

Задача 1 (5 баллов) id 5199

Условие

Настя и Наташа решили сыграть в следующую игру: для начала каждая из них берёт карточки и пишет на них случайные натуральные числа, суммарно получается N-карточек, далее эти карточки выкладываются в центр стола и начинается сама игра.

Первым делом Настя и Наташа выбирают случайное число, которое они будут пытаться достигнуть суммой карточек в центре стола, далее по очереди выбирают любую карточку по своему желанию и выкладывают в центр стола. Если сумма чисел на карточках выложенных в процессе игры в центре стола достигает задуманной, то игра завершается и побеждает та девочка, которая последней сделала ход, если сумма превышает, то побеждает другая девочка.

После нескольких партий Настя и Наташа поняли, что часто выбирают число, которое невозможно достигнуть суммой сделанных ими карточек, потому они попросили Вас написать программу, которая будет по исходному набору карточек выводить минимальное нечётное натуральное число, которое не может быть получено никаким образом из суммы карточек.

Вход:

На вход подаётся число $1 \leq N \leq 20$ - количество карточек, далее на N-строках натуральные числа на карточках (от 1 до 1000000), числа могут повторяться.

Выход:

На выходе вывести единственное число - минимальное нечётное натуральное число, которое никаким образом не может быть получено суммой из чисел на карточках.

Ввод	Вывод
5 1 3 1 8 9	7

Пример 1.

В данном примере можно получить числа

1,

$2 = 1 + 1$,

3,

$4 = 3 + 1$

$5 = 3 + 1 + 1$

6 - нельзя получить, но оно чётное, потому нас не устраивает.

Значит ответ будет 7 - так как число 7 получить нельзя никаким образом из суммы чисел на карточках. Таким образом оно является минимальным нечётным.

Проверочные тесты

Входные данные	Ожидаемый результат
5 1 3 1 8 9	7
8 26 56 2 1 79 17 18 78	5

9 30 90 8 45 1 75 2 16 32	5
9 18 1 89 55 35 74 59 100 2	5
10 36 2 43 37 22 85 15 8 81 1	5
5 87 87 1 3 43	5

7 3 48 50 25 20 1 50	5
10 91 66 2 90 69 97 92 62 60 1	5
10 2 98 97 30 17 1 59 31 80 27	5
10 40 75 42 78 95 1 85 3 21 1	7

9 20 62 94 50 87 1 24 74 3	5
5 1 2 47 74 47	5
10 89 96 1 93 35 3 4 97 12 43	9
6 1 16 3 29 2 75	7
5 1 10 3 37 67	5

5 1 44 28 92 2	5
9 10 2 19 1 48 38 67 76 37	5
10 7 45 79 66 1 3 25 76 63 13	5
5 35 92 17 1 3	5

8 60 49 3 61 1 28 70 54	5
8 13 3 1 15 82 97 18 62	5
10 3 1 52 30 1 37 59 34 93 90	7
10 3 24 1 90 62 70 21 91 25 1	7

5 2 63 86 1 51	5
5 88 99 3 1 14	5
8 34 23 3 65 57 1 77 25	5
7 2 1 36 57 37 5 53	9
6 82 34 48 82 1 3	5

10 46 1 77 15 21 2 89 23 70 37	5
9 42 2 23 90 1 90 87 54 76	5

Пример решения

```
weights = set()
```

```
def dfs(ind, mas, sm):
```

```
    weights.add(sm)
```

```
    if ind == len(mas) - 1:
```

```
        weights.add(sm)
```

```
    else:
```

```
        for i in range(ind+1, len(mas)):
```

```
            dfs(i, mas, sm + mas[i])
```

```
def run2(mas):
```

```
    dfs(-1, mas, 0)
```

```
i = 2

while i in weights:

    i += 2

print(i)


n = int(input())

a = [int(input()) for _ in range(n)]

run2(a)
```

Задача 2 (8 баллов) id 5200

Условие

Есть два города, город А и В.

Сложилось так, что большинство людей из города А работают в городе В, а люди из города В в городе А.

Чтобы добраться из города А в В и обратно - люди ездят на машине. Дорога из одного города в другой очень широкая, что позволяет вместить все машины, что едут по ней, но есть одна проблема - на одном участке пути из одного города в другой есть обрыв, а проезд по нему - это хлипкий мостик, который максимум выдерживает вес только ОДНОГО автомобиля.

Жители этих двух городов уже давно привыкли к такому принципу их жизни, потому создали общий чат в сети, в котором отмечают, когда подъехали к мосту, чтобы принять решение, кто поедет в данный момент времени по нему.

Поэтому Вам требуется определить, сколько минимально времени понадобится, чтобы добраться всем, кто работает из города А в Б и из Б в А. При условии, что если машина уже стоит у моста, то она может только пропустить другие машины, но не может стоять одна и ждать другие машины, которые подъедут из города до моста позднее. Если за время ожидания освобождения моста подъехали новые машины, то решение о проезде принимается для всех, кто в данный момент стоит у начала моста.

Город А находится на 0 км

Город Б находится на К км

Начало моста на расстоянии Х км от города А

А длина моста - Р км

Вход

На первой строке подаются натуральные числа К, Х, Р [1;1000]

На второй строке - число N - кол-во машин [1;1000]

Далее на N-строках данных о машинах (откуда едет (латинская прописная буква А или В, а также скорость в км/ч (натуральное число))

Например, А 45 (машина едет из города А со скоростью 45 км/ч).

