

Assignment #5: 20251009 cs101 Mock Exam寒露第二天

Updated 1651 GMT+8 Oct 9, 2025

2025 fall, Compiled by 郭旭杰、化学与分子工程学院

说明:

1. 解题与记录:

对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge, Codeforces, LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora <https://typoraio.cn> 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。

2. 提交安排: **提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的本人头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。

3. 延迟提交: 如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

E29895: 分解因数

implementation, <http://cs101.openjudge.cn/practice/29895/>

耗时: 10min

思路: 较简单, 10min速通。

需要注意的一点是不能直接找n最大的真因数，否则会TLE，而是用n整除其除了1以外最小的因数以节省时间。

代码

```
# Python3
n = int(input())
i = 2
while n % i != 0:
    i += 1
print(n // i)
```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

29895:分解因数

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

描述

已知正整数 n (合数) 是两个不同的因数的乘积 (可能有多种分解方式), 试求出其中最大的那个因数 (不包含自己, 原题说法有疏漏)。

输入

输入一个正整数 n 。 $1 \leq n \leq 10^{10}$

输出

输出一个正整数 p , 即较大的那个因数。

样例输入

21

样例输出

7

提示

implementation

来源

https://www.luogu.com.cn/problem/P1075(TA-hhy)

查看 提交 统计 提问

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

#50282518提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
n = int(input())
i = 2
while n % i != 0:
    i += 1
print(n // i)
```

基本信息

#: 50282518

题目: 29895

提交人: 2500011906

内存: 3576kB

时间: 20ms

语言: Python3

提交时间: 2025-10-09 20:54:43

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

E29940: 机器猫斗恶龙

greedy, <http://cs101.openjudge.cn/practice/29940/>

耗时: 15min

思路: 较简单, 15min速通。

简单的贪心算法, 就是假设机器猫的血量可以为负值或0, 一开始的血量为0, 然后将其血量的变化“平移”, 使其最小值为1, 输出“平移”的距离即可。

代码

```
n = int(input())
js = list(map(int, input().split()))
sgm = 0
s = []
for j in js:
    sgm += j
    s.append(sgm)
m = min(s)
if m < 0:
    res = 1 - m
else:
    res = 1
print(res)
```

代码运行截图 (至少包含有"Accepted")

29940:机器猫斗恶龙

查看

提交

统计

提问

总时间限制: 1000ms 内存限制: 65536kB

描述

机器猫出门斗恶龙了！他需要通过 n 个关卡。

每个关卡要么是与怪物战斗，扣除一定的血量；要么是营地，给机器猫增加一定的血量。

在旅途中，机器猫任意时刻的血量不能低于或等于 0。问机器猫至少需要多少的初始血量，才能完成任务。

血量为正整数。

输入

第一行，一个正整数 n，表示关卡数量。

第二行，共 n 个整数 a_i，表示每个关卡。

- 若 a_i>0，则表示这个关卡是营地，增加 a_i 的血量

- 若 a_i<0，则表示这个关卡是战斗，机器猫血量代价为 a_i

所有数据满足 $n \leq 100000$, $1 \leq |a_i| \leq 1000$ 。

输出

仅一行，一个正整数，表示机器猫需要的初始血量。

样例输入

```
sample1 input:
3
-100 -200 -300

sample1 output:
601
```

全局题号 **29940**

添加于 **2025-10-09**

提交次数 **111**

尝试人数 **73**

通过人数 **73**

Other language versions

[English](#)

你的提交记录

#	结果	时间
1	Accepted	2025-10-09



CS101 / 题库（包括计概、数算题目）

题目

排名

状态

提问

#50282853提交状态

查看

提交

统计

提问

状态: Accepted

源代码

