II (районний) етап Всеукраїнської учнівської олімпіади з інформатики

Київ, 15 грудня 2013 р.

Максимальна оцінка за кожну з чотирьох задач — 100 балів.
Для всіх задач обмеження на час — 1 секунда / тест; обмеження на пам'ять — 256 МБ.

Розв'язки задач і набір тестів буде оприлюднено на сайтах **kievoi.narod.ru** та **soi.org.ua**.
Автор задач — Данило Мисак.

---

1. Письмо (назва програми: `writing.pas` / `writing.cpp`)

На уроці письма й каліграфії Петрик один або кілька разів поспіль виписав у зошиті одне й те саме натуральне число. В результаті утворилося шестицифрове число n. Допоможіть учительці з'ясувати, яке найменше число міг виписувати Петрик.

Вхідні дані
У вхідному файлі вказане шестицифрове натуральне число n.

Вихідні дані
У вихідний файл виведіть найменше натуральне число, яке міг виписувати Петрик.

Приклади

Вхідний файл `writing.in`	Вихідний файл `writing.out`
333333	3
525252	52
171171	171
240982	240982

2. Література (назва програми: `reading.pas` / `reading.cpp`)

Щоб уникнути чергової двійки з літератури, Петрик має виконати домашнє завдання: вибрати з хрестоматії два різних твори загальним обсягом s сторінок і прочитати їх. Знаючи обсяг кожного з n творів, що є в хрестоматії, допоможіть хлопцю з'ясувати, скільки він має варіантів вибору.

Вхідні дані
У першому рядку вхідного файлу вказано натуральні числа s та n, не менші за 2. У другому рядку записано n натуральних чисел — обсяги (кількості сторінок) відповідних творів із хрестоматії. Усі числа у вхідному файлі (включно з числами s та n) не перевищують $200\,000$.

Вихідні дані
У вихідний файл виведіть єдине число — кількість способів вибрати з хрестоматії пару творів загальним обсягом рівно s сторінок. Відомо, що ця кількість не перевищує 10^9.

Приклади

Вхідний файл **reading.in**	Вихідний файл **reading.out**
4 5 2 2 3 2 1	4
10 3 6 2 10	0

Пояснення до прикладів

У першому прикладі Петрик може вибрати один із чотирьох варіантів: прочитати перший і другий твори з хрестоматії; або перший і четвертий; або другий і четвертий; або третій і п'ятий.

У другому прикладі жодні два твори не дають у сумі рівно 10 сторінок.

3. Фізкультура (назва програми: **pe.pas** / **pe.cpp**)

На уроці фізкультури учні вишукувалися в шеренгу в деякому зручному для них порядку. Конче потрібно, однак, щоб усі стояли за зростом — у порядку від найвищого учня до найнижчого. Для цього вчитель може один або кілька разів зробити таке: назвати ім'я деякого учня і попросити його перейти в кінець шеренги. За яку найменшу кількість подібних дій учителю вдасться впорядкувати шеренгу бажаним чином?

Зріст кожного з n учнів відомий. Якщо кілька з них однакового зросту, то між собою такі учні в кінцевому розташуванні можуть стояти в довільному порядку (на вибір учителя). За бажання вчитель може переставляти одного й того самого учня кілька разів.

Вхідні дані

У першому рядку вхідного файлу вказано натуральне число n, не менше за 2 і не більше за $200\,000$. У другому рядку записано n натуральних чисел, що не перевищують $2 \cdot 10^9$, — зріст кожного з n учнів у порядку, в якому вони стоять у шерензі на початку уроку.

Вихідні дані

У вихідний файл виведіть єдине число — найменшу кількість переставлянь, які доведеться зробити вчителю, щоб упорядкувати шеренгу за незбільшенням зросту учнів.

Приклади

Вхідний файл **pe.in**	Вихідний файл **pe.out**
7 126 145 141 134 130 141 134	3
3 173 172 169	0

Пояснення до прикладів

У першому прикладі вчитель може діяти так: спочатку переставити в кінець шеренги першого з двох учнів зросту 134, далі переставити учня зросту 130, а потім — 126. Швидше ніж за три дії впорядкувати шеренгу вчителю не вдасться.