Выход

Одно число - через сколько минимум часов (с округлением вверх до ближайшего целого (без точки!!!)) все машины смогут добраться из одного города в другой, при условии, что машины могут пропускать друг друга на мост, как с одной стороны, так и с другой, но не могут ждать, пока подъедут другие.

Ввод	Выход
300 100 20 3 А 50 В 45 В 44	8
220 20 100 2 А 20 В 100	11

Пояснение к примеру №2:

Оба автомобиля подъедут к мосту через 1 час.

Если проедет сперва машина из города В со скоростью 100 км/ч, то машина из города А начнёт движение только через час ожидания у моста, далее она проедет оставшиеся до города В 200 км за 10 часов, таким образом, все машины доберутся до нужного города за 12 часов.

Но если первой проедет машина из города А, то он доедет весь путь за $1 + 5 + 5 = 11$ часов, а машина из города В за $1 + 5$ (ожидание) + 1 (проезд по мосту) + 12 минут = 7.2 часа, таким образом минимальное время составит 11 часов.

Проверочные тесты

Входные данные	Ожидаемый результат
300 100 20 3 А 50 В 45 В 44	8
220 20 100 2 А 20 В 100	11
277 196 63 9 А 73 В 81 В 56 А 38 В 29 В 38 А 76 А 86 В 34	13

138 58 36 6 B 27 A 58 B 44 B 35 A 58 B 56	7
263 16 99 9 B 19 A 74 A 47 B 10 B 17 B 67 A 12 A 54 A 81	40
99 64 7 8 B 19 A 69 B 70 B 18 B 34 A 78 B 45 B 60	6
124 22 54 6 A 40 B 88 B 69 B 52 B 36 A 42	8

335 115 164 10 A 83 A 76 A 61 A 17 B 66 A 29 B 62 B 23 B 100 A 100	39
445 136 123 8 A 83 A 52 A 70 A 43 A 58 A 42 B 80 B 60	21
421 123 114 8 B 63 A 33 B 98 A 86 A 33 B 47 A 83 A 58	20
135 43 28 8 A 79 A 60 B 84 B 22 B 53 A 42 A 60	7

A 88	
83 70 10 9 B 48 B 32 A 98 B 13 A 63 A 10 A 38 B 42 A 69	9
489 402 13 9 B 34 B 97 B 86 A 30 A 55 A 90 B 21 B 70 B 80	24
375 204 42 8 A 39 B 42 A 90 A 57 B 91 B 42 A 89 A 65	11

137 50 45 7 B 16 A 76 A 14 B 97 B 45 B 88 B 66	13
81 29 4 7 B 20 B 13 A 30 B 68 A 27 B 42 A 34	7
342 61 208 7 B 22 B 51 A 66 B 99 B 59 B 12 B 74	45
51 30 14 5 A 18 A 50 B 61 A 42 B 68	3

494 257 147 8 A 57 B 33 A 22 A 28 A 64 A 48 B 64 A 73	32
267 122 9 6 A 17 A 61 A 18 B 78 B 11 A 90	25
257 233 11 8 A 35 B 51 A 72 B 21 B 82 A 100 A 21 B 47	13
437 222 165 7 A 99 A 27 A 80 A 35 B 41 A 95 A 48	26

263 96 147 10 A 31 B 26 B 28 B 20 A 74 B 48 B 68 B 95 A 10 A 15	57
389 341 37 7 A 55 A 97 A 71 A 84 B 88 A 52 B 11	36
124 12 96 8 B 98 B 31 B 84 A 23 A 11 A 34 A 85 B 78	24
448 129 275 10 B 38 A 61 B 48 A 17 B 88 B 89 B 77 B 79	68

B 32 A 27	
458 426 1 10 B 76 B 40 A 82 A 71 B 74 A 29 B 98 A 55 B 61 B 37	16
209 31 160 8 B 67 B 71 B 43 B 51 A 30 A 83 A 35 B 69	27
210 145 29 7 A 58 B 42 B 33 B 52 A 38 A 94 A 33	7

96 11 69	11
10	
A 52	
B 73	
B 85	
A 44	
A 91	
A 84	
B 67	
B 95	
B 75	
B 51	

Пример решения

```
import math
```

```
def srt(x):
    return x[0], -x[1]
```

```
K, X, P = map(int, input().split())
```

```
N=int(input())
```

```
c=[]
```

```
for i in range(N):
```

```
    s=input()
```

```
    s=s.split()
```

```
    c.append([s[0], int(s[1])])
```

```
A=[]
```

```
B=[]
```

```
for i in range(N):
```

```
    ct2 = P / c[i][1]
```

```
    if c[i][0]=="A":
```

```
        ct=X/c[i][1]
```

```
        ct3=(K-(X+P))/c[i][1]
```

```
        A.append([ct,ct2,ct3])
```

```
    else:
```

```
        ct=(K-(X+P))/c[i][1]
```

```
        ct3=X/c[i][1]
```

```
        B.append([ct,ct2,ct3])
```

```
A.sort(key=srt)
```

```
B.sort(key=srt)
```

```
t=0
```

```
mxt = 0
```

```
while A or B:
```

```

if A and B:
    if A[0][0] < B[0][0]:
        el = A.pop(0)
    elif A[0][0] == B[0][0] and A[0][1] > B[0][1]:
        el = A.pop(0)
    else:
        el = B.pop(0)
elif A:
    el = A.pop(0)
else:
    el = B.pop(0)
t = el[0] + el[1]
mxt = max(mxt, t + el[2])
for el in A:
    if el[0] < t:
        el[0] = t
for el in B:
    if el[0] < t:
        el[0] = t
A.sort(key=srt)
B.sort(key=srt)

print(math.ceil(mxt))

```

Задача 3 (10 баллов) id 5202

Условие

Лёша работает в лучшей IT-компании, он уважаемый программист, но у него есть небольшой минус - он очень раздражительный, а при повышении его раздражительности, его продуктивность (скорость написания кода) падает.