```
n = int(input())
js = list(map(int, input().split()))
sgm = 0
s = []
for j in js:
    sgm += j
    s.append(sgm)
m = min(s)
if m < 0:
    res = 1-m
else:
    res = 1
print(res)
```

基本信息

#: 50282853
题目: 29940
提交人: 郭旭杰
内存: 14816kB
时间: 58ms
语言: Python3
提交时间: 2025-10-09 21:08:56

M29917: 牛顿迭代法

implementation, <http://cs101.openjudge.cn/practice/29917/>

耗时: 前前后后加起来一个多小时。

思路: 这题不难, 照着题目意思写代码就可以了。但是我肯定不可能在考试中将其做对。

只因我当时对不定行输入一无所知。一直要么WA要么RE。

不定行输入:

while True:

try:

代码

except:

break

代码

```
while True:
    try:
        t = float(input())
        x = 1
        y = 1 - t
        k = 0
        while abs(y/(2*x)) > 1e-6:
            x -= y / (2*x)
            y = x**2 - t
            k += 1
```

```
x -= y / (2*x)
k += 1
print(k, "%.2f" % x)
except:
    break
```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号



CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

29917:牛顿迭代法

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

描述

用牛顿法求一个数的平方根。

要求：
初始取 1 为近似根，因此第一次迭代是过曲线上横坐标为 1 的点作切线。
迭代终止条件为，上一次迭代求出的近似根，与这一次迭代求出的更精确的近似根的差，小于等于 1E-6。
输出迭代终止时的迭代次数。

迭代公式：
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

输入

多行，每行可能是一个整数或一个小数。

输出

对输入的每行，输出一行，为对应的迭代次数和近似根，用空格分隔。

迭代次数为一个整数。
根为一个浮点数，保留小数点后两位。

样例输入

12
25
144

样例输出

全局题号 29917

添加于 2025-10-09

提交次数 362

尝试人数 112

通过人数 104

你的提交记录

#	结果	时间
1	Accepted	2025-10-10

#50285760提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
while True:
    try:
        t = float(input())
        x = 1
        y = 1 - t
        k = 0
        while abs(y / (2 * x)) > 1e-6:
            x -= y / (2 * x)
            y = x ** 2 - t
            k += 1
        x -= y / (2 * x)
        k += 1
        print(k, "%.2f" % x)
    except:
        break
```

基本信息

#: 50285760
题目: 29917
提交人: 2500011906
内存: 3592kB
时间: 24ms
语言: Python3
提交时间: 2025-10-10 07:51:05

M29949: 贪婪的哥布林

greedy, <http://cs101.openjudge.cn/practice/29949/>

耗时: 看书学会思路, 不断优化终于成功。

思路: 需要对二维数组有较灵活的运用能力, 有一定难度。考试时候对二维数组理解不深, 总是想用Hash Table, 就没能在考试期间完成。

这题是我在看完《算法图解》之后完成的。

音响 (S)

笔记本电脑 (L)

吉他 (G)

	1	2	3	4
音响 (S)	∅	∅	∅	\$3000
笔记本电脑 (L)	∅	∅	\$2000	\$3000
吉他 (G)	\$1500	\$1500	\$2000	\$3500

答案没有变化。也就是说，各行的排列顺序无关紧要。

9.2.3 可以逐列而不是逐行填充网格吗

自己动手试试吧！就这个问题而言，这没有任何影响，但对于其他问题，可能有影响。

9.2.4 增加一件更小的商品将如何呢

假设你还可以偷一条项链，它重0.5磅，价值1000美元。前面的网格都假设所有商品的重量为整数，但现在你决定把项链给偷了，因此余下的容量为3.5磅。在3.5磅的容量中，可装入的商品的最大价值是多少呢？不知道！因为你只计算了容量为1磅、2磅、3磅和4磅的背包可装下的商品的最大价值。现在，你需要知道容量为3.5磅的背包可装下的商品的最大价值。

由于项链的加入，你需要考虑的粒度更细，因此必须调整网格。

吉他

音响

笔记本电脑

项链

	0.5	1	1.5	2	2.5	3	3.5	4
吉他								
音响								
笔记本电脑								
项链								


9.2.5 可以偷商品的一部分吗 *greedy Goblin*

假设你在杂货店行窃，可偷成袋的扁豆和大米，但如果整袋装不下，可打开包装，再将背包倒满。在这种情况下，不再是要么偷要么不偷，而是可偷商品的一部分。如何使用动态规划来处理这种情形呢？


