

Испорченное число

Решение .

Если формулировать решение коротко, то хватит двух слов - динамическое программирование. Знатокам этого достаточно - задача несложная. Тем же, кто не знает, что это за зверь, я советую сначала заглянуть в *учебные материалы*. Например, начать с того, что тщательно разобрать выложенную там статью

М.Рейтман. Динамическое программирование, "Квант", 10, 1991.

Далнейшее изложение не зависит от предварительных знаний, но с ними вы сможете не только увидеть *как* решается задача, но и *почему* она решается именно так, а это важнее, это - понимание метода. А метод этот применим очень часто, в том числе и в олимпиадных задачах. Изрядное количество примеров и прекрасное изложение метода приводится в классической книге

Т.Кормен, Ч.Лейзерзон, Р.Ривест, К.Штайн. Алгоритмы: построение и анализ, 2-е издание: М., Издательский дом "Вильямс", 2005,
фрагмент из которой также лежит в учебных материалах.

Кроме того отмечу книгу

Х.А.Таха. Введение в исследование операций, 6-е издание: М., Издательский дом "Вильямс", 2001,
в которой приводится масса примеров (и разборов, конечно) задач с "экономическим" содержанием.

И, завершая тему учебных материалов, всё-таки не удержусь и добавлю к ним небольшой фрагмент из книги группы авторов

Простое и сложное в программировании: М., Наука, 1988.
Уж очень хорошо там изложена задача о раскрое многоугольника. Эта задача – вечнозелёная, излагается во многих источниках, но мне нравится именно этот вариант.

О динамическом программировании написано ещё в очень многих книгах и статьях, но хватит, пожалуй, для начала и этого ☺.

Вернёмся, всё-таки, к нашей задаче...

Итак, обозначим $R_m(i)$ наименьшее m -значное число, которое при делении на N даёт остаток i и согласовано с числом, образованным первыми m цифрами испорченного числа (если в испорченном числе в какой-то позиции задана некоторая цифра, то и в соответствующей позиции числа $R_m(i)$ стоит та же самая цифра). Если такого числа не существует, то полагаем $R_m(i) = +\infty$. И тогда число $R_{\text{Length}}(0)$ будет ответом, где через Length обозначим длину испорченного числа.

Как же вычислять $R_m(i)$? В соответствии с методом – последовательно для $m=1, 2, \dots, \text{Length}$. Зная все $R_m(i)$, $i = 0, 1, \dots, N-1$, мы сможем найти и все $R_{m+1}(i)$, $i = 0, 1, \dots, N-1$

нижеописанным способом.

Сначала два очевидных замечания.

Замечание 1. Пусть A – некоторое m -значное число, а B – число, полученное дописыванием к A в конце цифры d . Тогда $B=10 \cdot A+d$ и, разумеется, $B \bmod N = (10 \cdot A+d) \bmod N = (10 \cdot (A \bmod N) + d) \bmod N$.

Замечание 2. Пусть A' и B' – два $(m+1)$ -значных числа, а A и B – два числа, получающихся из A' и B' , соответственно, отбрасыванием последней цифры. Тогда, если $A' \leq B'$, то $A \leq B$.

А теперь, зная все $R_m(i)$, построим все $R_{m+1}(i)$, $i = 0, 1, \dots, N-1$.

1 случай. На $(m+1)$ месте в испорченном числе стоит некоторая цифра d .

Тогда $(10 \cdot R_m(i)+d) \bmod N = (10 \cdot i+d) \bmod N$. Берём число $R_m(i)$, приписываем к нему справа цифру d , и записываем полученное число в $R_{m+1}((10 \cdot i+d) \bmod N)$. Если $(10 \cdot i+d) \bmod N$ равны для нескольких различных i , то в $R_{m+1}((10 \cdot i+d) \bmod N)$ записываем меньшее из получающихся чисел. Если каким-то из чисел $R_{m+1}(i)$ ни разу ничего не присваивали, то полагаем такие числа равными $+\infty$ – понятно, что в таком случае подходящих чисел просто не существует (если такие были, то мы получили бы наименьшее из них при выполнении описанного действия).

2 случай. На $(m+1)$ месте в испорченном числе стоит вопросительный знак.

Выполняем то же самое действие, только в качестве цифры d берём поочередно все цифры от 0 до 9 (если вопросительный знак стоит на первом месте – то от 1 до 9).

Вот, собственно, и весь алгоритм: последовательно вычисляем все $R_m(i)$, $m=1, 2, \dots, \text{Length}$, $i = 0, 1, \dots, N-1$; $R_{\text{Length}}(0)$ – искомый ответ. Осталось решить некоторые технические вопросы, но здесь особых трудностей не возникает: числа можно хранить в ASCII-строках, для вычисления всех $R_m(i)$, $m=1, 2, \dots, \text{Length}$, $i = 0, 1, \dots, N-1$ достаточно завести два одномерных массива из N чисел, инициализация массива выполняется так: $R_0(0) = 0$ (или пустая строка), $R_0(i) = +\infty$, $i = 1, 2, \dots, N-1$.

