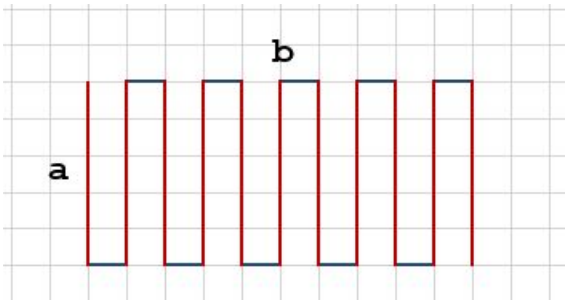
Разбор задач

Задача 1. Найди все палиндромы

Всего имеется 9 различных подстрок-палиндромов:
ABA
ABABA
ABBCBBA
BAB
BABAB
BABABAB
BB
BBCBB
BCB

Задача 2. Ночная смена

Полный путь охранника состоит из $(b + 1)$ вертикальных линий длины a и b горизонтальных линий длины 1.



Всего получается

$$(b + 1) * a + b * 1$$

что можно записать разными способами, например, если раскрыть скобки, то получится

$$a * b + a + b$$

Задача 3. Запертая комната

Ответ:

311
13
3312
133111
33312

Решение.

1. Найдем все числа, которые можно получить путем нажатия одной из кнопок (поскольку на индикаторе нечетное число, имеет смысл нажимать только первую и третью кнопку). Это числа 2 (1) и 5 (3). В скобках будем указывать последовательность нажатий кнопок для получения этого числа.

2. Найдем все числа, которые можно получить путем нажатия двух кнопок. Очевидно, что все такие числа возможно получить только из чисел, которые можно получить путем нажатия на одну кнопку (мы их нашли на предыдущем шаге) добавив в конец их кода 1, 2 или 3.

Из числа 2 можно получить числа 3 (11), 10 (13) и 1 (12). Последнее число уберем из списка — его мы можем получить вообще не нажимая ни одной кнопки (в общем случае будем добавлять в наш список только новые числа, поскольку если число уже встретилось нам ранее, очевидно, что оно имело более короткий код).

Из числа 5 получим числа 6 (31) и 25 (33). Выпишем все полученные на втором шаге числа в порядке возрастания (для облегчения дальнейшей работы): 3 (11), 6 (31), 10 (13) и 25 (33). Отметим, что число 10 присутствует в задании, ответ для этого числа — его код (13).

3. На следующих шагах будем поступать аналогично: из всех чисел, полученных на предыдущих шагах постараемся получить все возможные новые числа, сохраняя их коды. Вот числа, которые будут получены на третьем шаге (в порядке возрастания): 4 (111), 7 (311), 11 (131), 15 (113), 26 (331), 30 (313), 50 (133) и 125 (333). Отметим, что число 7 присутствует в задании, ответ для этого числа — его код (311).

4. Вот числа, которые будут получены на четвертом шаге (в порядке возрастания): 8 (3111), 12 (1311), 13 (3312), 16 (1131), 20 (1113), 27 (3311), 31 (3131), 35 (3113), 51(1331), 55(1313), 75(1133), 126(3331), 130(3313), 150(3133), 250(1333) и 625(3333). Отметим, что число 13 присутствует в задании, ответ для этого числа — его код (3312).

5. Этот алгоритм можно продолжать и дальше (в информатике подобный способ нахождения кратчайшего пути между двумя объектами носит название «волновой алгоритм» или «алгоритм Ли»), но чисел стало много, работать становится сложнее. К счастью, нам осталось найти коды всего для двух чисел. Попробуем зайти с обратной стороны.

   Число 63 из задания возможно получить только из чисел 62 (с помощью первой кнопки) и 126 (с помощью второй кнопки). Число 126 нам встретилось на 4 шаге, значит ответ для числа 63 равен коду для числа 126 с добавленной цифрой 2, то есть (33312).

   Число 53 из задания возможно получить только из чисел 52 (с помощью первой кнопки) и 106 (с помощью второй кнопки). Ни одно из этих чисел нам пока не встретилось, но можно заметить, что число 52 возможно получить из числа 51 (встретившимся нам на четвертом шаге). Тогда ответ для последнего числа из задания 53 будет (133111).

Также данную задачу возможно решить путем составления компьютерной программы и реализации алгоритма динамического программирования.