У другому прикладі учні відразу стоять у потрібному порядку. Переставляти нікого не потрібно.

4. Математика (назва програми: `math.pas` / `math.cpp`)

Щоб заробити чергові дванадцять балів з математики, Петрик має розв'язати багато однотипних задач на переливання. Типова задача з Петрикового підручника виглядає так.

Є три порожні посудини місткістю a, b та c літрів. За одну операцію можна виконати якусь із трьох дій:

- Повністю заповнити одну з посудин водою з джерела. До операції посудина може бути або порожньою, або вже частково наповненою. Кількість води у джерелі необмежена.
- Вилити з однієї з посудин усю воду, що в ній міститься. До операції посудина не обов'язково має бути повністю заповненою.
- Перелити з однієї посудини в іншу рівно стільки води, щоб або посудина, куди переливають воду, повністю заповнилася, або — якщо води для цього недостатньо — посудина, звідки переливають воду, спорожніла.

Операції кожного з трьох типів можна виконувати як завгодно багато разів. Треба з'ясувати, за яку найменшу кількість операцій і чи можливо взагалі «відміряти» n літрів, тобто зробити так, щоб принаймні в одній посудині опинилося рівно n літрів води.

Вхідні дані

У першому рядку вхідного файлу записано три натуральних числа a, b та c, а у другому рядку — натуральне число n. Усі чотири числа не перевищують 100.

Вихідні дані

У вихідний файл виведіть єдине число — найменшу кількість операцій, які потрібно провести з посудинами, щоб в одній з них опинилося рівно n літрів води. Якщо цього досягти неможливо, виведіть число 0.

Приклади

Вхідний файл `math.in`	Вихідний файл `math.out`
4 6 9 7	4
3 10 3 1	5
5 4 7 9	0

Пояснення до прикладів

У першому прикладі відміряти 7 літрів можна щонайменше за 4 кроки. Наприклад, таким чином:

1) Налити воду в 4-літрову посудину.
2) Перелити з неї усі 4 літри води в 6-літрову посудину.
3) Налити воду в 9-літрову посудину.
4) Перелити з неї 2 літри в 6-літрову посудину (та заповниться). У 9-літровій посудині залишиться рівно 7 літрів води.

У другому прикладі відміряти 1 літр можна щонайменше за 5 операцій. Приміром, так:

1) Налити воду в 10-літрову посудину.
2) Перелити з неї 3 літри в першу посудину.
3) Перелити з неї ж іще 3 літри в третю посудину. У 10-літровій посудині залишиться 4 літри води.
4) Вилити воду з першої 3-літрової посудини.
5) Перелити 3 літри води з 10-літрової посудини в порожню 3-літрову. У 10-літровій залишиться 1 літр.

У третьому прикладі кількість літрів, яку потрібно відміряти, перевищує місткість кожної з трьох посудин. Тож відміряти відповідну кількість води, очевидно, не вдасться.

Ідеї розв'язання

1. Письмо

З умови задачі випливає, що шукане число може бути лише одноцифровим, двоцифровим, трицифровим або шестицифровим.

Прямий спосіб розв'язання задачі передбачає розбиття зчитаного числа n на цифри (або посимвольне його зчитування) і порівняння між собою пар цифр:

- якщо всі цифри числа n однакові, то відповідь одноцифрова і дорівнює першій цифрі числа n;
- інакше якщо третя і п'ята цифри числа n дорівнюють першій, а четверта і шоста — другій, то відповідь двоцифрова і дорівнює числу, яке утворюють перші дві цифри числа n;
- інакше якщо четверта цифра числа n дорівнює першій, п'ята — другій, а шоста — третій, то відповідь трицифрова і дорівнює числу, яке утворюють перші три цифри числа n;
- інакше відповідь шестицифрова і дорівнює самому числу n.

