

Assignment #2: 编程练习

Updated 0953 GMT+8 Feb 24, 2024

2024 spring, Compiled by 天幕 化学与分子工程学院

说明:

1) The complete process to learn DSA from scratch can be broken into 4 parts:

- Learn about Time and Space complexities
- Learn the basics of individual Data Structures
- Learn the basics of Algorithms
- Practice Problems on DSA

2) 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），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。

3) 课程网站是Canvas平台, <https://pku.instructure.com>, 学校通知3月1日导入选课名单后启用。**作业写好后，保留在自己手中，待3月1日提交。**

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”区有上传的md或者doc附件。

4) 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 11 23H2

Python编程环境: Visual Studio Code 1.86.2

1. 题目

27653: Fraction类

http://cs101.openjudge.cn/2024sp_routine/27653/

思路: 思路: 比较直接，通过重载__add__与__str__方法简化代码，具体逻辑中使用了辗转相除法进行约分。

代码

```
1 class Fraction(object):
2     def __init__(self, a:int, b:int):#a/b
3         self.a = a
4         self.b = b
```

```

5     def __add__(self, other):
6         a = self.a * other.b + other.a * self.b
7         b = self.b * other.b
8         x, y = a, b
9         while b > 0:
10            a, b = b, a % b
11            x, y = x//a, y//a
12            return(Fraction(x, y))
13     def __str__(self):
14         if self.a == 0:
15             return 0
16         elif self.b ==1:
17             return(str(self.a))
18         else:
19             return(str(self.a) + "/" + str(self.b))
20
21 a1, b1, a2, b2 = map(int, input().split())
22 x, y = Fraction(a1, b1), Fraction(a2, b2)
23 print(x + y)

```

代码运行截图

状态: **Accepted**

源代码

```

class Fraction(object):
    def __init__(self, a:int, b:int):#a/b
        self.a = a
        self.b = b
    def __add__(self, other):
        a = self.a * other.b + other.a * self.b
        b = self.b * other.b
        x, y = a, b
        while b > 0:
            a, b = b, a % b
            x, y = x//a, y//a
            return(Fraction(x, y))
    def __str__(self):
        if self.a == 0:
            return 0
        elif self.b ==1:
            return(str(self.a))
        else:
            return(str(self.a) + "/" + str(self.b))

a1, b1, a2, b2 = map(int, input().split())
x, y = Fraction(a1, b1), Fraction(a2, b2)
print(x + y)

```

04110: 圣诞老人的礼物-Santa Clau's Gifts

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

思路：既然可以散装带走那么只需考虑尽量带走单价高的糖果。

代码

```
1 x, y = map(int, input().split())
2 listy = []
3 for _ in range(x):
4     v, w = map(int, input().split())
5     listy.append([v/w, w, v])
6 listy.sort(key= lambda x: x[0], reverse= True)
7 sumy = 0
8 i = 0
9 while y > 0 and i < x:
10     cost = listy[i][1]
11     if y >= cost:
12         y -= cost
13         sumy += listy[i][2]
14     else:
15         sumy += listy[i][0]*y
16         y = 0
17     i += 1
18 print("%.1f"%sumy)
```

代码运行截图

#44019771提交状态

状态: Accepted

源代码

```
x, y = map(int, input().split())
listy = []
for _ in range(x):
    v, w = map(int, input().split())
    listy.append([v/w, w, v])
listy.sort(key= lambda x: x[0], reverse= True)
sumy = 0
i = 0
while y > 0 and i < x:
    cost = listy[i][1]
    if y >= cost:
        y -= cost
        sumy += listy[i][2]
    else:
        sumy += listy[i][0]*y
        y = 0
    i += 1
print("%.1f"%sumy)
```

18182: 打怪兽

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

思路：思路直白，能使用就使用，优先使用伤害较高的。相比计算概论使用两次sort()的原始操作，改为使用lambda函数构造元组满足双关键字排序。

代码

```
1 n = int(input())
2 for _ in range(n):
3     skills, maxskillspertime, hp = map(int, input().split())
4     sety = [[int(x) for x in input().split()] for _ in range(skills)]
5     sety.sort(key= lambda x: (x[0], -x[1]))
6     def letsfight(sety, maxskillspertime, hp):
7         checky = 0
8         timey = 1
9         timeynow = 1
10        for i in range(skills):
11            timeynow = sety[i][0]
12            if timeynow != timey:
13                hp -= sety[i][1]
14                timey = timeynow
15            checky = 1
```

```

16         elif checky < maxskillspertime:
17             hp -= sety[i][1]
18             checky += 1
19         if hp < 1:
20             return(timey)
21     return("alive")
22     print(letsfight(sety, maxskillspertime, hp))

```

代码运行截图

状态: Accepted

源代码

```

n = int(input())
for _ in range(n):
    skills, maxskillspertime, hp = map(int, input().split())
    sety = [[int(x) for x in input().split()] for _ in range(skills)]
    sety.sort(key= lambda x: (x[0], -x[1]))
    def letsfight(sety, maxskillspertime, hp):
        checky = 0
        timey = 1
        timeynow = 1
        for i in range(skills):
            timeynow = sety[i][0]
            if timeynow != timey:
                hp -= sety[i][1]
                timey = timeynow
                checky = 1
            elif checky < maxskillspertime:
                hp -= sety[i][1]
                checky += 1
            if hp < 1:
                return(timey)
        return("alive")
    print(letsfight(sety, maxskillspertime, hp))