答案是没法处理。使用动态规划时，要么考虑拿走整件商品，要么考虑不拿，而没法判断该不该拿走商品的一部分。

但使用贪婪算法可轻松地处理这种情况！首先，尽可能多地拿价值最高的商品；如果拿光了，再尽可能多地拿价值次高的商品，以此类推。


例如，假设有如下商品可供选择。



燕麦
每磅6美元



木豆
每磅3美元



大米
每磅2美元

燕麦比其他商品都值钱，因此要尽量往背包中装燕麦！如果能够在背包

《算法图解》详细讲述了用动态规划处理“小偷问题”，其中略提及的用贪婪算法处理“偷商品的一部分”和这题几乎一模一样。就是按照矿石的价格排序，依次拿走，直至矿石被拿完或背包装满。

代码

```
#
n,m = map(int, input().split())
js = []
sgm = 0
earn = 0
for _ in range(n):
    v,w = map(int, input().split())
    js.append((v / w,v,w))
js.sort(reverse=True)
for j in js:
    if sgm + j[2] <= m:
        sgm += j[2]
        earn += j[1]
    else:
        earn += j[0] * (m - sgm)
        sgm = m
        break
print("%.2f" % earn)
```


代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号



CS101 / 题库 (包括计概、数算题目)

题目

排名

状态

提问

29949:贪婪的哥布林

查看

提交

统计

提问

总时间限制: 1000ms 内存限制: 65536kB

描述

艾泽拉斯的一只地精 (Goblin, 哥布林) 意外地发现了一个废弃的矿洞, 里面有数堆不同的矿石 (例如铋矿、铋爪矿、亚基矿等)。每一堆矿石都有其总价值和总重量。

哥布林的背包最多只能装下总重量为 W 的物品。幸运的是, 这些矿石都是粉末状的, 他可以从任何一堆矿石中取出任意一部分 (例如取走一堆铋矿的 1/3)。

作为一名贪婪的哥布林, 他希望背包里装的矿石总价值尽可能高。请你帮他计算出这个最大总价值。

输入

第一行包含两个整数 N 和 M (1 <= N <= 100, 1 <= M <= 10000), 分别代表矿石的堆数和背包的载重上限。

接下来 N 行, 每行包含两个整数 v 和 w (1 <= v, w <= 1000), 分别代表这堆矿石的总价值和总重量。

输出

输出一个浮点数, 代表能装入背包的最大总价值。结果保留两位小数。

样例输入

3 50
60 10
100 20
120 30

样例输出

240.00

提示

全局题号 29949

添加于 2025-10-09

提交次数 175

尝试人数 88

通过人数 85

你的提交记录

#	结果	时间
3	Accepted	2025-10-12
2	Runtime Error	2025-10-11
1	Runtime Error	2025-10-11

#50323429提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
n,m = map(int, input().split())
js = []
sgm = 0
earn = 0
for _ in range(n):
    v,w = map(int, input().split())
    js.append((v / w, v, w))
js.sort(reverse=True)
for j in js:
    if sgm + j[2] <= m:
        sgm += j[2]
        earn += j[1]
    else:
        earn += j[0] * (m - sgm)
        sgm = m
        break
print("%.2f" % earn)
```

基本信息

#: 50323429
题目: 29949
提交人: 郭旭杰
内存: 3620kB
时间: 22ms
语言: Python3
提交时间: 2025-10-12 14:07:02

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

M29918: 求亲和数

implementation, <http://cs101.openjudge.cn/practice/29918/>

耗时: 陆续做了一个下午和一个晚上

思路: 这题很难了, 考试时我想要先用较慢的代码输出所有结果, 代码后台运算的时候我先做其它题目, 然后对这一题暴力讨论。

结果我的代码复杂度太高, n 取最大值的时候Pycharm楞是后台运算了一个小时也只输出两组结果, 只能暂且放弃。

后来在讨论中我看到了其他同学的算法, 这种算法有大量诸如将 n 的讨论缩减到讨论到 $n^{*}0.5+1$ 等优化, 大大缩减了运行时间, 提高了效率。

代码

#我的 $O(n^{*}3)$ 的代码, 当 $n=100000$ 时, 一小时只能输出两组亲和数。

