

# Assignment #F: 十全十美

---

Updated 2330 GMT+8 Dec 20, 2023

2023 fall, Compiled by 钟明衡 物理学院

## 说明:

本周作业对零基础同学偏难, 如果耗时太长, 直接找答案看。两个题解, 经常更新。所以最好从这个链接下载最新的, <https://github.com/GMyhf/2020fall-cs101>。

- 1) 请把每个题目解题思路(可选), 源码Python, 或者C++ (已经在Codeforces/Openjudge上AC), 截图(包含Accepted, 学号), 填写到下面作业模版中(推荐使用 typora <https://typoraio.cn>, 或者用word)。AC 或者没有AC, 都请标上每个题目大致花费时间。
- 2) 提交时候先提交pdf文件, 再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、作业评论有md或者doc。
- 3) 如果不能在截止前提交作业, 请写明原因。

## 编程环境

操作系统: Windows\_NT x64 10.0.19045

Python编程环境: Visual Studio Code 1.76.1

C/C++编程环境: Visual Studio Code 1.76.1

## 1. 题目

---

如果耗时太长, 直接看解题思路, 或者源码

### 18155: 组合乘积

dfs, brute force, <http://cs101.openjudge.cn/practice/18155>

思路:

首先算出所有数的乘积 $m$ , 如果能组合, 则 $m$ 必定能整除 $t$ , 否则直接输出'NO'

要排除 $t = 1$ 但是数字中没有1的情况, 然后直接枚举就可以。同一次枚举, 检查是否为 $t$ 或者 $m//t$ , 可以快一些

## 代码

```
1  from sys import exit
2
3
4  def search(count, m, t, step, n, l):
5      if step == n:
6          if count == t or count*t == m:
7              print('YES')
8              exit()
9      else:
10         search(count, m, t, step+1, n, l)
11         search(count*l[step], m, t, step+1, n, l)
12     return
13
14
15 t = int(input())
16 l = list(map(int, input().split()))
17 m = 1
18 for a in l:
19     m *= a
20 if m % t != 0:
21     print('NO')
22 else:
23     if t == 1:
24         if t in l:
25             print('YES')
26         else:
27             print('NO')
28     else:
29         search(1, m, t, 0, len(l), l)
30         print('NO')
31
```

代码运行截图

状态: **Accepted**

源代码

```
from sys import exit

def search(count, m, t, step, n, l):
    if step == n:
        if count == t or count*t == m:
            print('YES')
            exit()
        else:
            search(count, m, t, step+1, n, l)
            search(count*l[step], m, t, step+1, n, l)
    return

t = int(input())
l = list(map(int, input().split()))
m = 1
for a in l:
    m *= a
if m % t != 0:
    print('NO')
else:
    if t == 1:
        if t in l:
            print('YES')
        else:
            print('NO')
    else:
        search(1, m, t, 0, len(l), l)
        print('NO')
```

基本信息

#: 43220598

题目: 18155

提交人: 23n2300011505(12号娱乐选手)

内存: 3648kB

时间: 33ms

语言: Python3

提交时间: 2023-12-19 13:56:53

## 20106: 走山路

bfs, <http://cs101.openjudge.cn/practice/20106/>

思路:

最开始想了一个bfs和dfs都用上的方法，把相邻相同高度的记为同一个区域，并且记录相邻的区域，然后寻找最佳的从区域到区域的路径，结果超时了

然后写了个单独的dfs，不出意料超时了

改用bfs，记录到达每个位置上的最少体力，先写了一个每个位置只走一次的，WA了，然后就把不再走某个位置的判定改为不再能使这个位置体力最小，然后终于AC了

代码

```
1 dx, dy = [1, 0, -1, 0], [0, 1, 0, -1]
2
3
4 def bfs(x, y, a, b):
5     global M, ans
6     lx, ly = [x], [y]
7     start, end = 0, 0
8     while end != len(lx):
```

```

9         start = end
10        end = len(lx)
11        for i in range(start, end):
12            for j in range(4):
13                newx, newy = lx[i]+dx[j], ly[i]+dy[j]
14                if M[newx][newy] != '#':
15                    newans = ans[lx[i]][ly[i]] + \
16                        abs(int(M[lx[i]][ly[i]])-int(M[newx][newy]))
17                    if ans[newx][newy] == -1 or newans < ans[newx][newy]:
18                        ans[newx][newy] = newans
19                        lx.append(newx)
20                        ly.append(newy)
21        return
22
23
24    def intt(s):
25        return int(s)+1
26
27
28    m, n, p = map(int, input().split())
29    M = [['#']*(n+2) for i in range(m+2)]
30    for i in range(m):
31        M[i+1] = ['#']+input().split()+['#']
32    for _ in range(p):
33        x, y, a, b = map(intt, input().split())
34        if M[x][y] == '#' or M[a][b] == '#':
35            print('NO')
36        else:
37            ans = [[-1]*(n+2) for i in range(m+2)]
38            ans[x][y] = 0
39            bfs(x, y, a, b)
40            if ans[a][b] == -1:
41                print('NO')
42            else:
43                print(ans[a][b])
44

```

代码运行截图

状态: Accepted

源代码

```
dx, dy = [1, 0, -1, 0], [0, 1, 0, -1]

def bfs(x, y, a, b):
    global M, ans
    lx, ly = [x], [y]
    start, end = 0, 0
    while end != len(lx):
        start = end
        end = len(lx)
        for i in range(start, end):
            for j in range(4):
                newx, newy = lx[i]+dx[j], ly[i]+dy[j]
                if M[newx][newy] != '#':
                    newans = ans[lx[i]][ly[i]] + \
                        abs(int(M[lx[i]][ly[i]])-int(M[newx][newy]))
                    if ans[newx][newy] == -1 or newans < ans[newx][newy]:
                        ans[newx][newy] = newans
                        lx.append(newx)
                        ly.append(newy)

    return

def intt(s):
    return int(s)+1

m, n, p = map(int, input().split())
M = [['#']*(n+2) for i in range(m+2)]
for i in range(m):
    M[i+1] = ['#']+input().split()+'#'
for _ in range(p):
    x, y, a, b = map(intt, input().split())
    if M[x][y] == '#' or M[a][b] == '#':
        print('NO')
    else:
        ans = [[-1]*(n+2) for i in range(m+2)]
        ans[x][y] = 0
        bfs(x, y, a, b)
        if ans[a][b] == -1:
            print('NO')
        else:
            print(ans[a][b])
```

基本信息

#: 43224424  
题目: 20106  
提交人: 23n2300011505(12号娱乐选手)  
内存: 3888kB  
时间: 1023ms  
语言: Python3  
提交时间: 2023-12-19 15:56:37

## 27314: 一键换词

implementation, string, <http://cs101.openjudge.cn/practice/27314/>

思路:

把输入的句子按空格拆开，然后逐个词判断：

如果最后一位是句号或者逗号，就记录，在最后输出，且如果是句号就打开下一位的首字母大写

然后把分离出来的单词变成小写，比较是否要替换，然后按照大写要求输出，再输出符号，如果不是最后一个单词就额外输出一个空格

## 代码

```
1 l = input().split()
2 s1, s2 = input().split()
3 s1, s2 = s1.lower(), s2.lower()
4 cap = True
5 for i in range(len(l)):
6     s = l[i]
7     b = False
8     c = False
9     if s[-1] == '.':
10         s = s[:-1]
11         b = True
12     if s[-1] == ',':
13         s = s[:-1]
14         c = True
15     s = s.lower()
16     if s == s1:
17         s = s2
18     if cap:
19         cap = False
20         s = s.capitalize()
21     print(s, end='')
22     if b:
23         cap = True
24         print('.', end='')
25     if c:
26         print(',', end='')
27     if i != len(l)-1:
28         print(' ', end='')
29     else:
30         print('')
31
```

代码运行截图

状态: Accepted

源代码

```
l = input().split()
s1, s2 = input().split()
s1, s2 = s1.lower(), s2.lower()
cap = True
for i in range(len(l)):
    s = l[i]
    b = False
    c = False
    if s[-1] == '.':
        s = s[:-1]
        b = True
    if s[-1] == ',':
        s = s[:-1]
        c = True
    s = s.lower()
    if s == s1:
        s = s2
    if cap:
        cap = False
        s = s.capitalize()
    print(s, end='')
    if b:
        cap = True
        print('.', end='')
    if c:
        print(',', end='')
    if i != len(l)-1:
        print(' ', end='')
    else:
        print('')
```

