Задача 1. Пятистенок

Для строительства одного ряда нужно три бревна длиной a и два бревна длиной b, поэтому для нахождения ответа нужно нацело поделить n на $3a + 2b$.

```
a = int(input())
b = int(input())
c = int(input())
ans = c // (3 * a + 2 * b)
print(ans)
```

Задача 2. Ёлочки

Ёлочка красоты n состоит из $2n$ ветвей длины 1, 2, ..., n и ствола длины $2n + 1$, поэтому нужно посчитать сумму $s = 1 + 2 + ... + n$ и вывести значение $2s + 2n + 1$.

Если для нахождения суммы чисел от 1 до n использовать цикл, то решение будет набирать 60 баллов. т.к. для больших n будет работать долго. Пример такого решения

```
n = int(input())
s = 0
for i in range(1, n + 1):
    s += i
print(2 * s + 2 * n + 1)
```

Для того, чтобы набрать 100 баллов, заметим, что $2s = n(n + 1)$, поэтому программа может просто вывести значение $n(n + 1) + 2n + 1$.

```
n = int(input())
print(n * (n + 1) + 2 * n + 1)
```

Задача 3. Соревнование делимости

Для того, чтобы набрать 60 баллов, можно просто перебрать все числа от x до y, посчитать число делящихся на k среди них, посчитать число делящихся на m, и вычесть из одного числа другое. В решении ниже эти количества не считаются отдельно, а считается сразу же ответ. Если число делится на k, то ответ увеличивается на 1, если число делится на m, то ответ уменьшается на 1, одно число может и увеличить, и сразу же уменьшить ответ на 1.

```
k = int(input())
m = int(input())
x = int(input())
y = int(input())

ans = 0
for i in range(x, y + 1):
    if i % k == 0:
        ans += 1
    if i % m == 0:
        ans -= 1
print(ans)
```

Чтобы набрать 100 баллов, нужно быстро посчитать количество чисел, делящихся на данное число на данном отрезке. Количество чисел от 1 до y, делящихся на k, равно $\lfloor \frac{y}{k} \rfloor$ (целая часть от частного $\frac{y}{k}$, то есть целочисленное деление y на k). А чтобы посчитать количество кратных k на отрезке от x до y нужно из количества кратных k на отрезке от 1 до y вычесть количество кратных

k на отрезке от 1 до $x-1$, то есть $\lfloor \frac{y}{k} \rfloor - \lfloor \frac{x-1}{k} \rfloor$. Аналогично посчитаем количество кратных m на отрезке от x до y. Пример решения.

```
k = int(input())
m = int(input())
x = int(input())
y = int(input())
print(y // k - (x - 1) // k - y // m + (x - 1) // m)
```

Задача 4. Задача из ЕГЭ

40 баллов можно набрать, если реализовать то, что требуется в условии задачи, "в лоб": вычислить $2^a + 2^b - 2^c$, перевести результат в двоичную систему счисления и посчитать число нулей и единиц.

Пример решения на языке Python.

```
a, b, c, d = [int(input()) for i in range(4)]
n = (1 << a) + (1 << b) - (1 << c)
print(bin(n)[2:].count(str(d)))
```

В этом решении для вычисления степени 2^a используются битовый сдвиг числа 1 на a бит влево: `1 << a`, но можно использовать и возведение в степень: `2 ** a`. Для перевода числа в двоичную систему счисления здесь используется функция `bin` языка Python, а для подсчёта числа символов «0» или «1» используется метод `count`. Если в языке программирования нет встроенной функции перевода числа в двоичную систему счисления, можно сделать это циклом, деля число на 2, например, так.

```
count = [0, 0]
while n > 0:
    count[n % 2] += 1
    n //= 2
```

Чтобы набрать полный балл необходимо разобрать различные случаи соотношения чисел a, b, c. Например, если $a > b > c$, то значение $2^a + 2^b - 2^c$ будет иметь в двоичной системе счисления вид 100...0011...1100...00, где старшая единица соответствует позиции a, а группа из единиц расположена между позициями b (не включительно) и c (включительно). Необходимо аккуратно разобрать все случаи и для каждого из них посчитать количество нулей и единиц. Пример решения на полный балл.

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

if a < b:
    a, b = b, a

if c <= b:
    if d == 0:
        print(c + a - b)
    else:
        print(b - c + 1)
elif a > c > b:
    if d == 0:
        print(c - 1)
    else:
        print(a - c + 1)
else: # c == a
```

```
    if d == 0:
        print(b)
    else:
        print(1)
```

Задача 5. Бизнесмен Василий

В этой задаче нужно посчитать количество способов разбить массив на три непрерывных непустых части с равной суммой. Для того, чтобы набрать 40 баллов достаточно перебрать все возможные разбиения, то есть перебрать значения границ i между первой и второй частью и границы j между второй и третьей частью. Посчитаем для каждой части сумму элементов в ней и если три эти суммы равны увеличим ответ на 1. Такое решение имеет сложность $O(n^3)$.

```
n = int(input())
a = [int(input()) for i in range(n)]
ans = 0
for i in range(1, n - 1):
    for j in range(i + 1, n):
        if sum(a[:i]) == sum(a[i:j]) == sum(a[j:]):
            ans += 1
print(ans)
```

Можно ускорить это решение, если сумму элементов в каждой части вычислять за $O(1)$, а не за $O(n)$. Это можно сделать при помощи префиксных сумм, или следующим образом: передвигая границу одной части на один элемент вправо, будем добавлять значение этого элемента к сумме той части, к которой добавился этот элемент. Если сумма всех элементов в массиве равна s, то значение s должно делиться на 3, и тогда достаточно проверить, что суммы первой и второй части равны $s/3$. Общая сложность решения будет $O(n^2)$. Такое решение набирает 60 баллов.

```
n = int(input())
a = [int(input()) for i in range(n)]
s = sum(a)
ans = 0
if s % 3 == 0:
    s = s // 3
    s1 = 0
    for i in range(n - 2):
        s1 += a[i]
        if s1 == s:
            s2 = 0
            for j in range(i + 1, n - 1):
                s2 += a[j]
                if s2 == s:
                    ans += 1
print(ans)
```

Уменьшим сложность решения до $O(n)$. Для этого будем перебирать значение j — границы между второй и третьей части. Если сумма элементов левее этой границы равна $2s/3$, то это подходящая граница. Также нам нужно знать количество подходящих границ между первой и второй частью левее данной границы — это количество таких границ, левее которых сумма элементов будет равна $s/3$. Количество таких подходящих границ будет хранить в переменной `count`, которую будем увеличивать, если знание текущей суммы (слева от рассматриваемой границы) равно $s/3$. А если значение текущей суммы равно $2s/3$, то будем к ответу добавлять значение `count`.

```
n = int(input())
a = [int(input()) for i in range(n)]
s = sum(a)
```

```
ans = 0
if s % 3 != 0:
    s //= 3
    count = 0
    curr_sum = 0
    for elem in a[:-1]:
        curr_sum += elem
        if curr_sum == 2 * s:
            ans += count
        if curr_sum == s:
            count += 1
print(ans)
```