```
def sumf(x):
    sumf_r = 0
    for _ in range(1, x):
        if x % _ == 0:
            sumf_r += _
    return sumf_r

n = int(input())
for i in range(1, n+1):
    for j in range(i+1, n+1):
        if sumf(i) == j and sumf(j) == i:
```

```

        print(i,j)
        continue

#=====
#同学提供了一种快得多的算法 $O(n\log n)$ ，虽然还是会TLE。
t = int(input())
n = [0]
for i in range(2,100001):
    d = 0
    for j in range(1,int(i**0.5)+1):
        if i % j == 0:
            if j ** 2 == i or j == 1:
                d += j
            else:
                d += j + i // j
    n.append(d)
for i in range(2,100001):
    if n[i-1] > 99999:
        continue
    elif n[n[i-1]-1] == i:
        if i < n[i-1] <= t:
            print(i,n[i-1])

#=====
#最终AC使用的方法：先用较快算法算出结果，然后就像下面这么做。这种算法 $O(1)$ ，实打实的brute force。
n = int(input())
if n >= 284:
    print(220,284)
if n >= 1210:
    print(1184,1210)
if n >= 2924:
    print(2620,2924)
if n >= 5564:
    print(5020,5564)
if n >= 6368:
    print(6232,6368)
if n >= 10856:
    print(10744,10856)
if n >= 14595:
    print(12285,14595)
if n >= 18416:
    print(17296,18416)
if n >= 76084:
    print(63020,76084)
if n >= 66992:
    print(66928,66992)
if n >= 71145:
    print(67095,71145)
if n >= 87633:
    print(69615,87633)
if n >= 88730:
    print(79750,88730)

```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

 **CS101 / 题库 (包括计概、数算题目)**

题目 排名 状态 提问

29918:求亲和数

总时间限制: 1000ms 内存限制: 65536kB

描述

遥远的古代, 人们发现某些自然数之间有特殊的关系: 设有两个数a和b, 若a的所有除本身以外的因数之和等于b, b的所有除本身以外的因数之和等于a, 则称a、b是一对亲和数。

例如220和284:

220的真因数包括: 1,2,4,5,10,11,20,22,44,55,110.

$1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284$

284的真因数包括: 1,2,4,71,142.

$1 + 2 + 4 + 71 + 142 = 220$

所以220和284是一对亲和数。

输入

一个正整数n, $1 \leq n \leq 100000$ 。

输出

所有亲和数对"a b", 满足a和b均小于等于n。

每个亲和数对占一行, 两个数之间用一个空格隔开, 较小数在前, 较大数在后。

对于多个亲和数对, 以较小数递增的顺序输出它们。

样例输入

```
1500
```

样例输出

```
220 284
1184 1210
```

提示

```
def sumf(x):
    sumf_r = 0
    for _ in range(1, x):
        if x % _ == 0:
            sumf_r += _
    return sumf_r

n = int(input())
for i in range(1, n+1):
    for j in range(i+1, n+1):
        if sumf(i) == j and sumf(j) == i:
            print(i,j)
            continue
```

查看

提交 统计 提问

全局题号 **29918**

添加于 **2025-10-09**

提交次数 **411**

尝试人数 **102**

通过人数 **99**

你的提交记录

#	结果	时间
3	Accepted	2025-10-09
2	Time Limit Exceeded	2025-10-09
1	Wrong Answer	2025-10-09



CS101 / 题库 (包括计概、数算题目)

[题目](#) [排名](#) [状态](#) [提问](#)

#50282560提交状态

[查看](#) [提交](#) [统计](#) [提问](#)状态: **Time Limit Exceeded**

源代码