基本信息

#: 43225131

题目: 27314

提交人: 23n2300011505(12号娱乐选手)

内存: 3636kB

时间: 25ms

语言: Python3

提交时间: 2023-12-19 16:19:20

## 19961: 最大点数(外太空2048)

matrices, <http://cs101.openjudge.cn/practice/19961/>

思路:

巨大的模拟题，直接写出四个方向移动后的结果，然后枚举即可

代码

```
1 from copy import deepcopy
2
3
4 def right(M):
5     global m, n
6     changed = False
7     while True:
8         c = False
9         for i in range(m):
10             for j in range(n-1):
11                 if M[i][j] > 0 and M[i][j+1] == 0:
12                     M[i][j], M[i][j+1] = 0, M[i][j]
13                     c = True
14         changed = c
```

```

15         if not c:
16             break
17     for i in range(m):
18         for j in range(n-1):
19             if M[i][j] == M[i][j+1]:
20                 M[i][j+1] *= 2
21                 M[i][j] = 0
22                 changed = True
23     while True:
24         c = False
25         for i in range(m):
26             for j in range(n-1):
27                 if M[i][j] > 0 and M[i][j+1] == 0:
28                     M[i][j], M[i][j+1] = 0, M[i][j]
29                     c = True
30         if not c:
31             break
32     return changed
33
34
35 def left(M):
36     global m, n
37     changed = False
38     while True:
39         c = False
40         for i in range(m):
41             for j in range(n-1, 0, -1):
42                 if M[i][j] > 0 and M[i][j-1] == 0:
43                     M[i][j], M[i][j-1] = 0, M[i][j]
44                     c = True
45         changed = c
46         if not c:
47             break
48     for i in range(m):
49         for j in range(n-1, 0, -1):
50             if M[i][j] == M[i][j-1]:
51                 M[i][j-1] *= 2
52                 M[i][j] = 0
53                 changed = True
54     while True:
55         c = False
56         for i in range(m):
57             for j in range(n-1, 0, -1):
58                 if M[i][j] > 0 and M[i][j-1] == 0:
59                     M[i][j], M[i][j-1] = 0, M[i][j]
60                     c = True
61         if not c:
62             break
63     return changed
64
65
66 def down(M):

```



```

67     global m, n
68     changed = False
69     while True:
70         c = False
71         for j in range(n):
72             for i in range(m-1):
73                 if M[i][j] > 0 and M[i+1][j] == 0:
74                     M[i][j], M[i+1][j] = 0, M[i][j]
75                     c = True
76         changed = c
77         if not c:
78             break
79     for j in range(n):
80         for i in range(m-1):
81             if M[i][j] == M[i+1][j]:
82                 M[i+1][j] *= 2
83                 M[i][j] = 0
84                 changed = True
85     while True:
86         c = False
87         for j in range(n):
88             for i in range(m-1):
89                 if M[i][j] > 0 and M[i+1][j] == 0:
90                     M[i][j], M[i+1][j] = 0, M[i][j]
91                     c = True
92         if not c:
93             break
94     return changed
95
96
97 def up(M):
98     global m, n
99     changed = False
100    while True:
101        c = False
102        for j in range(n):
103            for i in range(m-1, 0, -1):
104                if M[i][j] > 0 and M[i-1][j] == 0:
105                    M[i][j], M[i-1][j] = 0, M[i][j]
106                    c = True
107        changed = c
108        if not c:
109            break
110    for j in range(n):
111        for i in range(m-1, 0, -1):
112            if M[i][j] == M[i-1][j]:
113                M[i-1][j] *= 2
114                M[i][j] = 0
115                changed = True
116    while True:
117        c = False
118        for j in range(n):