```

230B. T-primes

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

思路：使用欧拉筛法生成素数表。改用浮点数的.is_integer()方法判断完全平方。好像结果不如题解的埃氏筛法，或许与测试数据有关。

代码

```
1  from math import sqrt
2
3  N = 1000000
4  primeList = []
5  pOn = [False] * 2 + [True] * (N-1)
6  p = 2
7  while p <= N:
8      if pOn[p]:
9          primeList.append(p)
10         for i in primeList:
11             x = p * i
12             if x > N:
13                 break
14             pOn[x] = False
15             if p % i == 0:
16                 break
17         p += 1
18
19  def istprime(n:int):
20      x = sqrt(n)
21      return x.is_integer() and pOn[int(x)]
22
23  yes = "YES"
24  no = "NO"
25
26  _ = int(input())
27  for x in list(map(int, input().split())):
28      print([no, yes][istprime(x)])
```

代码运行截图

```
from math import sqrt, ceil, floor

N = 1000000
primeList = []
pOn = [False] * 2 + [True] * (N-1)
p = 2
while p <= N:
    if pOn[p]:
        primeList.append(p)
        for i in primeList:
            x = p * i
            if x > N:
                break
            pOn[x] = False
            if p % i == 0:
                break
        p += 1

def istprime(n:int):
    x = sqrt(n)
    return x.is_integer() and pOn[int(x)]

yes = "YES"
no = "NO"

_ = int(input())
for x in list(map(int, input().split())):
    print([no, yes][istprime(x)])
```

1364A. XXXXX

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

思路：分别标记最左侧与最右侧出现的第一个不能被 x 整除的数，并比较大小。

代码

```
1  for _ in range(int(input())):
2      n, x = map(int, input().split())
3      lIndex = -1
4      rIndex = -1
5      sum = 0
6      listA = list(map(int, input().split()))
7      length = len(listA)
8      for i in range(length):
9          rmdr = listA[i] % x
10         if rmdr != 0:
11             if lIndex == -1:
12                 lIndex = i
13                 rIndex = i
14                 rIndex = max(rIndex, i)
15         sum += rmdr
```

```

16     if sum % x != 0:
17         print(n)
18     elif lIndex == -1:
19         print(-1)
20     else:
21         print(max(length-lIndex-1, rIndex))

```

代码运行截图

By Tian_mi, contest: Codeforces Round 649 (Div. 2), problem: (A) XXXXX, **Accepted**, #, [Copy](#)

```

for _ in range(int(input())):
    n, x = map(int, input().split())
    lIndex = -1
    rIndex = -1
    sum = 0
    listA = list(map(int, input().split()))
    length = len(listA)
    for i in range(length):
        rmdr = listA[i] % x
        if rmdr != 0:
            if lIndex == -1:
                lIndex = i
                rIndex = i
            rIndex = max(rIndex, i)
        sum += rmdr
    if sum % x != 0:
        print(n)
    elif lIndex == -1:
        print(-1)
    else:
        print(max(length-lIndex-1, rIndex))

```

18176: 2050年成绩计算

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

思路：套壳T-primes，与之相同。

代码

```

1  from math import sqrt
2
3  N = 10000
4  primeList = []
5  pOn = [False] * 2 + [True] * (N-1)
6  p = 2
7  while p <= N:
8      if pOn[p]:
9          primeList.append(p)
10     for i in primeList:

```



```
11     x = p * i
12     if x > N:
13         break
14     pOn[x] = False
15     if p % i == 0:
16         break
17     p += 1
18
19 def istprime(n:int):
20     x = sqrt(n)
21     return x.is_integer() and pOn[int(x)]
22
23 yes = "YES"
24 no = "NO"
25
26 m, _ = map(int, input().split())
27 for i in range(m):
28     vs = list(map(int, input().split()))
29     v = 0
30     for i in vs:
31         if istprime(i):
32             v += i
33     if v == 0:
34         print(0)
35     else:
36         print(format(v/len(vs), ".2f"))
```

代码运行截图

状态: Accepted

源代码

```
from math import sqrt

N = 10000
primeList = []
pOn = [False] * 2 + [True] * (N-1)
p = 2
while p <= N:
    if pOn[p]:
        primeList.append(p)
        for i in primeList:
            x = p * i
            if x > N:
                break
            pOn[x] = False
            if p % i == 0:
                break
        p += 1

def istprime(n:int):
    x = sqrt(n)
    return x.is_integer() and pOn[int(x)]

yes = "YES"
no = "NO"

m, _ = map(int, input().split())
for i in range(m):
    vs = list(map(int, input().split()))
    v = 0
    for i in vs:
        if istprime(i):
            v += i
    if v == 0:
        print(0)
    else:
        print(format(v/len(vs), ".2f"))
```

2. 学习总结和收获

复健第二周。虽然也有很多做过的题，但是还是学到了一些新的东西，主要是来自T-primes和2050年成绩计算这两道题。前年计概课上230B就难到了一堆人，当时写的代码（参见上周的Goldbach Conjecture，只是做了偶数的排除）可以说是卡线过的（还要感谢PyPy，因为不用过不了），再次见面想着优化一下，结果发现最后投进去很多时间，看了一些数论相关的算法，比如Miller Rabin（加个列表存的话过得了230B但是过不了2050年成绩计算），也看了埃氏筛和欧氏筛，最后使用欧氏筛AC，虽然效果好像没有埃氏筛好，但也算是学到了新东西吧。