```
t = int(input())
n = [0]
for i in range(2,100001):
    d = 0
    for j in range(1,int(i**0.5)+1):
        if i % j == 0:
            if j ** 2 == i or j == 1:
                d += j
            else:
                d += j + i // j
    n.append(d)
for i in range(2,100001):
    if n[i-1] > 99999:
        continue
    elif n[n[i-1]-1]== i:
        if i < n[i-1] <= t:
            print(i,n[i-1])
```

基本信息

#: 50282560
题目: 29918
提交人: 2500011906
内存: 7280kB
时间: 2358ms
语言: Python3
提交时间: 2025-10-09 20:56:06

#50282810提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
n = int(input())
if n >= 284:
    print(220,284)
if n >= 1210:
    print(1184,1210)
if n >= 2924:
    print(2620,2924)
if n >= 5564:
    print(5020,5564)
if n >= 6368:
    print(6232,6368)
if n >= 10856:
    print(10744,10856)
if n >= 14595:
    print(12285,14595)
if n >= 18416:
    print(17296,18416)
if n >= 76084:
    print(63020,76084)
if n >= 66992:
    print(66928,66992)
if n >= 71145:
    print(67095,71145)
if n >= 87633:
    print(69615,87633)
if n >= 88730:
    print(79750,88730)
```

基本信息

#: 50282810
题目: 29918
提交人: 2500011906
内存: 3636kB
时间: 20ms
语言: Python3
提交时间: 2025-10-09 21:07:00

T29947:校门外的树又来了 (选做)

<http://cs101.openjudge.cn/practice/29947/>

耗时: 从10月9日到10月12日, 陆续做了4天, 不断修改终于成功。

思路:

代码

```
n,line = list(map(int, input().split()))
tree = n + 1
cut = []
for _ in range(line):
    a,b = list(map(int, input().split()))
    cut.append((a,b))
cut.sort(key=lambda x:x[0])

init = 0
kil = 0
```

```
while init <= line-1:
    at = cut[init][0]
    bt = cut[init][1]
    while init < line-1 and cut[init+1][0] <= bt:
        init += 1
        bt = max(cut[init][1], cut[init-1][1],bt)
    init += 1
    kil += bt - at + 1

print(tree - kil)
```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

29947:校门外的树又来了

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

描述

某校大门外长度为L的马路上有一排树，每两棵相邻的树之间的间隔都是1米。我们可以把马路看成一个数轴，马路的一端在数轴0的位置，另一端在L的位置；数轴上的每个整数点，即0，1，2，……，L，都种有一棵树。马路上有一些区域要用来建地铁，这些区域用它们在数轴上的起始点和终止点表示。已知任一区域的起始点和终止点的坐标都是整数，区域之间可能有重合的部分。现在要把这些区域中的树（包括区域端点处的两棵树）移走。你的任务是计算将这些树都移走后，马路上还有多少棵树。

输入

输入的第一行有两个整数L（1 <= L <= 10^9）和 M（1 <= M <= 100），L代表马路的长度，M代表区域的数目，L和M之间用一个空格隔开。接下来的M行每行包含两个不同的整数，用一个空格隔开，表示一个区域的起始点和终止点的坐标。

输出

输出包括一行，这一行只包含一个整数，表示马路上剩余的树的数目。

样例输入

500 3
150 300
100 200
470 471

样例输出

298

来源

yan

全局题号 29947

添加于 2025-10-09

提交次数 264

尝试人数 95

通过人数 86

你的提交记录

#	结果	时间
12	Accepted	2025-10-12
11	Wrong Answer	2025-10-12
10	Wrong Answer	2025-10-12
9	Wrong Answer	2025-10-12
8	Wrong Answer	2025-10-11
7	Wrong Answer	2025-10-11
6	Wrong Answer	2025-10-11
5	Wrong Answer	2025-10-11
4	Wrong Answer	2025-10-11
3	Time Limit Exceeded	2025-10-11
2	Memory Limit Exceeded	2025-10-11
1	Memory Limit Exceeded	2025-10-10