Математичний спосіб розв'язання ґрунтується на спостереженні, що шестицифрове число n дорівнює добутку числа-відповіді та одного з таких чотирьох чисел: $111\,111$ (якщо відповідь одноцифрова), $10\,101$ (якщо відповідь двоцифрова), 1001 (якщо відповідь трицифрова) або просто 1 (якщо відповідь шестицифрова). Таким чином, відповіддю є частка числа n та першого з чисел $111\,111$, $10\,101$, 1001, 1, на яке n ділиться без остачі.

2. Література

Безпосередній перебір принесе лише частковий бал, бо для великих n програма не встигне за відведений час (секунду) перебрати всі можливі пари чисел. Замість цього можна скористатися іншим підходом. Для кожного числа від 1 до $200\,000$ слід підрахувати, скільки творів мають даний обсяг: завести масив cnt, у комірці $cnt[p]$ якого буде зберігатися кількість творів обсягу p ($1 \leqslant p \leqslant 200\,000$), і після зчитування кожного наступного числа збільшувати відповідну комірку масиву на 1. Далі через $\left[\frac{s-1}{2}\right]$ ми позначаємо найбільше ціле число, що не перевищує $\frac{s-1}{2}$. Для всіх значень p у межах від 1 до $\left[\frac{s-1}{2}\right]$ включно добуток $cnt[p] \cdot cnt[s-p]$ дорівнюватиме кількості пар творів, один із яких має обсяг p, а інший — обсяг $s-p$ сторінок; при цьому $p < s-p$. Сума всіх $\left[\frac{s-1}{2}\right]$ таких добутків буде кількістю варіантів вибрати пару творів сумарним обсягом s сторінок за умови, що обсяги цих двох творів різні. Залишається врахувати пари творів однакового обсягу: якщо s непарне, таких пар немає, а інакше їх рівно $\frac{cnt[s/2]\cdot(cnt[s/2]-1)}{2}$.

3. Фізкультура

Передусім зауважимо, що переставляти одного й того самого учня кілька разів учителю не знадобиться: із послідовності дій можна вилучити всі переставляння даного учня, крім останнього, не збільшивши загальної кількості дій і не змінивши кінцевого результату. Стане в пригоді й таке спостереження: всі учні, яких учитель не переставляв у кінець шеренги, в підсумку будуть стояти на її початку, причому в тому ж відносному порядку, в якому вони стояли до переставлянь. Отже, це найвищі з усіх учні, що стоять у порядку незбільшення зросту. І навпаки: якщо в початковій шерензі ми зафіксуємо і не будемо рухати кількох найвищих учнів, що стоять у порядку незбільшення зросту, а інших переставлятимемо в кінець шеренги в порядку від найвищого до найнижчого, то в підсумку впорядкуємо учнів потрібним чином. Найменшій кількості переставлянь відповідає при цьому найдовша послідовність найвищих учнів, що стоять у порядку незбільшення зросту: якщо таких учнів k, то шукана кількість дій дорівнює $n-k$. Залишається знайти найдовшу таку послідовність, тобто число k.

Розв'язання сортуванням. Послідовність чисел, подану у вхідному файлі, можна занести відразу у два масиви, причому після зчитування один з них слід відсортувати за незростанням. Рухаючись зліва на-

право у невідсортованому масиві, треба додавати до лічильника (змінної k) одиницю кожного разу, коли натрапляємо на чергове число з відсортованого масиву.

Відзначимо, що просте сортування за квадратичний час дасть лише частковий бал, тож потрібно використати впорядкування злиттям або інший швидкий метод сортування масиву.

Розв'язання зі стеком. Назвімо найдовшу послідовність найвищих учнів, що стоять у порядку незбільшення зросту, *суперпідпослідовністю* даної послідовності. Суперпідпослідовність можна побудувати за один прохід вхідного масиву, тобто відразу під час зчитування файлу. Зчитавши перші p чисел вхідної послідовності, ми матимемо такі дані:

- стек s, що містить суперпідпослідовність послідовності з уже зчитаних p чисел (якщо $p = 0$, стек порожній);
- змінну m, яка дорівнює найбільшому з тих уже зчитаних чисел, які не входять у стек s (якщо $p = 0$, покладемо $m = 0$).