```

```

119         for i in range(m-1, 0, -1):
120             if M[i][j] > 0 and M[i-1][j] == 0:
121                 M[i][j], M[i-1][j] = 0, M[i][j]
122                 c = True
123             if not c:
124                 break
125         return changed
126
127
128 def move(M, step):
129     global ans
130     ans.append(max(max(l) for l in M))
131     if step == 0:
132         return
133     newM = deepcopy(M)
134     if right(newM):
135         move(newM, step-1)
136     newM = deepcopy(M)
137     if left(newM):
138         move(newM, step-1)
139     newM = deepcopy(M)
140     if down(newM):
141         move(newM, step-1)
142     newM = deepcopy(M)
143     if up(newM):
144         move(newM, step-1)
145     return
146
147
148 m, n, p = map(int, input().split())
149 M, ans = [], []
150 for _ in range(m):
151     M.append(list(map(int, input().split())))
152 move(M, p)
153 print(max(ans))
154

```

代码运行截图

状态: Accepted

源代码

```
from copy import deepcopy

def right(M):
    global m, n
    changed = False
    while True:
        c = False
        for i in range(m):
            for j in range(n-1):
                if M[i][j] > 0 and M[i][j+1] == 0:
                    M[i][j], M[i][j+1] = 0, M[i][j]
                    c = True
        changed = c
        if not c:
            break
    for i in range(m):
        for j in range(n-1):
            if M[i][j] == M[i][j+1]:
                M[i][j+1] *= 2
                M[i][j] = 0
                changed = True
    while True:
        c = False
        for i in range(m):
            for j in range(n-1):
                if M[i][j] > 0 and M[i][j+1] == 0:
                    M[i][j], M[i][j+1] = 0, M[i][j]
                    c = True
        if not c:
            break
    return changed

def left(M):
    global m, n
    changed = False
    while True:
        c = False
        for i in range(m):
            for j in range(n-1, 0, -1):
                if M[i][j] > 0 and M[i][j-1] == 0:
                    M[i][j], M[i][j-1] = 0, M[i][j]
                    c = True
        changed = c
        if not c:
            break
    for i in range(m):
        for j in range(n-1, 0, -1):
            if M[i][j] == M[i][j-1]:
                M[i][j-1] *= 2
                M[i][j] = 0
                changed = True
    while True:
        c = False
        for i in range(m):
            for j in range(n-1, 0, -1):
                if M[i][j] > 0 and M[i][j-1] == 0:
                    M[i][j], M[i][j-1] = 0, M[i][j]
                    c = True
        if not c:
            break
    return changed

def down(M):
    global m, n
    changed = False
    while True:
        c = False
        for j in range(n):
            for i in range(m-1):
                if M[i][j] > 0 and M[i+1][j] == 0:
                    M[i][j], M[i+1][j] = 0, M[i][j]
                    c = True
        changed = c
```

基本信息

#: 43228478  
题目: 19961  
提交人: 23n2300011505(12号娱乐选手)  
内存: 4100kB  
时间: 981ms  
语言: Python3  
提交时间: 2023-12-19 17:59:27

```

        if not c:
            break
    for j in range(n):
        for i in range(m-1):
            if M[i][j] == M[i+1][j]:
                M[i+1][j] *= 2
                M[i][j] = 0
                changed = True
    while True:
        c = False
        for j in range(n):
            for i in range(m-1):
                if M[i][j] > 0 and M[i+1][j] == 0:
                    M[i][j], M[i+1][j] = 0, M[i][j]
                    c = True
        if not c:
            break
    return changed

def up(M):
    global m, n
    changed = False
    while True:
        c = False
        for j in range(n):
            for i in range(m-1, 0, -1):
                if M[i][j] > 0 and M[i-1][j] == 0:
                    M[i][j], M[i-1][j] = 0, M[i][j]
                    c = True
        changed = c
        if not c:
            break
    for j in range(n):
        for i in range(m-1, 0, -1):
            if M[i][j] == M[i-1][j]:
                M[i-1][j] *= 2
                M[i][j] = 0
                changed = True
    while True:
        c = False
        for j in range(n):
            for i in range(m-1, 0, -1):
                if M[i][j] > 0 and M[i-1][j] == 0:
                    M[i][j], M[i-1][j] = 0, M[i][j]
                    c = True
        if not c:
            break
    return changed

def move(M, step):
    global ans
    ans.append(max(max(l) for l in M))
    if step == 0:
        return
    newM = deepcopy(M)
    if right(newM):
        move(newM, step-1)
    newM = deepcopy(M)
    if left(newM):
        move(newM, step-1)
    newM = deepcopy(M)
    if down(newM):
        move(newM, step-1)
    newM = deepcopy(M)
    if up(newM):
        move(newM, step-1)
    return

m, n, p = map(int, input().split())
M, ans = [], []
for _ in range(m):
    M.append(list(map(int, input().split())))
move(M, p)
print(max(ans))

