t != g:#如果当前这个数不是g的倍数
        left=temp+1
        temp=right
    if right-left+1>=2:re+=right-left
print(re)
```

试题 F: 交通信号

pass

试题 G: 点亮

Pass

试题 H: 打折

pass

试题 I: owo

pass

试题 J: 替换字符

```python
# 暴力模拟就OK,只过了80%但非常多了，这还是最后一道题目，相当于是写出来了的。
s = input()
m = int(input())
for i in range(m):
    nums = input().split()
    l = int(nums[0])
    r = int(nums[1])
    x = nums[2]
    y = nums[3]

    s1 = s[0:l-1]
    s2 = s[l-1:r]
    s3 = s[r:]
    s2 = s2.replace(x,y)
    s = s1+s2+s3
print(s)
```

第十四届国赛python_B组

> 代码及题目参考：
>
> 荷碧TongZJ
>
> 蓝桥杯【第14届国赛】Python B组
>
> https://blog.csdn.net/qq_55745968/article/details/131141353?spm=1001.2014.3001.5506

A：弹珠堆放

题目

> 小蓝有 20230610 颗磁力弹珠，他对金字塔形状尤其感兴趣，如下图所示： 高度为 1 的金字塔需要 1 颗弹珠；高度为 2 的金字塔需要 4 颗弹珠；高度为 3 的金字塔需要 10 颗弹珠；高度为 4 的金字塔需要 20 颗弹珠。小蓝想要知道用他手里的弹珠可以摆出的最高的金字塔高度是多少?

题解

> 等差数列求出每一层需要多少弹珠，算出不能堆出的最高层之后减1就是答案。

```
ans=20230610
for i in range(1,ans):
    need=(i*(i+1)*(i+2))/6
    if need>ans:
        break
print(i-1)
```

B：划分

题目

> 给定 40 个数，请将其任意划分成两组，每组至少一个元素。每组的权值为 组内所有元素的和。划分的权值为两组权值的乘积。请问对于以下 40 个数，划分的权值最大为多少。
>
> 5160 9191 6410 4657 7492 1531 8854 1253 4520 9231 1266 4801 3484 4323 5070 1789 2744 5959 9426 4433 4404 5291 2470 8533 7608 2935 8922 5273 8364 8819 7374 8077 5336 8495 5602 6553 3548 5267 9150 3309
>
> 在试题包中有一个名为 nums.txt 的文本文件，文件中的数与题面上的数 相同。

题解

> 就是分成正方形最大面积那样子类似于，所以肯定是两边相等，那么能不能凑出两边相等呢，堵他能相等就行。

```python
with open('nums.txt') as f:
    array = sorted(map(int, f.read().split()), reverse=True)
print((sum(array)//2)**2)
```

C：偶串

题目

> 小蓝特别喜欢偶数，当他看到字符串时，他总数要检查一下是不是每种字 符都是出现偶数次。给定一个字符串，请帮助小蓝检查一下该字符串是否满足 要求。

题解

> Counter无需多言

```python
from collections import Counter
def solve(s):
    c=Counter(list(s))
    for item in c.values():
        if item%2!=0:
            return False
    return True

s=input()
if solve(s):
    print('YES')
else:
    print('NO')
```

D：交易账本

题目

> 小蓝最近研发了一种新的记账方式，并邀请了一些用户参加测试。交易账 本可以看作是交易记录的集合，每条交易记录都有着一个独一无二的交易编号 txId （编号大小反映了交易记录产生的时间顺序，txId 小的交易记录先发生于 txId 大的交易记录），每条交易记录包含一个或多个输入信息以及一个或多个输 出信息。
>
> 其中输入来自于已经发生过的某比交易的某个输出，可以理解为这笔钱从 某比交易输出后继续输入到了当前这比交易中，输入信息主要包含以下数据： fromTxId、fromTxOutNumber ，这表示当前输入来自于交易编号为 fromT xId 的第 fromTxOutNumber (fromTxOutNumber = 0, 1, 2, · · ·) 个输出；输出信息主 要包含以下数据：account、val ，表示将 val 数目的钱转移到了账户编号为 account 的账户上。注意，当 fromTxId 和 fromTxOutNumber 都为 −1 时，表明这是一笔特殊交易，由系统账户直接产生输出，特殊交易只含有一个输入和 一个输出，可以认为系统账户拥有无限多数目的钱，特殊交易一定可以成功。

> 个合法的账本应满足以下条件：1）对于每笔交易记录，所有的输入中涉 及到的钱的总数目应和所有输出中钱的总数目相等；2）交易中的一个输出要么 不使用，要使用的话输出中的钱应该全部分配给下一个输入，而不能分配给多 个输入（特殊交易除外）；3）交易按照顺序进行，不可以在某比交易中引用还 未发生的交易。
>
> 现在已知一共有 N 个不同的账户，初始时所有账户钱数目都为 0，账本上 总计有 M 条交易记录（按照交易完成的顺序进行记录），请你来判断下账本上 的记录是否是合法的。

题解

> 大模拟题目，去年在D也是一道这样的题目，这两道题目需要好好看一下，准备准备。