Оценим сложность алгоритма. Каждая цифра во входном (испорченном) числе требует $O(N)$ операций, каждый вопросительный знак – $O(pN)$, где p – основание системы счисления, в нашем случае, конечно же, 10. Итого, получаем, сложность алгоритма $O(N \cdot (\text{Length} + pq))$, где q – количество вопросительных знаков, $p=10$ – основание системы счисления, Length – длина испорченного числа.

Это решение реализовано в файле `multiple0.pas`.

Неприятное ощущение остаётся от того, что мы довольно много времени тратим на обработку цифр испорченного числа. В самом деле, избавиться от этого совсем несложно – давайте отдельно обрабатывать цифры испорченного числа, и отдельно – вопросительные знаки.

Итак, разобьём испорченное число на сумму двух чисел, A_1 и A_2 : первое получается заменой всех вопросительных знаков в испорченном числе на 0, а второе – заменой на 0 всех цифр в испорченном числе. Понятно, что испорченное число равно сумме $A_1 + A_2$. Вычислить $A_1 \bmod N$ не представляет никаких трудностей. И тогда нам остаётся найти наименьшее A_2 , которое при делении на N даёт определённый остаток:

$$A_2 \bmod N = (N - A_1) \bmod N.$$

Нулевые цифры в числе A_2 на остаток не влияют. Остаются только вопросительные знаки. Пусть в испорченном числе имеется q вопросительных знаков, и они расположены на местах $d_1 > d_2 > \dots > d_q$. Здесь мы отсчитываем позиции справа налево, считая, что крайний правый символ стоит на нулевом месте. Если в числе A_2 вместо i -го вопросительного знака (на месте d_i) стоит цифра c_i , то

$$\begin{aligned} A_2 \bmod N &= (c_1 \cdot 10^{d_1} + c_2 \cdot 10^{d_2} + \dots + c_q \cdot 10^{d_q}) \bmod N = \\ &= (c_1 \cdot (10^{d_1} \bmod N) + c_2 \cdot (10^{d_2} \bmod N) + \dots + c_q \cdot (10^{d_q} \bmod N)) \bmod N. \end{aligned}$$

Задача свелась к следующей: найти наименьшее возможное число A_2 такое, что $A_2 \bmod N = (N - A_1) \bmod N$, а в тех позициях, в которых в испорченном числе стоят цифры, у него стоят 0. И решать мы её будем аналогично.

Пусть $R_m(i)$ – это наименьшее число, которое при делении на N даёт в остатке i , все цифры, соответствующие цифрам (не вопросительным знакам) испорченного числа равны 0, и все цифры, соответствующие вопросительным знакам, стоящим правее i -го вопросительного знака (на позициях $d_{i+1}, d_{i+2}, \dots, d_q$), также равны 0.

Замечание 1'. Пусть D – число, полученное установкой в числе $R_m(i)$ на позицию d_{i+1} цифры c_{i+1} . Тогда $D \bmod N = (R_m(i) + c_{i+1} \cdot 10^{d_{i+1}}) \bmod N =$
 $= (R_m(i) \bmod N + c_{i+1} \cdot (10^{d_{i+1}} \bmod N)) \bmod N = (i + c_{i+1} \cdot (10^{d_{i+1}} \bmod N)) \bmod N.$

И теперь, действуя аналогично тому, как мы действовали в первом решении, поочередно вычисляем все $R_m(i)$, $m=1, 2, \dots, q$, $i=0, 1, \dots, N-1$; затем подставляем цифры числа $R_m((N - A_1) \bmod N)$ вместо вопросительных знаков в испорченное число.

Это решение реализовано в файле `multiple.pas`.

И, в заключение, ещё немного материалов, на этот раз – в форме интернет-ссылок:

<http://ips.ifmo.ru/courses/course1/chG/> - сайт Internet Programming School. Эта ссылка ведёт на главу, посвящённую динамическому программированию, но там есть и много других интересных уроков.

<http://club.shelek.com/viewart.php?id=131> - материал во многом пересекается с главой из уже упоминавшейся (и выложенной в учебных материалах) книги *Т.Кормен и др.* Алгоритмы: построение и анализ, но зато изложен довольно кратко.

http://potential.org.ru/bin/view/Info/ArtDt200509292052PH3C1J9#Инварианты_и_оптимальные_решения - пара интересных примеров

<http://rain.ifmo.ru/cat/view.php/theory/algorithm-analysis/dynamic-programming-2004> - просто очень интересный сайт, посвящённый алгоритмам. Краткость изложения сочетается с ясностью, что всегда приятно, а достигнуть этого непросто. И, опять-таки, очень много и другого материала, посвященного анализу и решению алгоритмических задач.