В очередной день Лёша захотел проследить за своей продуктивностью (скоростью написания кода) при решении задач.

Он посчитал, что он пишет K строчек кода в минуту.

Он не может полностью отключить телефон, так как у него в нём важные рабочие чаты, потому он ставит телефон на беззвучный режим с вибрацией, с указанием тех, кто может ему писать, а кто нет.

Однако он упустил один важный момент - он не указал в списке «Не беспокоить» свою подругу Полину, которая большая любительница писать сообщения по одному слову в каждом сообщении.

Если в течение 1 минуты Лёше на телефон приходит 5 и более сообщений, то он отвлекается на 1 минуту, чтобы изучить, кто ему написал (например, в 8:10 ему прислали 6 сообщений, значит в 8:11 он отвлекается и начинает изучать сообщения).

Если он видит, что все сообщения были от Полины, он раздражается и его продуктивность падает в M раз (скорость написания кода уменьшается в M раз, но не менее 1 строчки кода в минуту) и он также отвечает на сообщения, что были от Полины и что были не отвечены до этого, тратя при этом по 1 минуте на каждое сообщение.

Если не все сообщения были от Полины, то он тратит также по 1 минуте на каждое сообщение, чтобы написать ответ, включая все сообщения, на которые он до этого не отвечал.

Стоит учесть, что если в процессе написания ответа были получены новые сообщения, то Лёша на них также отвечает. Если Лёша уже отвлекся на телефон и в процессе ответа на сообщения пришло 5 и более сообщений от Полины за 1 минуту, то это не влияет на его раздражительность и продуктивность, так как он уже отвлекся от работы.

Если Лёшу ничего не раздражало в течение 60 минут, его продуктивность повышается в L раз (скорость написания кода увеличивается в L раз, НО НЕ БОЛЕЕ K).

Начальник Лёши попросил Вас написать программу, которая по истории уведомлений сможет определить, какой стала его скорость к концу рабочего дня (спустя 8 часов), а также, сколько строчек кода он написал за 8 часов.

Вход:

На первой строке подаётся число $1 \leq K \leq 30$ - количество строк кода, которые Лёша пишет за минуту.

На второй строке подаются два вещественных числа $1.1 \leq M, L \leq 2.0$.

На третьей строке подаётся число N - количество записей в истории уведомлений

Далее на N строках подаются записи об уведомлениях:

- $1 \leq X \leq 479$ минут от начала рабочего дня, когда пришло сообщение

- от кого пришло сообщение (P - Полина, A - от команды (буквы большие и латинские)).

Выход:

Скорость написания кода Лёшей к концу дня (округлённая вниз до ближайшего целого числа), а также количество строк кода написанных днём Лёшей.

Ввод	Вывод
20 2 2 12 10 P 301 A 302 P 10 P 10 P 10 P 10 P 401 A 28 A 300 A 400 A 402 A	20 8880

Проверочные тесты

Входные данные	Ожидаемый результат
-------------------	---------------------

20 2 2 12 10 P 301 A 302 P 10 P 10 P 10 P 10 P 401 A 28 A 300 A 400 A 402 A	20 8880
21 1.9 1.9 5 91 P 38 A 300 A 148 A 134 P	21 10080
20 1.3 1.5 15 68 A 51 A 300 P 31 P 103 A 180 P 233 A 171 P 271 A 294 A 233 P 191 A 127 A 67 P 253 A	20 9600

29 1.8 1.1 18 223 P 207 A 268 P 3 A 80 A 187 P 215 P 290 P 72 A 89 P 78 P 239 P 19 A 288 A 34 P 249 P 219 A 113 A	29 13920
24 1.4 1.4 13 173 P 10 A 115 P 256 A 21 P 147 P 198 A 282 A 45 A 136 A 159 A 228 P 37 P	24 11520

27 1.3 1.6 18 22 A 28 A 166 P 157 P 175 P 138 A 27 P 110 P 238 P 240 A 173 P 293 A 130 P 64 A 159 A 250 A 11 A 275 P	27 12960
10 1.1 1.6 18 22 A 230 A 183 A 234 P 138 A 126 A 136 P 166 A 251 A 124 P 195 P 113 P 151 P 18 P 25 A 112 P 273 P 209 P	10 4800

11 1.5 1.6 10 176 A 37 A 225 P 211 P 253 A 208 A 187 A 261 A 277 P 43 P	11 5280
23 1.3 2.0 19 50 P 82 P 52 P 84 A 138 P 43 A 183 A 195 P 5 P 90 P 32 P 56 A 121 P 247 A 123 P 179 A 33 P 3 P 233 P	23 11040

19 1.5 1.4 20 217 A 21 P 292 P 168 P 147 A 130 P 77 P 93 A 110 P 64 P 47 A 134 A 223 A 268 A 176 P 199 A 8 A 77 P 194 A 89 A	19 9120
22 1.3 1.7 5 264 A 203 A 223 A 53 A 51 P	22 10560
23 1.2 1.3 5 300 P 97 P 183 A 148 P 117 P	23 11040

