

# Assignment #2: 编程练习

2024 spring, Compiled by 23工院 武昱达

## 编程环境

操作系统: Windows 11

Python编程环境: pyCharm 2023.1.4 (Professional Edition)

## 1. 题目

### 27653: Fraction类

[http://cs101.openjudge.cn/2024sp\\_routine/27653/](http://cs101.openjudge.cn/2024sp_routine/27653/)

思路:

#### 代码

```
1 import math
2 a1,a2,b1,b2=map(int,input().split())
3 temp1=a1*b2+b1*a2
4 temp2=a2*b2
5 a=math.gcd(temp1,temp2)
6 print(str(temp1//a)+"/"+str(temp2//a))
```

代码运行截图 (至少包含有"Accepted")

### #43957910提交状态

状态: Accepted

#### 源代码

```
import math
a1,a2,b1,b2=map(int,input().split())
temp1=a1*b2+b1*a2
temp2=a2*b2
a=math.gcd(temp1,temp2)
print(str(temp1//a)+"/"+str(temp2//a))
```

©2002-2022 POJ 京ICP备20010980号-1

### 04110: 圣诞老人的礼物-Santa Clau's Gifts

greedy/dp, <http://cs101.openjudge.cn/practice/04110>

思路:

#### 代码

```
1 n,w=map(int,input().split())
2 lst_candy=[]
3 for _ in range(n):
4     lst_candy.append(list(map(int,input().split())))
```

```

5  for list_value_weight in lst_candy:
6      value,weight=list_value_weight[0],list_value_weight[1]
7      list_value_weight.append(value/weight)
8  lst_candy.sort(key=lambda x: x[2],reverse=True)
9  # print(lst_candy)
10 #lst_candy的格式为[[value,weight,value_per_weight],...]
11 weight=0
12 value=0
13 flag=0
14 while weight<w:
15     try:
16         if weight+lst_candy[flag][1]<w:
17             value+=lst_candy[flag][0]
18             weight+=lst_candy[flag][1]
19             flag+=1
20
21         else:
22             left_weight=w-weight
23             value+=lst_candy[flag][2]*left_weight
24             break
25     except:
26         break
27 print(round(float(value),1))

```

代码运行截图 (至少包含有"Accepted")

#41821670提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

n,w=map(int,input().split())
lst_candy=[]
for _ in range(n):
    lst_candy.append(list(map(int,input().split())))
for list_value_weight in lst_candy:
    value,weight=list_value_weight[0],list_value_weight[1]
    list_value_weight.append(value/weight)
lst_candy.sort(key=lambda x: x[2],reverse=True)
# print(lst_candy)
#lst_candy的格式为[[value,weight,value_per_weight],...]
weight=0
value=0
flag=0
while weight<w:
    try:
        if weight+lst_candy[flag][1]<w:
            value+=lst_candy[flag][0]
            weight+=lst_candy[flag][1]
            flag+=1

        else:
            left_weight=w-weight
            value+=lst_candy[flag][2]*left_weight
            break
    except:
        break
print(round(float(value),1))

```

基本信息

#: 41821670  
 题目: 04110  
 提交人: 23n2300011119 (武)  
 内存: 3600kB  
 时间: 21ms  
 语言: Python3  
 提交时间: 2023-10-20 19:39:06

## 18182: 打怪兽

implementation/sortings/data structures, <http://cs101.openjudge.cn/practice/18182/>

思路:

代码

```
1  nCases=int(input())
2  lst_1=[]
3  for _ in range(nCases):
4      n,m,b=map(int,input().split())
5      dict_raw={}
6      set_ti=set()
7      for i in range(n):
8          ti,xi=map(int,input().split())
9          set_ti.add(ti)
10         try:
11             temp=type(dict_raw[ti])
12             dict_raw[ti].append(xi)
13         except:
14             dict_raw[ti]=[]
15             dict_raw[ti].append(xi)
16     # print(dict_raw)
17     lst_ti=[i for i in set_ti]
18     lst_ti.sort()
19     # print(lst_ti)
20
21     flag=0
22     while True:
23         try:
24             if b>0:
25                 temp=dict_raw[lst_ti[flag]]
26                 temp.sort(reverse=True)
27                 times=0
28                 while times<m:
29                     try:
30                         b-=temp[times]
31                         times+=1
32                         if b<=0:
33                             lst_1.append(lst_ti[flag])
34                             break
35                     except:
36                         break
37                 flag+=1
38             else:
39                 break
40         except:
41             break
42     if b>0:
43         lst_1.append("alive")
44     for i in lst_1:
45         print(i)
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```