#50325390提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
n,line = list(map(int, input().split()))
tree = n + 1
cut = []
for _ in range(line):
    a,b = list(map(int, input().split()))
    cut.append((a,b))
cut.sort(key=lambda x:x[0])

init = 0
kil = 0
while init <= line-1:
    at = cut[init][0]
    bt = cut[init][1]
    while init < line-1 and cut[init+1][0] <= bt:
        init += 1
        bt = max(cut[init][1], cut[init-1][1],bt)
    init += 1
    kil += bt - at + 1

print(tree - kil)
```

基本信息

#: 50325390

题目: 29947

提交人: 郭旭杰

内存: 3636kB

时间: 19ms

语言: Python3

提交时间: 2025-10-12 16:07:44

2. 学习总结和收获

通过这次作业，我学会了基本的贪心算法（E29895: 分解因数；E29940: 机器猫斗恶龙，M29949: 贪婪的哥布林；T28701炸鸡排）重点掌握了超时的优化方法（E29895: 分解因数；M29918: 求亲和数；T29947:校门外的树又来了）学会了不定行输入（M29917: 牛顿迭代法）对二维数组有了更深刻的认识（M29949: 贪婪的哥布林；T29947:校门外的树又来了）

这次考试让我看到了我的真实水平，还需要加强练习。

如果作业题目简单，有否额外练习题目，比如：OJ“计概2025fall每日选做”、CF、LeetCode、洛谷等网站题目。

周五夜间参加CodeForces比赛做出2题，九千多名，仍然是新手(Newbie)。

9839 LittleBeetroot

1133

48800:06

64500:35

-5

PROBLEMS

SUBMIT CODE

MY SUBMISSIONS

STATUS

HACKS

ROOM

STANDINGS

CUSTOM INVOCATION

Problems

#	Name		
A	Circle of Apple Trees	standard input/output 1 s, 256 MB	x20318
B	Bitwise Reversion	standard input/output 1.5 s, 256 MB	x17938
C	Symmetrical Polygons	standard input/output 2 s, 256 MB	x8306
D	Not Alone	standard input/output 2 s, 256 MB	x3882
E	Zero Trailing Factorial	standard input/output 3 s, 512 MB	x550
F	Odd Queries on Odd Array	standard input/output 10 s, 1024 MB	x102

Complete problemset

Codeforces Round 1057 (Div. 2)

Finished

Practice

Virtual participation

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use

Codeforces Hot News!

Wow! Coder **LittleBeetroot** competed in Codeforces Round 1057 (Div. 2) and gained +219 rating points taking place 931!

Share it!

A. Circle of Apple Trees

time limit per test: 1 second
memory limit per test: 256 megabytes

There are n apple trees arranged in a circle. Each tree bears exactly one apple, and the beauty of the apple on the i -th tree is given by b_i for all $1 \leq i \leq n$. You start in front of tree 1.

At each tree, you may choose to either eat the apple or skip it. After making your choice, you move to the next tree: from tree i , you move to tree $i + 1$ for $1 \leq i \leq n - 1$, and from tree n , you move back to tree 1. This process continues indefinitely as you move through the trees in a cycle.

However, you have a special condition: you may only eat an apple if its beauty is strictly greater than the beauty of the last apple you ate. For example, if $b = [2, 1, 2, 3]$ and you eat the apple on tree 1 (beauty 2), you cannot eat the apples on trees 2 and 3 because their beauties are not greater than 2. However, you may eat the apple on tree 4 since $b_4 = 3 > 2$.

Note that you are allowed to skip an apple when you first encounter it, and you can choose to eat it later on a subsequent cycle.

Your task is to determine the maximum number of apples you can eat if you make optimal decisions on when to eat or skip each apple.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 500$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \leq n \leq 100$) — the number of apple trees.

The second line of each test case contains n integers b_1, b_2, \dots, b_n ($1 \leq b_i \leq n$) — the beauty of the apples on the trees.

Note that there are no constraints on the sum of n over all test cases.

Output

For each test case, output a single integer representing the maximum number of apples you can eat.

Example

input