25 1.5 1.3 18 295 A 296 A 63 A 288 P 43 A 5 P 9 P 12 A 76 A 172 A 120 A 214 P 232 P 123 A 159 P 81 P 280 P 36 P	25 12000
18 1.2 1.9 17 70 P 114 P 253 A 254 P 279 P 196 P 43 A 228 A 234 A 44 P 42 P 46 A 46 A 65 A 145 A 91 P 124 A	18 8640

24 2.0 1.8 13 293 P 292 P 172 P 15 P 225 P 201 P 36 P 210 A 295 P 190 A 250 P 204 A 142 P	24 11520
30 2.0 1.6 14 239 P 142 A 109 P 194 A 5 A 240 P 197 P 92 P 47 P 188 P 49 A 104 A 183 P 85 A	30 14400
19 1.8 1.4 7 112 A 165 P 173 A 191 P 226 A 288 A 281 P	19 9120

23 2.0 1.7 12 256 P 300 A 96 P 105 A 196 A 13 A 77 P 103 P 173 A 300 P 185 P 176 P	23 11040
25 1.2 1.4 11 136 P 15 A 180 A 54 A 121 A 189 P 77 P 152 A 57 P 22 P 209 A	25 12000

17 1.7 1.6 16 14 P 232 A 85 A 227 P 39 A 172 P 59 A 11 P 54 P 234 A 253 P 19 A 208 P 57 A 41 P 106 P	17 8160
14 1.5 1.2 12 266 A 181 A 61 P 196 P 170 A 280 P 55 P 132 A 281 P 21 P 188 A 87 P	14 6720

22 1.3 1.7 18 290 A 174 A 58 P 236 A 241 A 140 P 299 P 298 P 245 A 151 A 184 P 82 A 69 A 286 A 131 P 12 P 143 P 126 A	22 10560
22 1.4 1.5 6 90 P 154 P 8 P 3 P 234 P 95 A	22 10560
13 1.3 1.4 10 283 P 76 A 290 P 131 A 287 P 91 A 144 A 125 P 123 P 120 P	13 6240

30 1.3 1.9 11 194 P 1 P 55 P 91 P 57 A 85 A 109 A 135 A 173 P 7 P 283 A	30 14400
28 1.6 1.8 17 167 A 96 A 88 P 212 P 63 P 299 P 217 P 65 A 80 P 9 P 1 A 293 P 139 P 232 A 237 A 55 P	28 13440

166 P	
22 1.4 1.3 19 110 P 222 A 79 P 289 P 209 A 88 A 183 P 94 A 148 P 126 P 165 P 56 A 85 P 48 P 225 P 251 P 216 A 229 P 91 P	22 10560

19 1.6 1.9 6 101 A 104 A 113 A 21 P 257 P 252 A	19 9120
23 1.6 1.9 7 179 P 169 P 187 P 174 A 200 A 245 A 20 A	23 11040
25 1.8 2.0 7 148 A 76 A 275 P 281 P 261 A 184 P 193 A	25 12000

Пример решения

```

K=int(input())
k=K
M,L=map(int, input().split())
N=int(input())
a=[]
for i in range(N):
    x=input()
    x=x.split()
    a.append([int(x[0]), x[1]])
a.sort()
s=0

```

```

ans=0
i=1
t=0
while i<=480:
    t+=1
    ans+=k
    if t>=60:
        k=min(K, k*L)
        k1 = a.count([i, "A"])
        k2 = a.count([i, "P"])
        if k1==0 and k2>=5:
            i += 1
            k /= M
            k=max(1, k)
            s+=(k1+k2)
            while s:
                i+=1
                s-=1
                k1 = a.count([i, "A"])
                k2 = a.count([i, "P"])
                s+=(k1+k2)
            t=0
        elif (k2+k1)>=5:
            i += 1
            s+=(k1+k2)
            while s:
                i+=1
                s-=1
                k1 = a.count([i, "A"])
                k2 = a.count([i, "P"])
                s+=(k1+k2)
        else:
            s+=(k1+k2)

    print(i, ans)
    i += 1
print(k, int(ans))

```

Задача 4 (12 баллов) id 5204

Условие

Лариса всегда мечтала путешествовать по миру, потому она передвигалась как на поездах, так и на самолётах, кораблях и прочем транспорте. В очередном путешествии её корабль потерпел крушение и она оказалась на необитаемом архипелаге, состоящем из нескольких островов. Этот архипелаг был очень странный,

ведь все острова представляли собой будто ячейки прямоугольной матрицы размером $N \times M$.

Лариса заметила, что каждый остров обладает своей уникальной красотой, которую Лариса решила выразить в виде натурального числа - чем больше число, тем выше красота острова.

Ларисе стало интересно, какую максимальную сумму красот она может собрать, если пойдёт из левого верхнего угла матрицы, а двигаться может только вправо или вниз, но обязательно хотела учесть следующие нюансы:

- путешествие всегда Лариса начинает из левого верхнего угла
- конечная точка путешествия Ларисы может быть любой из возможных, главное - достигнуть максимально возможной суммы
- перемещаться Лариса может только вправо или вниз
- путь должен быть у Ларисы разнообразным, значит, что она не будет считать результатом достигнутую сумму, если на её пути было два и более одинаковых значения красот островов
- при оценке красоты Лариса заметила, что некоторые острова оказались очень опасными, потому значения красоты этих островов она обозначила за отрицательное число, потому, такие острова она избегает 100%
- левый верхний угол матрицы ВСЕГДА имеет положительное число

Помогите Ларисе понять, какую максимальную сумму она могла набрать при начале путешествия из левого верхнего угла матрицы.

