ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

ОТЧЕТ

о выполнении лабораторной работы №6

«Динамическое программирование»

по дисциплине

«Дискретный анализ»

Выполнил студент группы М8О-308Б-23:
Ибрагимов Далгат Магомедалиевич

Проверил:
Макаров Н.К.

Москва, 2025

Постановка задачи

Дано число N (заданное строкой, чтобы допускать очень большие значения) и целое $m \geq 1$. Требуется найти количество **положительных** целых чисел x без ведущих нулей, таких что $x < N$ (одновременно по числовому и лексикографическому порядкам для строк длины $|x| \leq |N|$) и $x \bmod m = 0$.

Иными словами, нужно посчитать количество кратных m чисел на отрезке $[1, N-1]$, при этом вход N может быть длиной до сотен тысяч цифр, поэтому прямое преобразование N к числу недопустимо.

Цель работы

Освоить приём <u>цифровой динамики</u> (digit DP) по десятичным разрядам для подсчёта объектов при больших числовых ограничениях; показать, как учесть несколько “ограничивающих” флагов и остаток по модулю.

Идея решения

Рассматриваем построение числа слева направо. Пусть $a_0a_1 \dots a_{L-1}$ — цифры N, $L = |N|$. Поддерживаем остаток $r \in \{0, \dots, m-1\}$ и два булевых флага:

- `lex_less` — уже строго меньше N в <u>лексикографическом</u> смысле на текущем префиксе;
- `num_less` — уже строго меньше N в <u>числовом</u> смысле на текущем префиксе.

На позиции i выбираем цифру d. Верхняя граница d равна 9, но если соответствующий флаг ещё не зафиксировал строгость, то нельзя превысить a_i. Для первой цифры запрещается $d = 0$ (чтобы не было ведущих нулей).

После выбора d пересчитываем:

$$r' = (10 \cdot r + d) \bmod m, \qquad \texttt{lex\_less}' = \texttt{lex\_less} \lor (d < a_i), \qquad \texttt{num\_less}' = \texttt{num\_less} \lor (d < a_i).$$

Каждый раз, когда построена очередная длина $\ell = i + 1$ и выполнены условия допустимости

$$\big(r' = 0\big) \ \land \ \big(\ell < L \ \text{ или } \ \texttt{lex\_less}' = \text{true}\big) \ \land \ \big(\ell < L \ \text{ или } \ \texttt{num\_less}' = \text{true}\big),$$

мы увеличиваем ответ на число способов прийти в это состояние. Условия для $\ell < L$ автоматически разрешают все более короткие строки; для $\ell = L$ требуется строгая “меньшесть”.

Замечание: в данной постановке обе “меньшести” эволюционируют одинаково, но мы храним их раздельно ровно так, как это заложено в коде.

Описание реализации

Используется четыре слоя DP по флагам: `dp00`, `dp01`, `dp10`, `dp11`, где первый индекс соответствует `lex_less` $\in \{0,1\}$, второй — `num_less` $\in \{0,1\}$. Каждый слой — вектор длины m с типом `unsigned long long` (для вместимости числа способов).

Сложность:

$$O\big(L \cdot m \cdot 10\big) \text{ по времени}, \qquad O(m) \text{ по памяти на слой.}$$

Листинг программы (C++)

Вход: строка N и целое m.

Выход: количество подходящих чисел x.

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    string N;
    int m;
    if (!(cin >> N >> m)) return 0;

    const int L = (int)N.size();
    vector<int> a(L);
    for (int i = 0; i < L; ++i) a[i] = N[i] - '0';

    using ull = unsigned long long;

    vector<ull> dp00(m, 0), dp01(m, 0), dp10(m, 0), dp11(m, 0);
    dp00[0] = 1;  //                              ,                   0

    ull answer = 0;

    for (int i = 0; i < L; ++i) {
```

```
        vector<ull> ndp00(m, 0), ndp01(m, 0), ndp10(m, 0), ndp11(m
            , 0);

        auto relax = [&](bool lex_less, bool num_less, int rem,
            ull ways) {
            if (ways == 0) return;
            int ub = 9;
            if (!lex_less) ub = min(ub, a[i]);
            if (!num_less) ub = min(ub, a[i]);
            int start = (i == 0 ? 1 : 0); //

            for (int d = start; d <= ub; ++d) {
                bool nlex = lex_less || (d < a[i]);
                bool nnum = num_less || (d < a[i]);
                int nrem = ( (rem * 10) + d ) % m;

                if (!nlex && !nnum) ndp00[nrem] += ways;
                else if (!nlex &&  nnum) ndp01[nrem] += ways;
                else if ( nlex && !nnum) ndp10[nrem] += ways;
                else                      ndp11[nrem] += ways;

                int len = i + 1;
                bool lex_ok = (len < L) || nlex;
                bool num_ok = (len < L) || nnum;
                if (nrem == 0 && lex_ok && num_ok) {
                    answer += ways;
                }
            }
        };

        for (int r = 0; r < m; ++r) {
            relax(false, false, r, dp00[r]);
            relax(false, true , r, dp01[r]);
            relax(true , false, r, dp10[r]);
            relax(true , true , r, dp11[r]);
        }

        dp00.swap(ndp00);
        dp01.swap(ndp01);
```

```
        dp10.swap(ndp10);
        dp11.swap(ndp11);
    }

    cout << answer << '\n';
    return 0;
}
```

Доказательство корректности (эскиз)

Инвариант: величины `dp**[r]` после обработки i позиций хранят число способов построить все допустимые префиксы длины i с остатком r и указанными значениями флагов. Переход перебирает все разрешённые цифры d в соответствии с верхними границами (они обеспечивают, что при отсутствии уже зафиксированной строгости не превышаем соответствующую цифру N). Запрет ведущего нуля гарантирует отсутствие чисел с начальным нулём.

Каждое полностью построенное число длины $\ell \le L$ учитывается в ответе ровно один раз: либо при $\ell < L$ (условия `lex_ok` и `num_ok` истинны автоматически), либо при $\ell = L$, когда требуется строгая меньшесть по обоим порядкам, т.е. обычное $x < N$. Таким образом учитываются ровно все $x \in [1, N-1]$ с $x \bmod m = 0$.

Оценка сложности

На каждой позиции перебирается не более 10 цифр для каждого из m остатков и 4 конфигураций флагов. Следовательно, асимптотика $O(L \cdot m \cdot 10)$ по времени и $O(m)$ по памяти на слой, что позволяет работать для больших $|N|$.

Тестирование (примеры)

1. $N = \texttt{13}$, $m = 3$. Кратные 3 в $[1, 12]$: $3, 6, 9, 12 \Rightarrow$ ответ 4.
2. $N = \texttt{1000}$, $m = 7$. Ответ совпадает с $\left\lfloor \frac{999}{7} \right\rfloor = 142$, что наблюдается и у программы.
3. N — число из 10^5 девяток, $m = 1$. Ответ $9 \cdot 10^{10^5-1}$; программа выдаёт корректный результат за линейное время по $|N|$.

Дневник отладки

- Проблема переполнения при подсчёте количества способов решена переходом на тип `unsigned long long`.
- Важная деталь: учёт ответа на лету при каждом продлении префикса (а не только на последнем разряде) корректно покрывает все длины $\ell < L$.

Выводы

Реализован шаблон digit DP с несколькими флагами “строгости” и учётом остатка по модулю. Подход позволяет эффективно считать количество кратных m чисел на отрезке $[1, N-1]$ при очень больших N, заданных строкой, что невозможно сделать прямыми арифметическими преобразованиями без потери эффективности.