nCases=int(input())
lst_1=[]
for _ in range(nCases):
    n,m,b=map(int,input().split())
    dict_raw={}
    set_ti=set()
    for i in range(n):
        ti,xi=map(int,input().split())
        set_ti.add(ti)
        try:
            temp=type(dict_raw[ti])
            dict_raw[ti].append(xi)
        except:
            dict_raw[ti]=[]
            dict_raw[ti].append(xi)
    # print(dict_raw)
    lst_ti=[i for i in set_ti]
    lst_ti.sort()
    # print(lst_ti)

    flag=0
    while True:
        try:
            if b>0:
                temp=dict_raw[lst_ti[flag]]
                temp.sort(reverse=True)
                times=0
                while times<m:
                    try:
                        b-=temp[times]
                        times+=1
                        if b<=0:
                            lst_1.append(lst_ti[flag])
                            break
                    except:
                        break
                flag+=1
            else:
                break

```

基本信息

#: 42183532  
 题目: 18182  
 提交人: 23n2300011119 (武)  
 内存: 3772kB  
 时间: 81ms  
 语言: Python3  
 提交时间: 2023-11-02 17:03:26

## 230B. T-primes

binary search/implementation/math/number theory, 1300, <http://codeforces.com/problemset/problem/230/B>

思路: 判断完全平方数; 欧拉筛

代码

```

1  import math
2  #判断是否为完全平方数 不是则pass (参考drunkjailor)
3  def is_perfect(x):
4      sqrt=math.sqrt(x)
5      if sqrt.is_integer():
6          return True
7      else:
8          return False
9  #如果是完全平方数, 判断因子个数
10 def Euler_sieve(n):
11     primes = [True for _ in range(n+1)]
12     p = 2
13     while p*p <= n:
14         if primes[p]:
15             for i in range(p*p, n+1, p):
16                 primes[i] = False
17             p += 1
18     return primes
19 tuple_primes=tuple(Euler_sieve(1000000))

```

```

20 def is_T_primes(x):
21     set_1=set()
22     if not is_perfect(x):
23         return False
24     if x==4 or x==9:
25         return True
26     elif x==1:
27         return False
28     else:
29         a=int(math.sqrt(x))
30         tuple_basic=(0,2,3,4)
31         if a%6 in tuple_basic:
32             return False
33         else:
34             if tuple_primes[a]:
35                 return True
36             else:
37                 return False
38
39
40 n=int(input())
41 int_tuple=tuple(map(int,input().split()))
42 int_set=set(int_tuple)
43 dict_1={i:is_T_primes(i) for i in int_set}
44 for i in int_tuple:
45     print("YES" if dict_1[i] else "NO")

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

General

| #         | Author              | Problem                   | Lang     | Verdict  | Time      | Memory   |
|-----------|---------------------|---------------------------|----------|----------|-----------|----------|
| 228247167 | Practice:<br>wuyuda | <a href="#">230B</a> - 28 | Python 3 | Accepted | 998<br>ms | 26396 KB |

→ Source

```

import math
#判断是否为完全平方数 不是则pass (参考drunkjailor)
def is_perfect(x):
    sqrt=math.sqrt(x)
    if sqrt.is_integer():
        return True
    else:
        return False
#如果是完全平方数, 判断因子个数
def Euler_sieve(n):
    primes = [True for _ in range(n+1)]
    p = 2
    while p*p <= n:
        if primes[p]:
            for i in range(p*p, n+1, p):
                primes[i] = False
            p += 1
    return primes
tuple_primes=tuple(Euler_sieve(1000000))
def is T primes(x):

```

## 1364A. XXXXX

brute force/data structures/number theory/two pointers, 1200, <https://codeforces.com/problemset/problem/1364/A>

思路:

代码

```
1 def prefix_sum(nums):
2     prefix = []
3     total = 0
4     for num in nums:
5         total += num
6         prefix.append(total)
7     return prefix
8
9 def suffix_sum(nums):
10    suffix = []
11    total = 0
12    # 首先将列表反转
13    reversed_nums = nums[::-1]
14    for num in reversed_nums:
15        total += num
16        suffix.append(total)
17    # 将结果反转回来
18    suffix.reverse()
19    return suffix
20
21
22 t = int(input())
23 for _ in range(t):
24     N, x = map(int, input().split())
25     a = [int(i) for i in input().split()]
26     aprefix_sum = prefix_sum(a)
27     asuffix_sum = suffix_sum(a)
28
29     left = 0
30     right = N - 1
31     if right == 0:
32         if a[0] % x != 0:
33             print(1)
34         else:
35             print(-1)
36         continue
37
38     leftmax = 0
39     rightmax = 0
40     while left != right:
41         total = asuffix_sum[left]
42         if total % x != 0:
43             leftmax = right - left + 1
44             break
45         else:
46             left += 1
47
```

```

48     left = 0
49     right = N - 1
50     while left != right:
51         total = aprefix_sum[right]
52         if total % x != 0:
53             rightmax = right - left + 1
54             break
55         else:
56             right -= 1
57
58     if leftmax == 0 and rightmax == 0:
59         print(-1)
60     else:
61         print(max(leftmax, rightmax))
62

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

| General   |                     |                            |          |          |           |          |
|-----------|---------------------|----------------------------|----------|----------|-----------|----------|
| #         | Author              | Problem                    | Lang     | Verdict  | Time      | Memory   |
| 227265165 | Practice:<br>wuyuda | <a href="#">1364A</a> - 15 | Python 3 | Accepted | 374<br>ms | 24748 KB |

→ Source

```

def prefix_sum(nums):
    prefix = []
    total = 0
    for num in nums:
        total += num
        prefix.append(total)
    return prefix

def suffix_sum(nums):
    suffix = []
    total = 0
    # 首先将列表反转
    reversed_nums = nums[::-1]
    for num in reversed_nums:
        total += num
        suffix.append(total)
    # 将结果反转回来
    suffix.reverse()
    return suffix

t = int(input())
for _ in range(t):
    N, x = map(int, input().split())

```

## 18176: 2050年成绩计算

<http://cs101.openjudge.cn/practice/18176/>

思路:

代码

```

1 import math
2 def is_perfect(x):
3     sqrt = math.sqrt(x)
4     if sqrt.is_integer(): return True
5     else: return False
6 def Euler_sieve(n):
7     primes = [True for _ in range(n + 1)]
8     p = 2

```

```

9     while p * p <= n:
10         if primes[p]:
11             for i in range(p * p, n + 1, p):
12                 primes[i] = False
13         p += 1
14     return primes
15 tuple_primes = tuple(Euler_sieve(10000))
16 def is_T_primes(x):
17     set_1 = set()
18     if not is_perfect(x):
19         return False
20     if x == 4 or x == 9: return True
21     elif x == 1: return False
22     else:
23         a = int(math.sqrt(x))
24         tuple_basic = (0, 2, 3, 4)
25         if a % 6 in tuple_basic: return False
26     else:
27         if tuple_primes[a]: return True
28         else: return False
29
30 m,n=map(int,input().split())
31 res=[]
32 for i in range(m):
33     temp=list(map(int,input().split()))
34     valid=[]
35     for j in temp:
36         if is_T_primes(j):
37             valid.append(j)
38     if valid:res.append("{:.2f}".format(sum(valid)/len(temp)))
39     else:res.append(0)
40
41 for i in res:
42     print(i)

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")



状态: Accepted

源代码

```
import math
def is_perfect(x):
    sqrt = math.sqrt(x)
    if sqrt.is_integer():return True
    else:return False
def Euler_sieve(n):
    primes = [True for _ in range(n + 1)]
    p = 2
    while p * p <= n:
        if primes[p]:
            for i in range(p * p, n + 1, p):
                primes[i] = False
            p += 1
    return primes
tuple_primes = tuple(Euler_sieve(10000))
def is_T_primes(x):
    set_1 = set()
    if not is_perfect(x):
        return False
    if x == 4 or x == 9:return True
    elif x == 1:return False
    else:
        a = int(math.sqrt(x))
        tuple_basic = (0, 2, 3, 4)
        if a % 6 in tuple_basic:return False
        else:
            if tuple_primes[a]:return True
            else:return False

m,n=map(int,input().split())
res=[]
for i in range(m):
    temp=list(map(int,input().split()))
    valid=[]
    for j in temp:
        if is_T_primes(j):
            valid.append(j)
    if valid:res.append("{:.2f}".format(sum(valid)/len(temp)))
    else:res.append(0)

for i in res:
    print(i)
```

基本信息

#: 43977185  
题目: 18176  
提交人: 23n2300011119 (武)  
内存: 5196kB  
时间: 80ms  
语言: Python3  
提交时间: 2024-02-24 11:22:51

## 2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。

题目都做过了，非常简单；额外完成了一些有关树的题目。