Входные данные

На первой строке подаются два числа N - количество строк матрицы, M - количество столбцов матрицы ($1 \leq N, M \leq 100$).

Далее на N строках по M чисел, которые содержат целые числа от -1000 до 1000 (кроме 0), которые обозначают красоту или же опасность острова.

Выходные данные

На выходе вывести одно число - максимальную сумму красот островов, которую может собрать Лариса.

Ввод	Вывод
------	-------

4 5 1 2 -3 4 5 3 -4 5 6 7 6 7 8 -9 10 8 9 10 11 -1	48
--	----

Проверочные тесты

Входные данные	Ожидаемый результат
6 7 297 867 690 -60 -142 283 581 -486 -771 -821 -799 238 -371 804 -358 629 -647 94 682 -273 -352 -845 515 616 64 230 -397 -975 -9 -686 7 344 -567 -974 -704 474 -282 -551 465 -362 -978 -994	1854
6 8 86 50 -76 46 -50 56 -10 -11 -78 -54 49 90 -37 -22 74 -69 99 -44 -67 76 -57 -42 -51 -94 57 -58 36 85 -24 -64 -25 66 48 40 17 80 83 -27 -95 -39 -36 -19 11 -80 35 19 54 67	136
10 6 31 32 -58 80 -76 -66 4 85 -42 59 69 -64 75 -52 -99 91 -15 -9 7 2 43 87 -17 -21 -39 -30 -38 61 -70 16 60 -81 -29 -35 -77 -89 71 92 -68 -47 -44 -49 36 74 15 26 68 94 53 -55 65 83 -13 -10 -88 13 -53 96 99 -1	310

<p>9 5</p> <p>46 49 -76 -6 -87</p> <p>-7 -11 -13 -58 76</p> <p>39 -55 23 -94 15</p> <p>51 -71 -18 4 -91</p> <p>33 73 -93 53 12</p> <p>41 -1 85 -70 -47</p> <p>66 6 72 -53 77</p> <p>-21 -68 32 -60 69</p> <p>81 -19 -57 88 -72</p>	95
<p>9 5</p> <p>68 -85 -95 53 50</p> <p>-43 97 61 -22 -20</p> <p>41 20 67 36 24</p> <p>-53 -21 -65 87 -3</p> <p>46 84 -82 -77 -50</p> <p>-83 52 74 22 -63</p> <p>95 -84 -25 -11 -80</p> <p>26 43 -6 -41 79</p> <p>-79 38 -52 -45 56</p>	68
<p>10 7</p> <p>49 24 -42 97 -48 -7 -94</p> <p>-17 35 95 48 9 -65 67</p> <p>-39 -80 -12 8 -28 -25 -43</p> <p>71 -83 -66 77 -37 69 79</p> <p>59 98 -100 82 16 -85 -79</p> <p>-3 -61 -76 -6 -35 -23 78</p> <p>-40 -49 -1 43 94 -54 11</p> <p>66 -72 65 10 -46 -74 93</p> <p>87 -34 -71 -67 40 -36 46</p> <p>-18 -30 -63 56 -15 41 19</p>	434
<p>8 9</p> <p>13 -15 4 31 78 24 -72 -45 74</p> <p>38 -62 -63 -89 57 51 93 34 -22</p> <p>-26 -3 -67 -31 42 -73 7 -99 -74</p> <p>90 86 27 -43 70 -50 39 56 29</p> <p>-39 -23 36 -90 -52 8 -47 -2 -94</p> <p>69 -77 60 -30 19 33 -21 62 79</p> <p>66 89 85 -28 -48 -54 -58 -10 5</p> <p>-81 -25 92 17 -36 0 -20 -46 -66</p>	51

10 9 75 7 9 88 50 27 -86 -90 64 -70 83 -22 -3 35 -53 -37 15 -43 0 -63 -44 62 -91 -81 36 -11 41 -98 -49 -72 -55 72 85 -69 45 -82 -50 29 -4 74 61 -65 -92 -51 -1 -57 -83 -62 -100 -6 -48 94 -85 20 -30 -89 52 -31 -40 -7 -15 37 -5 -34 -10 -2 -96 -68 -67 -42 -33 -93 22 95 -94 49 92 -20 40 3 84 87 -54 -73 63 -32 38 6 57 47	264
7 9 35 -47 63 -40 -71 82 38 66 -91 -1 -5 -58 13 43 -88 -41 22 52 -82 70 88 -22 -7 -83 -26 15 -8 98 -18 -36 87 -14 -46 -67 -90 80 18 -100 97 93 71 3 -50 76 44 42 -2 33 27 85 34 69 40 16 -31 64 -17 8 -76 89 26 -92 -66	35
7 5 3 -33 61 -10 94 -76 85 -56 -61 -89 -77 7 -58 26 76 -35 -41 88 -57 -5 -23 -26 -99 4 83 -37 -95 64 -81 -16 45 91 14 -70 87	3
7 6 76 -64 -45 22 31 80 28 69 -97 -90 35 58 62 7 -76 -1 -77 -57 -31 55 -27 -18 -86 -21 47 8 75 -41 -39 90 46 -8 -68 -58 72 -10 27 95 96 -19 2 -30	318