Очевидно, за таких умов після зчитування всіх n чисел змінна s буде містити саме ту послідовність, яку й потрібно відшукати. Залишається розібратися, як підтримувати змінні s та m в актуальному стані. Після зчитування $(p + 1)$-го числа, яке позначимо через h, їх слід оновити таким чином:

1) Зі стека s видалити всі числа, що строго менші за h. При цьому, якщо доведеться, відповідним чином збільшити значення m.
2) Якщо h не менше за m, додати його в стек; інакше проігнорувати.

Покажемо, що якщо після p-го кроку стек s містить суперпідпослідовність перших p чисел, то після $(p + 1)$-го кроку стек міститиме суперпідпослідовність нової послідовності з $p + 1$ числа. Насамперед, як випливає зі схеми роботи алгоритму, жодне число зі стека не може бути меншим за поточне значення змінної m. Тепер випишемо можливі варіанти:

- число h є строго більшим за останній (найменший) елемент стека;
- число h не перевищує останній елемент стека, але й не менше за m;
- число h є меншим за m.

Можна переконатися, що в кожному з цих трьох випадків алгоритм справді потрібним чином оновлює вміст стека.

Насамкінець додамо, що алгоритм працює лінійний час: кожне число послідовності додають у стек та видаляють із нього щонайбільше по одному разу. Саме в цьому полягає перевага даного способу розв'язання перед концептуально простішим методом сортування.

4. Математика

Станом назвімо впорядковану трійку чисел $(x; y; z)$, де x, y та z — кількості літрів води у першій, другій і третій посудинах відповідно. Якщо зі стану А за одну операцію з посудинами можна дістати стан Б, казатимемо, що стан Б *досяжний* зі стану А.

Для розв'язання задачі можна використати алгоритм, аналогічний пошуку в ширину на графі:

1) Заведемо чергу, в якій зберігатимемо стани — трійки цілих чисел. На практиці можна використати три окремих черги, елементами яких є цілі числа (відповідні компоненти трійок), і використовувати ці черги паралельно. Додамо в чергу початковий стан $(0; 0; 0)$.

2) Розглянемо стан А, що є на даний момент першим у черзі, і видалимо його з черги. Додамо в чергу всі стани, які досяжні зі стану А і які ми не додавали в чергу раніше. Запам'ятаємо, що кількість операцій, необхідна для одержання з початкового стану $(0; 0; 0)$ кожного з цих станів, на 1 більша, ніж для самого стану А. Щоби пам'ятати цю кількість, можна використати, наприклад, тривимірний масив, індексами якого слугує трійка чисел, що задає стан, а значенням комірки є відповідна кількість операцій. Цей же масив дозволить визначати, чи ми вже додавали в чергу той чи інший стан.

3) Якщо до черги було додано трійку, хоча б один компонент якої дорівнює n, відповідь знайдено (це кількість операцій, необхідна для досягнення даної трійки). Якщо черга спорожніла, а відповідь так і не знайдено, то відміряти n літрів води неможливо. Якщо ж черга ще не спорожніла і відповідь поки не знайдено, повертаємось до другого кроку алгоритму.

Додамо, що немає потреби окремо розглядати випадок $n > \max\{a; b; c\}$ або інші, коли відразу зрозуміло, що n літрів відміряти неможливо: алгоритм і так це з'ясує.

Щоб оцінити час виконання алгоритму, зауважимо, що з кожного стану можна дістати щонайбільше 12 інших: по 3 для операцій наповнення й виливання води та 6 для операцій переливання. Тому в найгіршому випадку — якщо n літрів відміряти важко або неможливо — час буде пропорційним загальній кількості досяжних станів (тобто всіх таких, які можна отримати з початкового стану $(0; 0; 0)$ за одну чи кілька послідовних операцій). Число ж досяжних станів не перевищує добутку $(a + 1)(b + 1)(c + 1)$.