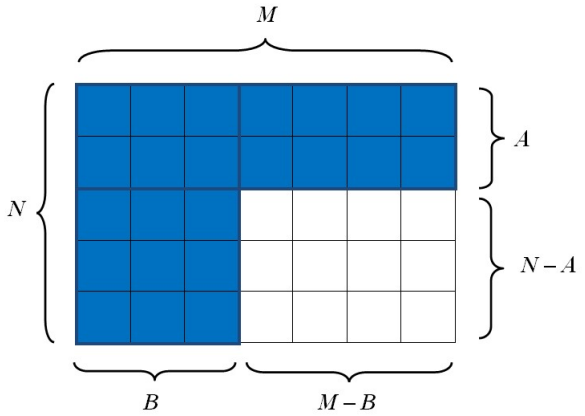
Задача 1. Полосатая раскраска

Подгруппа $B = 0$

В этой подгруппе Маша красит только строчки, а значит, останется ровно $N - A$ полностью белых строк по M клеток в каждой. Итоговая формула: $(N - A) \cdot M$.

Полное решение

Заметим, что не важно, в каком порядке красить строки и столбцы и какие именно. Допустим, покрашены верхние A строк и левые B столбцов:



Тогда незакрашенным остался прямоугольник размера $(N - A) \times (M - B)$, он содержит $(N - A) \cdot (M - B)$ клеток. Получим решение:

```
N = int(input())
M = int(input())
A = int(input())
B = int(input())
print((N - A) * (M - B))
```

Задача 2. Перчатки

Для начала заметим, что если $B \neq 0$ или $C \neq 0$, то все A и B детей смогут встать в ряд. Таким образом, рассмотрим два случая:

1) $B = 0$ и $C = 0$

Тогда ученики только в белых и только в чёрных перчатках не смогут встать рядом в цепочку и ответ будет равен $max(A, D)$.

2) $B \neq 0$ или $C \neq 0$

Заметим, что ученики в бело-чёрных перчатках могут стоять рядом только с учениками в чёрно-белых перчатках и наоборот, таким образом, из B и C учеников в ряд могут встать только $min(B, C) + [B \neq C]$, а также все дети только в чёрных и только в белых перчатках, т.е. всего $A + D + min(B, C) + [B \neq C]$.

```
a = int(input())
b = int(input())
c = int(input())
d = int(input())

if b == 0 and c == 0:
    print(max(a, d))
else:
    print(a + d + 2 * min(b, c) + (b != c))
```

Задача 3. Странная планета

Рассмотрим решение, правильно работающее при $M_1 = M_2, Y_1 = Y_2$. В таком случае весь эксперимент проходит в течение одного месяца одного года и можно просто вывести разность между днём

окончания эксперимента и днём начала, увеличенную на один (так как день окончания включается в эксперимент). Получаем следующий вариант решения:

```
d1 = int(input())
input()
input()
d2 = int(input())
print(d2 - d1 + 1)
```

Для построения решения, работающего при $Y_1 = Y_2$, можно воспользоваться следующей идеей. Для дней начала и окончания эксперимента определим их номера в году и обозначим как NUM_1 и NUM_2. Для этого к исходному номеру дня прибавим длительности всех месяцев, с номерами меньшими номера месяца данного дня. После этого ответом будет разность данных номеров. Получится такое решение:

```
d1 = int(input())
m1 = int(input())
y1 = int(input())
d2 = int(input())
m2 = int(input())
y2 = int(input())
n = int(input())
num1 = d1
num2 = d2
for index_m in range(1, n + 1):
    len_m = int(input())
    if index_m < m1:
        num1 += len_m
    if index_m < m2:
        num2 += len_m
print(num2 - num1 + 1)
```

Построим общее решение. Кажется, что длительность эксперимента для случая $Y_1 < Y_2$ будет состоять из трёх слагаемых — количество дней от даты начала эксперимента до конца года, длительность лет полностью охваченных экспериментом, количество дней от начала года до даты окончания эксперимента. Однако, можно получить формулу проще. Как и в прошлом решении для дней начала и окончания эксперимента определим их номера в году. Ответом будет разность данных номеров плюс разность между Y_2 и Y_1, умноженная на количество дней в году, и плюс один.