```

## 27401: 最佳凑单

dp, sparse bucket, <http://cs101.openjudge.cn/practice/27401/>

思路:

反过来看这个问题, 把所有东西都放进购物车, 再看删掉哪些可以让溢价最小

如果全都买, 价格还不够, 就无法凑单, 输出0

如果可以凑单, 按照从大到小排序, 在另一个列表里面记录如果删了第 $i$ 个东西, 还溢出多少价格 (从第一个可以删掉的位置 $start$ 开始)

这个新的溢出价格 $ans[i]$ , 在前面所有的溢出价格 $ans[j] (j < i)$ 中, 寻找大于物品价格 $l[i]$ 的最小的那个

即:  $ans[i] = \min(ans[j]) - l[i]$ , 要求 $j < i$ ,  $ans[j] \geq l[i]$  (找不到的话, 就取最开始的溢出值)

代码

```
1  n, t = map(int, input().split())
2  l = sorted(list(map(int, input().split()))), reverse=True)
3  count = sum(l)-t
4  if count < 0:
5      print(0)
6  else:
7      ans = [count]*(n+1)
8      start = 0
9      while l[start] > count:
10         start += 1
11         if start == n:
12             break
13     for i in range(start, n):
14         minn = count
15         for j in range(start, i):
16             if ans[j] >= l[i]:
17                 minn = min(ans[j], minn)
18         ans[i] = minn-l[i]
19     print(min(ans)+t)
20
```

代码运行截图

状态: Accepted

源代码

```
n, t = map(int, input().split())
l = sorted(list(map(int, input().split())), reverse=True)
count = sum(l)-t
if count < 0:
    print(0)
else:
    ans = [count]*(n+1)
    start = 0
    while l[start] > count:
        start += 1
        if start == n:
            break
    for i in range(start, n):
        minn = count
        for j in range(start, i):
            if ans[j] >= l[i]:
                minn = min(ans[j], minn)
        ans[i] = minn-l[i]
    print(min(ans)+t)
```

基本信息

#: 43252906

题目: 27401

提交人: 23n2300011505(12号娱乐选手)

内存: 3656kB

时间: 23ms

语言: Python3

提交时间: 2023-12-20 16:51:28

## 27384: 候选人追踪

heap, <http://cs101.openjudge.cn/practice/27384/>

熊江凯, 这题应该不超纲的, 感觉还是挺好的

思路:

判断方法很简单, 只要候选人的最低票数 &gt; 非候选人的最高票数即可

非候选人最高票数很好统计, 但是候选人的最低票数不太好统计, 不超时的方法如下: 沿用之前一次作业里的想法, 记录每个候选人的票数, 同时记录票数为某个值的候选人的个数, 如果票数最少的候选人的个数为1且此时该候选人得了一票, 那么就把最低票数加一

有一个巨大的坑, 就是如果所有人都是候选人 (即 $k = 314159$ ), 那么要直接输出最大时间, 而不是正常计算

代码

```
1 n, k = map(int, input().split())
2 l = list(map(int, input().split()))
3 s = list(map(int, input().split()))
4 left, right, ans, last = 0, 0, 0, 0
5 iss, countC, counts, vote = {}, {}, {}, {}
6 for i in range(n):
7     iss[l[2*i+1]] = False
8     vote[l[2*i]] = []
9 vote = dict(sorted(vote.items(), key=lambda x: x[0]))
10 if k == 314159:
11     print(max(vote.keys()))
12     exit()
13 for i in range(n):
```

```
14     vote[l[2*i]].append(l[2*i+1])
15 for a in s:
16     iss[a] = True
17 for i in range(n):
18     if iss[l[2*i+1]]:
19         counts[l[2*i+1]] = 0
20     else:
21         countC[l[2*i+1]] = 0
22 count = [0]*(n+1)
23 count[0] = k
24 for i in vote.keys():
25     if right > left:
26         ans += i-last
27     for a in vote[i]:
28         if iss[a]:
29             count[counts[a]] -= 1
30             counts[a] += 1
31             count[counts[a]] += 1
32             if count[right] == 0:
33                 right += 1
34         else:
35             countC[a] += 1
36             left = max(left, countC[a])
37     last = i
38 print(ans)
39
```