```
for _ in range(int(input())):
    n, m = map(int, input().split())
    memory = []
    for _ in range(m):
        __info = map(int, input().split())
        # fromTxId, fromTxOutNumber
        in_dat = [[next(__info) for _ in range(2)] for _ in range(next(__info))]
        # accout, val, exist
        out_dat = [[next(__info) for _ in range(2)] + [True] for _ in
range(next(__info))]
        # 存储该交易信息
        memory.append([in_dat, out_dat])
    # 如果不是特殊交易，判断是否合法
    try:
        for idx, (in_dat, out_dat) in enumerate(memory):
            if in_dat[0] != [-1, -1]:
                in_cnt = 0
                for f, i in in_dat:
                    assert f < idx, f'该交易还未发生 {f} -> {idx}'
                    tmp = memory[f][1][i]
                    assert tmp[2], f'该输出已被占用 {tmp}'
                    memory[f][1][i][2] = False
                    in_cnt += tmp[1]
                out_cnt = sum(x[1] for x in out_dat)
                assert in_cnt == out_cnt, f'输入输出不符 {in_cnt}, {out_cnt}'
        print('Yes')
    except Exception as e:
        print('NO')
```

E：背包问题

题目

> 小蓝是一位狂热的积木爱好者，家里堆满了自己用积木组装的建筑模型。 最近，有两款新出的积木组件上市，小蓝自然不会错过，他带上了自己的三个 背包来到了积木商城，打算将尽可能多的积木组件带回家，每个背包都有一个 固定的空间大小。小蓝只会购买这两种新出的积木组件 A 和 B，A 和 B 各自会 占用背包的一部分空间，但对于同一种类型的积木占用的空间是相同的。小蓝想知道自己最多能带走多少数量的积木组件。
> 可以认为小蓝有足够的货币，只要背包可以装下的积木他都有能力购买。 商场内的积木数量也是有限制的。

题解

> 这道题目没有写出来没有关系，把自己能看懂的努力学会就ok

F：翻转

题目

> 小蓝制作了 n 个工件，每个工件用一个由小写英文字母组成的，长度为 2 的字符串表示，第 i 个工件表示为 si 。小蓝想把 n 个工件拼接到一起，方便转 移到另一个地方完成下一道工序，而拼接后的工件用字符串 S = s1 + s2 +...+ sn 表示，其中 + 表示一种奇特的拼接方式：对于 c = a + b 来说，如果 a 的第二 个字符和 b 的第一个字符相同，则拼接后的结果 c 长度为 3 而不是 4 ，中间相同的字符可以省略一个，比如 xy + yz = xyz 而 xy + zy = xyzy 。小蓝为了让拼 接后的字符串 S 的长度尽量小，可以将若干个工件进行左右翻转之后再进行拼 接，请问拼接后的字符串 S 的最小长度是多少?
> 请注意所有工件必须按出现顺序依次拼接，可以翻转任意工件。

题解

> 首先自然是读入所有的工件，然后判断是否旋转的时候这样，遍历从第二个往后的所有，先判断前一个是否需要翻转，然后判断后面一个是否需要翻转。最后拼接的时候进行一个模拟计算即可

```python
n = int(input())
subs = [input() for _ in range(n)]
# e.g.: ax, ba, cb
for i in range(1, n):
    # 翻转上一个工件
    if subs[i - 1][0] in subs[i]:
        subs[i - 1] = subs[i - 1][::-1]
    # 翻转当前的工件
    if subs[i][1] == subs[i - 1][1]:
        subs[i] = subs[i][::-1]
# 拼接并得到答案
res = subs.pop(0)
for x in subs:
    res += x[1] if res[-1] == x[0] else x
print(len(res))
```

G：最大阶梯

题目

> 小蓝特别喜爱阶梯图案，阶梯图案可以看做是由若干个大小和颜色都相同 的方格组成的，对于大小为 N 的阶梯图案，包含了 N 个连续的列，其中第 i 列 恰好有 i（1 ≤ i ≤ N）个方格，将这 N 列的底部对齐后便组成了一个阶梯图案， 将其按照 90 度旋转若干次后仍是阶梯图案，下图展示了几个不同大小的阶梯图案小蓝有一块大小为 H × H 的布匹，由 H × H 个大小相同的方格区域组成， 每一个方格都有自己的颜色。小蓝可以沿着方格的边缘对布匹进行裁剪，他想 要知道自己能得到的最大的同色阶梯图案的大小是多少?

题解

> 看博客吧，这道题目也没看，应该能看懂的，看不懂也没事，在最后写一个总结看一个这套题目有多少会的，不会的就不写就行，没事。