```
3
4
2 2 2 2
5
1 4 5 1 2
6
5 4 2 1 2 3
```

output

```
1
4
5
```

Codeforces Round 1057 (Div. 2)

Finished
Practice

Virtual participation

Virtual contest is a way to take part in past contests, as close as possible to participation on time. It is supported only in ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you: you will see the problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you: you will see the problems in the archive. Never use someone else's code, read the tutorials or communicate with other people during a virtual contest.

Clone Contest to Mashup

You can clone this contest to a mashup.

Submit?

Language: PyPy 3.10 (7.3.15, 64bit)

Choose file: 选择文件 未选择文件

Submit

Last submissions

Submission	Time	Verdict
342930479	Oct/10/2025 17:41	Accepted

Problem tags

B. Bitwise Reversion

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes

You are given three non-negative integers x , y and z . Determine whether there exist three non-negative integers a , b and c satisfying the following three conditions:

- $a \& b = x$
- $b \& c = y$
- $a \& c = z$

where $\&$ denotes the bitwise AND operation.

The first and only line of each test case contains three integers x , y and z ($0 \leq x, y, z \leq 10^9$) — the target values of $a \& b$, $b \& c$ and $a \& c$, respectively.

Output

For each test case, output "YES" if there exists three non-negative integers a , b , and c satisfying the above conditions, and "NO" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "YES", "yes", "YEs", and "YeS" will be recognized as positive responses.

Example

input

```
3 3 3
2 0 0
4 8 12
9 98 124
```

output

```
YES
YES
NO
NO
```

Note

In the first test case, $a = 3$, $b = 5$, and $c = 9$ satisfies the condition as $3 \& 5 = 1$, $5 \& 9 = 1$, and $3 \& 9 = 1$.

In the second test case, $a = 7$, $b = 3$, and $c = 22$ satisfies the condition as $7 \& 3 = 3$, $3 \& 22 = 2$, and $7 \& 22 = 6$.

In the third test case, it can be proven that there are no three non-negative integers a , b , and c such that $a \& b = 4$, $b \& c = 8$, and $a \& c = 12$.

Codeforces Round 1057 (Div. 2)

Finished
Practice

Virtual participation

Virtual contest is a way to take part in past contests, as close as possible to participation on time. It is supported only in ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you: you will see the problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you: you will see the problems in the archive. Never use someone else's code, read the tutorials or communicate with other people during a virtual contest.

Clone Contest to Mashup

You can clone this contest to a mashup.

Submit?

Language: PyPy 3.10 (7.3.15, 64bit)

Choose file: 选择文件 未选择文件

Submit

Last submissions

Submission	Time	Verdict
342930479	Oct/10/2025 17:41	Accepted

Problem tags

General

#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged
342961547	Contestant: LittleBeetroot	21538 - 18	PyPy 3-64	Accepted	312 ms	8984 KB	2025-10-10 18:10:06	2025-10-10 20:20:03

Source