5 10 13 19 10 -43 -99 -58 62 -45 58 -75 -48 27 67 21 81 24 49 -57 90 68 -94 47 -37 91 -65 53 29 16 -40 -64 -59 87 99 12 61 -22 -31 -39 -96 -77 97 54 -46 94 18 -72 93 -84 -11 69	416
7 10 69 -52 -29 -47 70 92 9 -8 20 -62 -82 30 17 6 99 -89 40 -100 -94 46 -18 77 38 62 -58 81 39 -10 31 56 -91 49 90 -75 -57 -67 -51 37 -45 95 -30 -88 58 23 16 -86 34 4 45 -1 -11 -19 89 52 -87 -13 -72 -46 65 -38 -25 -85 55 -48 54 1 -3 42 -24 -26	69
6 9 3 0 -10 -67 97 -92 -44 -26 -31 50 -74 15 91 -12 -81 -8 -56 -72 58 74 62 -68 98 -69 66 20 41 63 34 -21 -65 57 53 93 86 -98 -76 67 -95 -20 -50 -37 24 -78 -61 46 -23 42 82 -90 -36 -38 37 49	286
10 6 66 -25 37 -91 -30 -71 21 -32 38 9 -66 -48 12 -34 -90 -81 79 70 -9 96 30 -74 3 10 23 42 -8 -59 24 69 -18 -21 53 41 -95 20 -79 97 -49 88 -7 43 -22 57 49 64 -67 65 31 -69 -23 28 48 39 33 63 -53 -60 80 -82	99
5 10 95 -34 69 -75 46 60 -32 -10 -85 -12 -83 -16 11 -33 86 37 33 -51 6 74 57 -97 -8 13 64 -5 -86 -47 -40 52 29 -73 82 -56 89 -71 91 93 1 -38 7 -94 -82 42 -45 87 88 -52 -53 97	95

8 10 37 88 -72 -84 -65 -1 5 31 -35 -79 -16 53 72 12 -26 82 -97 -75 -40 -23 -43 -61 -14 -5 30 -15 -96 57 -68 -88 -41 87 -81 9 51 96 90 76 -3 -39 36 -36 7 -38 -54 84 -98 -48 62 97 -95 21 -52 -11 38 46 -18 50 71 -74 93 -6 39 -20 -42 91 35 -59 54 -8 -33 2 74 -85 -2 -12 -57 52 -37 -10	262
5 9 9 76 -41 50 17 0 30 -36 22 -62 2 89 86 47 -47 11 -24 75 94 -42 -94 -71 24 -55 -90 77 20 99 10 1 5 -69 96 56 -8 -65 -19 37 -15 -54 51 -73 82 8 73	333
6 6 18 71 -45 -69 72 10 17 0 -4 16 -59 1 -75 -30 94 96 47 -54 -14 -26 53 -41 -52 -10 4 58 -15 -97 5 20 12 -3 -18 -17 -22 32	89
10 8 59 0 -57 -75 41 -29 95 94 -72 -80 -34 -21 -18 25 17 12 -99 -84 -6 69 -10 92 -49 -64 -16 63 -100 73 62 29 -60 -28 88 30 6 76 -52 80 18 77 -97 8 -76 72 -7 50 5 26 -8 -89 -90 40 -61 -85 -55 -58 4 45 -40 -71 57 -27 58 22 27 -86 -67 3 52 83 -17 -48 90 37 91 81 -19 -15 64 71	59

8 7 36 -76 99 7 -97 -36 -15 4 -73 -24 -52 58 17 94 74 37 -34 30 97 -47 -100 55 -18 -64 22 -42 -19 -80 98 23 18 -40 -43 49 -25 21 92 24 3 -93 -99 57 89 16 62 -72 -29 60 96 47 48 84 -17 -38 93 29	556
8 5 90 51 -40 17 -79 -32 -59 82 84 -99 10 -74 -83 4 61 -64 -48 55 65 -29 19 -94 80 24 37 -58 -60 -35 59 39 -78 23 -24 54 29 -76 60 87 58 -54	141
6 5 66 -65 -9 85 -39 -22 13 15 -6 97 95 44 -28 -16 46 3 43 -100 82 -63 90 70 -55 7 86 21 -53 71 -99 48	66
9 7 59 57 -34 19 -11 26 7 -35 -62 48 -50 -52 -59 31 52 -49 66 -6 -42 28 89 75 68 -7 -23 30 -85 81 74 14 45 4 -41 -96 53 91 43 29 94 1 -95 96 -10 8 -33 -37 -73 -68 20 6 -5 -32 16 78 21 -77 2 12 -19 -92 -54 -97 3	116