```
h = int(input())
array = [list(map(int, input().split())) for _ in range(h)]


def valtrig(arr, idx):
    mid, v = len(arr) // 2, set()
    # 左、右半区
    if idx in (0, 3):
        for r in range(len(arr)):
            for c in range(r + 1 if r < mid else len(arr) - r):
                if idx == 3: c = len(arr) - 1 - c
                v.add(arr[r][c])
                if len(v) > 1: return False
    # 上、下半区
    elif idx in (1, 2):
        for c in range(len(arr)):
            for r in range(c + 1 if c < mid else len(arr) - c):
                if idx == 2: r = len(arr) - 1 - r
                v.add(arr[r][c])
                if len(v) > 1: return False
    return True


def search(x):
    # 枚举左上顶点，计算方阵边界
    for l in range(h):
        r = l + x
        if r <= h:
            for t in range(h):
                b = t + x
                if b <= h:
                    arr = [row[l: r] for row in array[t: b]]
                    # 左右任意半区 + 上下任意半区 = 阶梯图案
                    if (valtrig(arr, 0) or valtrig(arr, 3)) and (
                        valtrig(arr, 1) or valtrig(arr, 2)): return True
    return False


l, r = 1, h
# 二分答案
while l + 1 != r:
    mid = (l + r) // 2
    if search(mid):
        l = mid
    else:
        r = mid
print(l)
```

H：最长回文前后缀

题目

> 给定一个字符串 S ，请找出 S 的一个前缀和后缀，使得它们拼接后是一个 回文串。请输出这个串的最长长度

题解

> 暴力骗分,没有必要用其他技巧，我知道有其他板子算法，但是关键是板子算法记不住啊，自己也手残写不出来~

```
def main():
    s = input()
    for x in range(len(s) * 2, 1, -1):
        # 枚举前缀的长度
        for i in range(max(1, x - len(s)),
                       min(len(s) + 1, x)):
            # 后缀的长度: x - i
            tmp = s[:i] + s[i - x:]
            if tmp == tmp[::-1]: return x
    return 1

print(main())
```

I：贸易航线

题目

> 小蓝要带领一支船队依次经过 n 个地点。小蓝的船上可以携带 k 单位的商品，商品共有 m 种，在每个地点的价格各不相同，部分商品在部分地区无法交易。
>
> 一开始小蓝的船是空的，小蓝可以在每个地点任意地买入各种商品，然后 在之后经过的地点卖出。小蓝的钱很多，你可以认为小蓝买入商品时只会受到 船队的容量的限制。
>
> 问小蓝到达终点时的最大收益是多少。

题解

> 因为货物和钱是无限的，只有容量是有限的，所以只有两种状态：空载、满载某一种货物以 dp[i] 表示载满第 i 种货物时的收益，额外增加 dp[-1] 表示空载时的收益
>
> 每到达一个地点，先把所有的货物卖出，把满载某一种货物的状态 dp[:-1] 传递到 dp[-1]，得到空载时的最优状态
>
> 然后再买入货物，把 dp[-1] 传递到 dp[:-1] 得到满载某一种货物的最优状态

```python
n, m, k = map(int, input().split())

# dp[-1] 表空载, dp[i] 表载满第 i 种货物时的收益
dp = m * [-float('inf')] + [0]
for _ in range(n):
    price = tuple(map(lambda x: int(x) * k, input().split()))
    # 满载时, 卖出不买入
    for i, p in enumerate(price):
        if p > 0: dp[-1] = max(dp[-1], dp[i] + p)
    # 空载时再买入
    for i, p in enumerate(price):
        if p > 0: dp[i] = max(dp[i], dp[-1] - p)
print(dp[-1])
```

J：困局

题目

> 小蓝发现有个小偷在坐标 0 处，正在往 x 轴正方向逃离并保持逃离方向不 变。小蓝不想让他快速逃离（即走到坐标 n + 1 处），准备设立 k 道双向传送门 拖延其时间，每道传送门可以连接 x 轴上的两个不同的整点坐标 p ↔ q ，其中 p, q ∈ [1, n] ，同时每个坐标上最多作为一道传送门的端点。
> 当小偷达到一个整点时，如果其上有传送门，则会触发传送门到达传送门 另一端，当然同一个传送门不能连续触发，当无法传送时小偷会保持向 x 轴正 方向移动。小蓝想通过设置这些传送门使得小偷被至少传送 2k 次，请问有多少 种设置传送门的方式可以完成目标?

题解

> 最后一题不用写管啥呢

X：总结

> 能写的就是AB两个填空题，C考counter，D考大模拟，F算是遍历？能写但好像自己想不到，H直接暴力就行不管算法，H是动态规划不一定写得出来，但是代码很短

> 不用写的题目：E动态规划不用写，G是二分但是不止二分写出来有难度，J不用写。

看自己能力的话，现在应该是改到只有2道填空题，6道大题，总共8道题目了，填空题第一个要写出来，第二题写不出就算了，填空题第一道要写出来，第二道都是大模拟，沉下心来冷静写，再后面四道大题基本就是两个动态规划一个简单一个困难，然后一个字符串，一个路线或者图论之类的。后面四道题目能写出来两道就非常不错了，因为最后一道肯定不会写，剩下就是三道题目，有两道会那挺厉害了，本来就没学多少。