Поймём, почему это так. Легко видеть, что если $D_1 = D_2, M_1 = M_2$ (то есть $NUM_1 = NUM_2$), то ответ будет равен разности между Y_2 и Y_1, умноженной на количество дней в году плюс один. Заметим, что если $NUM_1 < NUM_2$, то к полученному ответу достаточно прибавить разность NUM_2 и NUM_1, а если $NUM_1 > NUM_2$, то вычесть разность NUM_1 и NUM_2. Но это то же самое, что просто прибавить разность NUM_2 и NUM_1. Поэтому итоговое решение примет следующий вид:

```
d1 = int(input())
m1 = int(input())
y1 = int(input())
d2 = int(input())
m2 = int(input())
y2 = int(input())
n = int(input())
num1 = d1
num2 = d2
len_y = 0
for index_m in range(1, n + 1):
    len_m = int(input())
```

```
    if index_m < m1:
        num1 += len_m
    if index_m < m2:
        num2 += len_m
    len_y += len_m
print(num2 - num1 + len_y * (y2 - y1) + 1)
```

Задача 4. Бирмингем

Рассмотрим конкретный номер из телефонного справочника. Заметим, что нас интересуют только те цифры, которые не совпадают с цифрами предполагаемого номера Артура. Их и нужно пытаться переставить. Если этих цифр больше K, то очевидно, что номер не подходит.

Далее надо было понять, можно ли переставить несовпадающие цифры в номере Артура, чтобы получить рассматриваемый номер из телефонной книги. В зависимости от подгруппы это можно было делать разными способами.

Для решения первой подгруппы можно написать полный перебор по множеству несовпадающих цифр.

Вторая подгруппа является более лёгкой версией полного решения, достаточно было использовать две переменные для каждой из строк и подсчитать количество 1 и 2 в каждом номере на несовпадающих позициях. Если количество 1 и 2 совпало, то этот номер подходит.

Для решения на полный балл нужно было завести два массива размера 10, где в i-й ячейке будет лежать количество вхождений цифры i на несовпадающих позициях, и сравнить эти массивы.

```
n = int(input())
l = int(input())
k = int(input())

s1 = input()
ans = 0

for i in range(n):
    s2 = input()
    cnt1 = [0] * 10
    cnt2 = [0] * 10
    cnt_ne = 0
    for j in range(l):
        if (s1[j] != s2[j]):
            cnt1[int(s1[j])] += 1
            cnt2[int(s2[j])] += 1
            cnt_ne += 1
    if (cnt_ne <= k and cnt1 == cnt2):
        ans += 1

print(ans)
```

Задача 5. Плитки

Формализуем условие: требуется найти минимальное количество упаковок Y такое $X \cdot Y$ является полным квадратом, то есть, что $X \cdot Y = A^2$ для некоторого целого числа A (стороны квадрата).

Заметим, что $Y = X$ всегда даст нам полный квадрат, поэтому ответ — это некоторое число от 1 до X. Переберём все такие числа, а для проверки, является ли $X \cdot Y$ полным квадратом воспользуемся, например, функцией `sqrt` или возведением в степень 0.5. Получим решение, которое работает за $\mathcal{O}(X)$ и проходит все тесты, для которых $N \leqslant 1000$.

```
from math import sqrt
```

```
def full_square(Z):
    sqrt_Z = int(round(sqrt(Z)))
    return sqrt_Z * sqrt_Z == Z

X = int(input())
for Y in range(1, X + 1):
    if full_square(X * Y):
        print(Y)
        break
```

Для решения на полный балл заметим, что число является полным квадратом тогда и только тогда, когда все простые множители входят в его разложение в чётной степени. Поэтому все простые множители, которые входят в X нечётное число раз, хотя бы по одному разу должны быть взяты в Y. С другой стороны, взятия каждого из них ровно по одному разу достаточно, чтобы $X \cdot Y$ стало полным квадратом, поэтому данный вариант будет минимально возможным из всех. Для разложения числа X на простые множители используем классический алгоритм за $\mathcal{O}(\sqrt{X})$.

```
X = int(input())
Y = 1
d = 2
while d * d <= X:
    d_power = 0
    while X % d == 0:
        X //= d
        d_power += 1
    if d_power % 2 == 1:
        Y *= d
    d += 1
Y *= X
print(Y)
```

Бонус: Улучшите описанный подход до $\mathcal{O}(\sqrt[3]{X})$.