5 8 87 30 35 38 -69 71 -98 -58 92 16 64 -45 62 -97 -49 49 21 -95 -63 -92 12 91 1 -19 72 3 -30 -86 90 -43 53 -3 -90 59 -70 83 0 13 80 31	334
9 6 93 56 11 6 -49 -40 27 -10 -52 -23 -92 -100 2 -96 -36 20 -58 34 95 67 -11 10 96 16 3 88 -20 51 -83 62 -84 41 22 66 -74 4 -75 -62 72 57 -14 -67 8 54 89 -94 69 -98 18 36 86 -5 44 40	682
6 10 43 28 -82 -69 -65 -42 8 -28 5 21 65 49 -86 -62 -4 -1 88 48 -78 -100 6 -92 -45 -27 53 -18 39 -25 -77 30 23 -67 -53 -30 25 -8 -36 75 -71 -68 -32 89 42 62 -23 24 -14 67 10 -96 -31 -19 18 26 50 87 82 -61 -84 81	157
8 6 54 -94 84 16 -88 -62 94 -6 56 -92 -98 -46 64 -13 47 15 70 -84 28 -89 98 49 -83 -77 -71 76 -76 87 -74 82 33 -41 69 -40 -20 30 12 39 -52 -95 -86 65 -32 40 21 62 92 -54	240
8 6 55 75 46 -13 -10 68 93 26 36 -82 23 39 43 48 74 -25 59 24 -62 -84 -20 98 35 52 -55 -38 -59 -92 65 -99 8 -75 71 90 -37 -17 -2 66 29 -19 88 14	313

31 25 7 -39 -46 -48	
5 8 46 -75 32 77 82 -93 -76 -3 -77 -63 -79 86 92 -92 -78 38 80 -73 58 31 39 -95 98 15 95 -51 -17 -64 -18 68 22 -24 76 -28 -35 75 69 19 52 42	46

Пример решения

```
def f(x, y, visit):
    if a[y][x] < 0:
        return 0
    elif x+1 < m or y+1 < n:
        visit += a[y][x]
        if x + 1 < m:
            k = f(x + 1, y, visit)
            if k:
                count.append(visit + k)
            else:
                count.append(visit)
        if y + 1 < n:
            k = f(x, y + 1, visit)
            if k:
                count.append(visit + k)
            else:
                count.append(visit)
    else:
        visit += a[y][x]
        return 0
```

```
a = []
n, m = list(map(int, input().split()))
for i in range(n):
    a.append(list(map(int, input().split())))
```

```
count = []  
  
f(0,0,0)  
  
print(max(count))
```

Задача 5 (15 баллов) id 5206

Условие

Мэрия города Алькс решила построить новый район, с новым домами и дорожками, но в этот раз, в честь юбилея города, мэрия приняла решения построить дома так, чтобы дорожки между ними образовывали сердце, чтобы граждане понимали, что мэрия их любит.

Данная задумка была донесена до всех работающих в городе архитекторов и инженеров, а через месяц каждая компания подала в мэрию свой проект района, который они видят. Так как некоторые компании оказались очень непрофессиональными, то в их проектах не было сердец, потому мэрия попросила Вас написать программу, которая принимает описание проекта в виде графа, без указания координат домов и дорог, а возвращает число - сколько всего сердец могло быть в этом проекте.

Сердцем будем считать кольцевой маршрут, который имеет не менее 4 вершин, которые соединены между собой.

Обязательными условиями считается то, что

- сердца не могут соприкоснуться между собой ни одной вершиной, таким образом, если найдено два возможных сердца и они соприкасаются между собой, то считаем, что ни одного сердца ни найдено
- дома в сердце должны быть симметричны относительно пиков сердца

Помогите мэрии отсеить часть проектов, чтобы дальнейшее изучение заняло меньше времени.

Входные данные

На первой строке подаются два числа N - количество домов в городе, M - количество дорог между домами ($1 \leq N, M \leq 100$).

Далее на M строках заданы два номера домов, которые соединяются между собой (нумерация домов начинается с 1 и не превышает 100). Дороги между домами двусторонние, потому, если задано, что существует путь от дома №1 и дома №2, значит существует путь от №2 до №1.

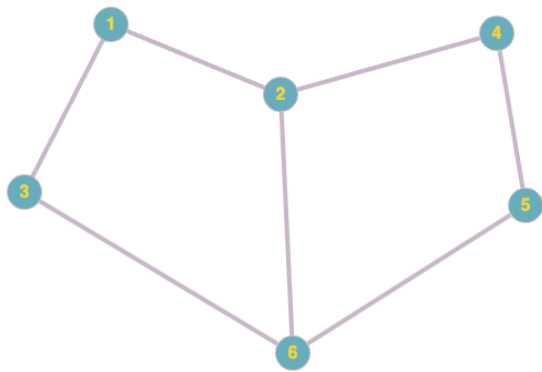
Выходные данные

На выходе вывести одно число - количество найденных возможных сердец в проекте.

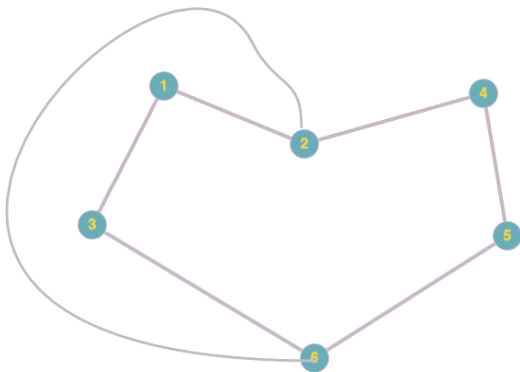
Ввод	Вывод
7 8 1 2 2 3 4 3 5 4 6 7 5 6 6 7 7 2	1
11 13 1 2 2 3 4 3 4 7 4 8 5 6 6 7 7 2 8 6 5 9 9 10 10 11 11 5	2

Стоит также учесть, что если через сердце может проходить дорога, то в проекте её можно просто провести не через центр сердца, таким образом сердце будет существовать.

Можно построить район так



А можно вот так



Таким образом сердце будет существовать.