```
1 = int(input())
2 = int(input())
3 = int(input())
4 = int(input())
5 = int(input())
6 = int(input())
7 = int(input())
8 = int(input())
9 = int(input())
10 = int(input())
11 = int(input())
12 = int(input())
13 = int(input())
14 = int(input())
15 = int(input())
16 = int(input())
17 = int(input())
18 = int(input())
19 = int(input())
20 = int(input())
21 = int(input())
22 = int(input())
23 = int(input())
24 = int(input())
25 = int(input())
26 = int(input())
27 = int(input())
28 = int(input())
29 = int(input())
30 = int(input())
31 = int(input())
32 = int(input())
33 = int(input())
34 = int(input())
35 = int(input())
36 = int(input())
37 = int(input())
38 = int(input())
39 = int(input())
40 = int(input())
41 = int(input())
42 = int(input())
43 = int(input())
44 = int(input())
45 = int(input())
46 = int(input())
47 = int(input())
48 = int(input())
49 = int(input())
50 = int(input())
51 = int(input())
52 = int(input())
53 = int(input())
54 = int(input())
55 = int(input())
56 = int(input())
57 = int(input())
58 = int(input())
59 = int(input())
60 = int(input())
61 = int(input())
62 = int(input())
63 = int(input())
64 = int(input())
65 = int(input())
66 = int(input())
67 = int(input())
68 = int(input())
69 = int(input())
70 = int(input())
71 = int(input())
72 = int(input())
73 = int(input())
74 = int(input())
75 = int(input())
76 = int(input())
77 = int(input())
78 = int(input())
79 = int(input())
80 = int(input())
81 = int(input())
82 = int(input())
83 = int(input())
84 = int(input())
85 = int(input())
86 = int(input())
87 = int(input())
88 = int(input())
89 = int(input())
90 = int(input())
91 = int(input())
92 = int(input())
93 = int(input())
94 = int(input())
95 = int(input())
96 = int(input())
97 = int(input())
98 = int(input())
99 = int(input())
100 = int(input())
```

用在《算法图解》上学习的递归函数完成了“炸鸡排”这一道OJ上较难的题目。

惊讶地发现：用"% .3f" % result就会报错，用f"{result:.3f}"就能AC。

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

28701:炸鸡排

查看 提交 统计 提问

总时间限制: 1000ms 内存限制: 65536kB

描述

小P买了n块鸡排, 想将它们做成美味的炸鸡排, 其中第i块鸡排需要t[i]秒炸熟。小P只有一个炸锅, 炸锅内可以放置k(k≤n)块鸡排。小P是个完美主义者, 他要求任意时刻炸锅内必须恰有k块鸡排。他可以在任意时刻改变锅内正在炸的鸡排, 只需保证已经熟了的鸡排不能继续留在锅中。小P希望知道炸鸡排最多可以持续多少时间。

例如, 小P的三块鸡排需要分别需要1,1,1的时间炸熟, 炸锅内需要放置2块鸡排。那么他决定在第一个0.5秒炸1和2两块鸡排, 第二个0.5秒炸2和3两块鸡排, 第三个0.5秒炸1和3两块鸡排, 共持续了1.5秒。

输入

第一行输入两个正整数n和k, k≤n
第二行输入n个正整数, 代表n块鸡排分别需要炸熟的时间t[1],t[2]...t[n]
输入数据保证, n≤1000, 0<t[i]≤1000000

输出

输出一个双精度浮点数, 代表炸鸡排最多可以持续的时间, 结果保留三位小数。

样例输入

```
4 2
5 1 1 2
```

样例输出

```
4.000
```

第1秒, 放置1和2
第2秒, 放置1和3
第3、4秒, 放置1和4
至此, 2, 3, 4已经全部熟了, 无法继续进行

全局题号 28701

添加于 2024-12-05

提交次数 470

尝试人数 191

通过人数 184

Other language verions

English

你的提交记录

#	结果	时间
3	Accepted	2025:10:12
2	Wrong Answer	2025:10:12
1	Wrong Answer	2025:10:12

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

#50326518提交状态

查看 提交 统计 提问

状态: Wrong Answer

源代码

```
def zhajipai(ja,k):
    ja.sort(reverse=True)
    res = sum(ja) / k
    if ja[0] <= res:
        return res
    else:
        new_ja = ja[1:]
        nk = k-1
        return zhajipai(new_ja,nk)

n,k0 = map(int,input().split())
ja0 = list(map(int,input().split()))
print(zhajipai(ja0,k0))
```

基本信息

#: 50326518
题目: 28701
提交人: 郭旭杰
内存: 3500kB
时间: 22ms
语言: Python3
提交时间: 2025-10-12 16:55:01

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问

#50326686提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def zhajipai(ja,k):
    ja.sort(reverse=True)
    res = sum(ja) / k
    if ja[0] <= res:
        return res
    else:
        new_ja = ja[1:]
        nk = k-1
        return zhajipai(new_ja,nk)

n,k0 = map(int,input().split())
ja0 = list(map(int,input().split()))
print(f"{zhajipai(ja0,k0):.3f}")
```

基本信息

#: 50326686
题目: 28701
提交人: 郭旭杰
内存: 6104kB
时间: 27ms
语言: Python3
提交时间: 2025-10-12 17:01:04

OpenJudge

题目ID, 标题, 描述

郭旭杰 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目 排名 状态 提问