代码运行截图

状态: Accepted

源代码

```

n, k = map(int, input().split())
l = list(map(int, input().split()))
s = list(map(int, input().split()))
left, right, ans, last = 0, 0, 0, 0
isS, countC, countS, vote = {}, {}, {}, {}
for i in range(n):
    isS[l[2*i+1]] = False
    vote[l[2*i]] = []
vote = dict(sorted(vote.items(), key=lambda x: x[0]))
if k == 314159:
    print(max(vote.keys()))
    exit()
for i in range(n):
    vote[l[2*i]].append(l[2*i+1])
for a in s:
    isS[a] = True
for i in range(n):
    if isS[l[2*i+1]]:
        countS[l[2*i+1]] = 0
    else:
        countC[l[2*i+1]] = 0
count = [0]*(n+1)
count[0] = k
for i in vote.keys():
    if right > left:
        ans += i-last
    for a in vote[i]:
        if isS[a]:
            count[countS[a]] -= 1
            countS[a] += 1
            count[countS[a]] += 1
            if count[right] == 0:
                right += 1
        else:
            countC[a] += 1
            left = max(left, countC[a])
    last = i
print(ans)

```

基本信息

#: 43254431  
 题目: 27384  
 提交人: 23n2300011505(12号娱乐选手)  
 内存: 122376kB  
 时间: 1976ms  
 语言: Python3  
 提交时间: 2023-12-20 17:44:46

## CF1883D. In Love

data structure, greedy, 1500, <https://codeforces.com/problemset/problem/1883/D>

黄源森、查达闻推荐

思路:

对于 $(l_1, r_1)$ 和 $(l_2, r_2)$ , 如果 $r_1 < l_2$ , 则它们不相交

因此, 每次输出只需要判断 $r$ 中的最小值是否小于 $l$ 中的最大值

直接对列表操作会超时, 因此使用heap, 不断弹出不符合要求的最小值 (最大值加个负号就是最小值了)

代码

```

1 from heapq import heappush, heappop
2 from collections import defaultdict
3 q = int(input())
4 l, r, ans = [], [], [False]*q
5 ll, rr = defaultdict(int), defaultdict(int)
6 for i in range(q):

```



```

7      s, a, b = input().split()
8      a, b = int(a), int(b)
9      if s == '+':
10         ll[a] += 1
11         rr[b] += 1
12         heappush(l, -a)
13         heappush(r, b)
14     else:
15         ll[a] -= 1
16         rr[b] -= 1
17     while l and ll[-l[0]] == 0:
18         heappop(l)
19     while r and rr[r[0]] == 0:
20         heappop(r)
21     if l and r and -l[0] > r[0]:
22         ans[i] = True
23 for a in ans:
24     if a:
25         print('YES')
26     else:
27         print('NO')
28

```

代码运行截图

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
238163907	Practice: MinghengZhong	<a href="#">1883D</a> - 17	Python 3	Accepted	436 ms	29264 KB	2023-12-20 18:23:10	2023-12-20 18:23:10	★	<a href="#">Compare</a>

[Source](#)
[Copy](#)

```

from heapq import heappush, heappop
from collections import defaultdict
q = int(input())
l, r, ans = [], [], [False]*q
ll, rr = defaultdict(int), defaultdict(int)
for i in range(q):
    s, a, b = input().split()
    a, b = int(a), int(b)
    if s == '+':
        ll[a] += 1
        rr[b] += 1
        heappush(l, -a)
        heappush(r, b)
    else:
        ll[a] -= 1
        rr[b] -= 1
    while l and ll[-l[0]] == 0:
        heappop(l)
    while r and rr[r[0]] == 0:
        heappop(r)
    if l and r and -l[0] > r[0]:
        ans[i] = True
for a in ans:
    if a:
        print('YES')
    else:
        print('NO')

```

## 2. 学习总结和收获

感觉这次作业特别难，七道题分别为：枚举剪枝、特殊bfs、字符串处理、暴力模拟、dp、桶、heap

第一次使用了heap，确实很快