Проверочные тесты

Входные данные	Ожидаемый результат
7 8 1 2 2 3 4 3 5 4 6 7 5 6 6 7 7 2	1

11 13	2
1 2	
2 3	
4 3	
4 7	
4 8	
5 6	
6 7	
7 2	
8 6	
5 9	
9 10	
10 11	
11 5	

10 42	1
9 9	
3 8	
2 5	
3 3	
7 8	
5 10	
1 3	
3 5	
4 6	
4 5	
4 10	
4 7	
2 7	
3 7	
1 9	
6 8	
1 6	
9 10	
2 6	
8 9	
1 5	
1 7	
1 2	
5 5	
6 7	
7 9	
3 4	
4 4	
10 10	
6 9	
2 10	
6 10	
2 4	
7 10	
8 8	
5 7	
1 4	
2 9	
6 6	
3 6	
1 8	
5 9	

10 11 2 7 1 2 1 4 2 9 5 10 1 9 7 8 3 7 3 6 6 8 5 9	1
9 11 3 8 5 8 1 5 2 3 1 7 2 7 5 7 2 9 1 6 6 8 5 9	1
7 12 4 4 1 3 3 5 2 2 4 6 4 5 1 5 2 3 4 7 2 7 1 4 3 4	1

10 41	1
3 9	
3 8	
4 8	
4 9	
2 5	
1 3	
3 5	
4 6	
4 10	
3 10	
4 5	
4 7	
2 7	
1 9	
6 8	
1 6	
8 10	
9 10	
8 9	
2 6	
5 8	
1 5	
1 7	
1 2	
5 6	
6 7	
7 9	
3 4	
6 9	
10 10	
2 10	
2 4	
2 3	
7 10	
1 4	
2 9	
5 7	
2 8	
3 6	
1 8	
5 9	

4 6 2 2 2 3 1 2 1 1 1 4 3 4	1
8 21 4 8 1 1 7 8 1 3 2 2 4 6 4 7 2 7 6 8 2 6 1 5 5 6 6 7 3 4 4 4 2 4 8 8 1 4 2 8 6 6 1 8	1
7 10 1 3 2 4 3 5 4 5 1 5 2 5 5 6 2 7 5 7 1 4	1

14 16 1 2 5 6 6 12 5 14 4 7 1 13 2 3 6 8 5 9 3 4 5 11 2 7 6 7 9 10 10 11 4 8	2
12 15 1 2 5 6 4 7 5 10 2 3 6 8 2 8 5 9 3 4 5 11 2 7 6 7 9 10 10 11 4 8	2

12 16 1 2 5 6 3 8 4 7 2 3 6 8 5 9 3 4 5 11 2 7 7 12 6 7 2 6 9 10 10 11 4 8	2
14 15 1 2 5 6 4 7 3 7 2 3 6 8 5 9 3 4 9 11 5 11 2 7 6 7 9 10 10 11 4 8	2

15 15 1 2 5 6 4 7 4 15 12 14 2 3 6 8 5 9 3 4 5 11 2 7 6 7 9 10 10 11 4 8	2
14 15 1 2 5 6 4 7 2 3 6 8 5 9 3 4 5 11 2 7 8 13 6 7 9 10 10 11 4 8 12 13	2

14 17 1 2 5 6 4 7 10 10 11 12 2 3 6 8 5 9 3 4 4 11 5 11 2 7 6 7 9 10 2 14 10 11 4 8	2
12 18 1 2 5 6 3 6 2 9 4 7 2 3 6 8 2 8 5 9 3 4 5 11 2 7 6 7 9 10 1 1 10 11 4 8 7 7	2

15 15 1 2 5 6 4 7 3 7 2 3 6 8 5 9 3 4 5 11 2 7 3 3 6 7 9 10 10 11 4 8	2
13 15 1 2 5 6 4 7 2 3 6 8 5 9 3 4 7 13 5 11 2 7 6 7 9 10 11 11 10 11 4 8	2

Пример решения

```
def find_cycles(graph):
    def dfs(node, start, path):
        nonlocal visited, cycles

        visited[node] = True
        path.append(node)

        for neighbor in graph[node]:
            if neighbor == start and len(path) > 2:
```

```

        cycles.add(tuple(sorted(path)))
    elif not visited[neighbor]:
        dfs(neighbor, start, path)

    path.pop()
    visited[node] = False

cycles = set()
visited = [False] * (len(graph) + 1)

for i in range(1, len(graph) + 1):
    dfs(i, i, [])

return cycles

def main():
    N, M = map(int, input().split())

    graph = {i: [] for i in range(1, N + 1)}

    for _ in range(M):
        a, b = map(int, input().split())
        graph[a].append(b)
        graph[b].append(a)

    cycles = find_cycles(graph)

    even_cycles = [cycle for cycle in cycles if len(cycle) % 2 == 0 and len(cycle) > 2]

    unique_cycles = []
    for cycle1 in even_cycles:
        is_contained = False
        for cycle2 in even_cycles:
            if cycle1 != cycle2 and set(cycle1).issubset(set(cycle2)):
                is_contained = True
                break
        if not is_contained:
            unique_cycles.append(cycle1)

    final_cycles = []
    for cycle in unique_cycles:
        share_vertices = False
        for other_cycle in unique_cycles:
            if cycle != other_cycle and set(cycle).intersection(set(other_cycle)):
                share_vertices = True
                break
        if not share_vertices:

```

```
final_cycles.append(cycle)
```

```
print(len(final_cycles))
```

```
if __name__ == "__main__":  
    main()
```