Задача 4. Земляничная поляна

Пример из 11 команд:

E>A
E>C
E>D
C>B
A>C
D>F
F>A
A>D
B>D
D>A
A>B

Пример решения задачи представлен в таблице:

шаг	20	3	5	6	7	30
1.	17	3	0	0	0	0
2.	11	3	0	6	0	0
3.	4	3	0	6	7	0
4.	выбыл	3	5	1	7	0
5.	выбыл	0	5	4	7	0
6.	выбыл	0	5	выбыл	0	7
7.	выбыл	3	5	выбыл	0	4
8.	выбыл	0	5	выбыл	3	выбыл
9.	выбыл	0	1	выбыл	7	выбыл
10.	выбыл	3	1	выбыл	4	выбыл
11.	выбыл	0	4	выбыл	выбыл	выбыл

Задача 5. Костяные войны

Частичное решение:

Переберем все возможные a и b, не превышающие P, и проверим выполнение равенства $2(a+b)=P$.

```
p = int(input())
count = 0
for a in range(1, p + 1):
    for b in range(1, p + 1):
        if 2 * (a + b) == p:
            count += 1
print(count)
```

Полное решение:

Очевидно, что если периметр нечётен, то способов нет ни одного, и нужно вывести ноль.

В чётном случае заметим, что один из них, пускай Марш, может пожертвовать отрезки длиной от 1 до $P/2-1$, а отрезки Копа определятся однозначно. Всего $P/2-1$ способ.

```
n = int(input())
if n % 2 == 1:
    print(0)
else:
    print(n // 2 - 1)
```

Задача 6. Потерявшееся число

Частичное решение:

Будем перебирать все числа, делящиеся на 6, получать их половину и треть и проверять, равна ли какая-нибудь пара чисел из этой тройки паре чисел, данной в условии. Если да — ответом будет третье число из тройки, если нет — переходим к следующему числу, делящемуся на 6, и проверяем его.

Полное решение:

Первый случай: остались карточки, на которых изначально были записаны исходное число и его половина, потерялась карточка с третью числа.

Признаком этого случая является следующее равенство: 2 * a = b. Значит, третье число можно вычислить по формуле b // 3.

Второй случай: остались карточки, на которых изначально были записаны исходное число и его треть, потерялась карточка с половиной числа.

Признаком этого случая является следующее равенство: 3 * a = b. Значит, третье число можно вычислить по формуле b // 2.

Третий случай: остались карточки, на которых изначально были записаны половина исходного число и его треть, потерялась карточка с самим числом.

Признаком этого случая является следующее равенство: 3 * a = 2 * b. Значит, третье число можно вычислить по формуле 3 * a.

```
a = int(input())
b = int(input())
if a * 2 == b:
    print(b // 3)
elif a * 3 == b:
    print(b // 2)
else:
    print(2 * b)
```

Задача 7. Два грузчика

Частные решения:

Первая подзадача (любую коробку может унести любой грузчик):

В этом случае каждой ходкой переносится по 2 коробки, поэтому...

Первый случай: если n — четное: $ans = n/2$.

Второй случай: если n — нечетное: $ans = n/2 + 1$ (последнюю коробку несет кто-то один).

В обеих случаях ответом будет $n/2$, округленное вверх до целого числа или $ans = \lfloor (n+1)/2 \rfloor$, где операция $\lfloor . \rfloor$ — целая часть от результата деления.

```
b = int(input())
n = int(input())
ans = (n + 1) // 2
print(ans)
```

Вторая подзадача (нет коробок, которые нужно тащить вдвоем):

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик (считывая вес очередной коробки будем увеличивать на 1 соответствующую переменную).

Первый случай: если $x \geqslant y$. В этом случае Федя перетаскает все свои коробки не раньше Шурика и общее количество ходок получится таким же, как и в первой подзадаче: ответом будет $n/2$ округленное вверх до целого числа или $ans = \lfloor (n+1)/2 \rfloor$.

Второй случай: если $y > x$. В этом случае Федя перетаскает все свои коробки позже Шурика и несколько последних ходок Шурик будет просто идти рядом с Федей с пустыми руками: $ans = y$.

```
b = int(input())
n = int(input())
L = list()
for i in range(n):
    L.append(int(input()))

x, y = 0, 0
for i in L:
    if i <= a:
```

```
        x += 1
    else:
        y += 1

ans = 0
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```

Полное решение:

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик, z — количество коробок, которые они могут перенести только вдвоем.

Тогда ответ точно равен z (столько коробок они несут вдвоём) плюс...

Первый случай: если $x \leqslant y$, то к ответу добавим y (переходов Феди с грузом больше, чем у Шурика).

Второй случай: иначе добавим $(x+y)/2$ с округлением вверх (Шурик не ходят без груза, кроме, возможно, последнего перехода, если осталась одна коробка на двоих).

```
a = int(input())
b = int(input())
n = int(input())
x, y, z = 0, 0, 0
for i in range(n):
    p = int(input())
    if p <= a:
        x += 1
    elif p <= b:
        y += 1
    else:
        z += 1
ans = z
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```