Школьный тур олимпиады по информатике.
9-11 классы.

Разбор задач

Задача A. Комета Дон Кихота (70 баллов)

В ограничениях задачи $A, B, C \leq 20000$ достаточно перебрать все годы от A до B и проверить каждый на делимость C.

Задача A. Комета Дон Кихота (100 баллов)

В ограничениях задачи $1 \le A, B, C \le 2^{31} - 1$

Достаточно посчитать сколько годы, кратные C встречались на отрезке от A до B.

Для этого найдем количество лет кратных C от «начала времен», их будет B, деленное нацело на C, и вычтем те годы, что были до A, т.е. $A - 1$, деленное нацело на C.

Итого: $\text{ans} = \frac{\text{B}}{\text{C}} - \frac{A-1}{C}$. Деление целочисленное, в Python используйте //.

Задача B. Дон Кихот и Санчо Панса (30 баллов)

Для решения на 30 баллов ($1 \leq \mathrm{n} \leq 3, 1 \leq k \leq 3, 1 \leq a_i \leq n-1$) можно все случаи обработать вручную.

Задача B. Дон Кихот и Санчо Панса (60 баллов)

Смоделируем процесс каждодневного выбора:

Пусть $spear[j]$это копье на позиции j в данный момент, позиции занумеруем с 0 до $n-1$. Инициализация $spear[j] = j+1,\ j = 0 \ldots n-1$. Выбор копья это:

1) Отсчет a_i копий от начала, поскольку число a_i может быть больше n, то будем рассматривать только смещение без учета полных проходов ряда, т.е. $a_i\ mod\ n$.
2) Удалим из массива $spear$ элемент на позиции $a_i\ mod\ n$.
3) Вставим удаленный элемент в начало массива.

Проделаем так k раз.

Затем найдем на какой позиции в массиве $spear$ стоит число m.

Задача B. Дон Кихот и Санчо Панса (100 баллов)

Рассмотрим как будет меняться позиция m - того копья в зависимости от a_i.

1) Если $a_i \bmod n < m - 1$, то i-тый день не влияет на позицию любимого копья, $mnew = mold$.
2) Если $a_i \bmod n = m - 1$, то в i-тый день выбрано любимое копье и его позиция станет равна 1, $mnew = 1$.
3) Если $a_i \bmod n > m - 1$, то в i-тый день перед любимым копьем станет на одно копье больше, его позиция станет равна $mnew = mold + 1$.

Таким образом достаточно поддерживать только позицию любимого копья в зависимости от a_i.

Задача C. Дон Кихот и ветряные мельницы (100 баллов)

Заметим, что если копье еще не сломалось, то идти за новым смысла нет.

Тогда будем идти вдоль мельниц слева направо и последовательно уничтожать их. Нам понадобится некоторое количество полных копий, за которыми понадобится возвращаться. После уничтожения мельницы последнее частично истраченное копье можно потратить на следующую мельницу.

Пусть у нас осталось $free$ ударов после уничтожения предыдущей мельницы, тогда на очередную мельницу уйдет $\left\lceil \frac{(a_i - free)}{k} \right\rceil$ копий, с учетом походом за последним копьем, которое может быть использовано не полностью. Общее время на удары мельниц копьями равно сумме a_i.

Задача D. Дон Кихот и трамвай (20, 40 баллов)

- При малых ограничениях на длину числа l можно решать задачу «в лоб», перебирая числа по возрастанию.

Задача D. Дон Кихот и трамвай (60 баллов)

Будем отдельно рассматривать младшую половину A и старшую половину B.

Для каждой возможной суммы цифр в младшей половине для какого числа больше A она получена. Аналогично для старшей половины.

Отдельно обработать случай $9 \ldots 9$.

Задача D. Дон Кихот и трамвай (100 баллов)

Будем конструировать следующий счастливый номер. Пусть sum_1, sum_2 это суммы в левой и правой половине числа соответственно, т.е. sum_1- сумма старших разрядов, sum_2 - сумма младших разрядов.

Будем рассматривать разряды младшей половины от младших к страшим. Если текущий разряд можно увеличить, так, чтобы равное изменение суммы цифр можно будет реализовать в старшей половине, то минимальное изменение найдено, необходимо сконструировать само число. Если в текущем разряде допустимого увеличения нет, то текущий разряд нужно обнулить и пересчитать значение sum_2.

Как только такая цифра найдена нужно сконструировать само число.

Отдельно следует обработать случай, когда число имеет вид $9 \ldots 9$.

Задача E. Дон Кихот и девиз (10 баллов)

При небольших ограничениях все подстроки можно сгенерировать и найти количество уникальных. Любым способом.

Задача E. Дон Кихот и девиз (30 баллов)

При ограничениях $n \leq 1000$.

Будем генерировать все подстроки и помещать их в set(C++), unordered_set(C++11), set(Python). Размер будет соответствовать количеству различных.

Задача E. Дон Кихот и девиз (60 баллов)

При ограничениях $n \leq 10000$ для подсчета количества уникальных строк можно использовать метод хэширования.

Полиномиальный хэш строки $s = s_0 s_1 \ldots s_{n-1}$ есть $p^{n-1}s_0 + p^{n-2}s_1 + \cdots + s_{n-1}$, где p обычно выбирают простым числом больше размера алфавита. Вычисления следует выполнять по некоторому большому модулю mod, который должен быть взаимно простым с p.

Просчитаем хеш для всех префиксов строки длины n, составленной повторением девиза.

Тогда полиномиальный хэш любой подстроки вычисляется за $O(1)$. Подсчет количества различных хэшей можно выполнить с помощью структуры unordered_set(C++), set(Python) или аналогичной.

Итоговая сложность $O(n^2)$.

Задача E. Дон Кихот и девиз (100 баллов)

Рассмотрим возможные способы образования новых построк с помощью повторения девиза.

Пусть наш девиз s, $k = len(s)$ составим строку из q девизов $\underbrace{s + \cdots + s}_{q \text{ раз}}$, тогда общий вид образуемых подстрок есть некоторый суффикс s', некоторое количество t полных девизов $\underbrace{s + \cdots + s}_{t}$, некоторый префикс девиза s''. Либо только префикс или только суффикс.

Для строки $s + s + s$ просчитаем ans_1 - количество различных подстрок, где $t \leq 1$.

Для этого применим технику для решения задачи на 60 баллов.

Задача E. Дон Кихот и девиз (100 баллов, продолжение).

Для ограничений $n \leq 1000000$ нужно эффективно подсчитать количество новых подстрок, которые образуются при приписывании справа еще одной копии девиза.

Для этого припишем справа к $s + s + s$ еще одну копию девиза, т.е получим строку $s + s + s + s$ и с помощью вычисления хешей найдем количество новых строк, которые образуются при приписывании некоторого префикса s длины i, запишем их в массив $count[i], i = 0, 1, \ldots, k$.

Аналогично учтем последнюю неполную копию девиза.

Итоговый ответ:

$ans = ans_1 + \frac{n-3\,k}{3\,k} * count[k] + count[(n - 3\,k)\ mod\ (3k)]$,
деление нацело, для Python //.

Итоговая сложность $O(k^2)$.