Олімпіада з інформатики Солом’янського району

2012/13 н. р.

Автор задач — Данило Мисак

1. Петрик П’яточкін рахує склади́
(назва програми: `syllable.pas` / `syllable.dpr` / `syllable.cpp`)

Допитливий київський школяр Петрик П’яточкін якось зацікавився мовознавством — наукою про мову. Для важливого дослідження Петрику потрібно з’ясувати, скільки на́зви різних натуральних чисел мають складів, тобто скільки містять голосних літер. Допоможіть хлопцю, написавши програму, яка дає відповідь на це питання.

Вхідні дані
У вхідному файлі вказане натуральне число n, кількість складів у якому необхідно порахувати. Число n не перевищує 100.

Вихідні дані
У вихідний файл виведіть кількість складів, які у своїй назві українською мовою має число n.

Приклади

Вхідний файл `syllable.in`	Вихідний файл `syllable.out`
1	2
25	3

Пояснення до прикладів
Слово «один» має два склади (містить дві голосні літери). Назва числа 25 — «двадцять п’ять» — має три склади (містить три голосні літери).

2. Петрик П’яточкін купує книги
(назва програми: `books.pas` / `books.dpr` / `books.cpp`)

Щоб краще підготуватися до різноманітних олімпіад, у яких бере участь Петрик П’яточкін, хлопець замовляє книги у зарубіжному інтернет-магазині. На жаль, Петрику доводиться платити не тільки за самі книги, але також і за їх доставку в Україну: незалежно від кількості книг, доставлених хлопцю за один раз, Петрик платить за одну доставку d гривень. Крім того, українська митниця за один раз дозволяє безплатно провозити через кордон щонайбільше m книг, а якщо кількість книг перевищує m, митниця бере за кожну понаднормово завезену книгу t гривень податку. Перед олімпіадою з інформатики Петрик хоче купити в інтернет-магазині n книг. Він може замовити доставку всіх книг разом або як завгодно розподілити книги на довільну кількість окремих доставок. Допоможіть хлопцю визначити, яку найменшу загальну суму грошей він повинен витратити на доставки та на мито, щоб перевезти усі n книг в Україну.

Вхідні дані
У єдиному рядку вхідного файла записані чотири натуральні числа n, d, m, t — відповідно кількість книг, які треба перевезти; вартість однієї доставки; найбільша кількість книг, яку можна перевезти за одну доставку без сплати мита; розмір мита за кожну понаднормово завезену книгу. Відомо, що $n(d+t) < 2 \cdot 10^9$ і $m < 2 \cdot 10^9$.

Вихідні дані
У вихідний файл виведіть єдине натуральне число — найменшу сумарну кількість грошей, яку має сплатити Петрик за доставку і як мито, щоб отримати замовлені в інтернет-магазині книги.

Приклади

Вхідний файл `books.in`	Вихідний файл `books.out`
5 4 3 6	8
3 2 2 1	3

Пояснення до прикладів

У першому прикладі потрібно перевезти 5 книг. Якщо замовити їх однією доставкою, доведеться сплатити 4 грн за доставку і за $5-3=2$ понаднормово завезені книги по 6 грн, усього 16 грн. Якщо ж розподілити книги на дві доставки, наприклад 3 книги на першу і 2 книги на другу, треба буде сплатити $2\cdot 4=8$ грн за дві доставки, зате під час жодної з доставок платити за понаднормово завезені книги не доведеться. Три чи більше доставок будуть коштувати явно більше за 8 грн, тому 8 грн — найменша сума, яку доведеться витратити.

У другому прикладі потрібно перевезти 3 книги. Якщо замовити їх однією доставкою, треба буде сплатити $2+(3-2)\cdot 1=3$ грн. А за дві чи більше доставок доведеться заплатити вже не менше ніж $2\cdot 2=4$ грн.

3. Петрик П'яточкін пише олімпіаду

(назва програми: `olympiad.pas` / `olympiad.dpr` / `olympiad.cpp`)

Прочитавши чимало книжок про алгоритми, Петрик П'яточкін прийшов на районну олімпіаду з інформатики. Перед початком олімпіади з'ясувалося, що багато хто з учасників знає одне одного. Тож організатори вирішили убезпечитися й розсадити всіх учасників по двох кабінетах, де проходить олімпіада, у такий спосіб, щоб жодні два знайомі між собою учасники не сиділи в одному кабінеті. Напишіть програму, яка допоможе організаторам зробити це або скаже, що розсадити в такий спосіб учасників неможливо.

Вхідні дані

У першому рядку вхідного файла вказано два натуральних числа n та m — кількість учасників районної олімпіади та кількість пар знайомих учасників; $2 \leqslant n < 2000$, $1 \leqslant m < 450\,000$. У кожному з наступних m рядків задано по два числа a_i, b_i — номери знайомих між собою учасників, $1 \leqslant a_i < b_i \leqslant n$, $1 \leqslant i \leqslant m$. Жодна пара номерів a_i, b_i у вхідному файлі не повторюється. Крім того, вхідні дані гарантують, що є не більше ніж один спосіб розсадити учасників по двох кабінетах, щоб жодні два знайомі учасники не сиділи в одному кабінеті.

Вихідні дані

У першому рядку вихідного файла виведіть у порядку зростання номери учасників, які мають сидіти в кабінеті разом із учасником під номером 1 (Петриком П'яточкіним) включно із самим числом 1 (Петриком). У другому рядку виведіть у порядку зростання номери учасників, які повинні сидіти в іншому кабінеті. Якщо жодне розташування учасників не задовольняє умову задачі, в обох рядках виведіть по нулю.

Приклади

Вхідний файл `olympiad.in`	Вихідний файл `olympiad.out`
5 6 2 5 2 3 1 5 4 5 3 4 1 3	1 2 4 3 5
3 3 1 2 2 3 1 3	0 0

Ідеї розв'язання

Петрик П'яточкін рахує склади́

Звичайно, можна вручну підрахувати кількість складів у кожному з чисел від 1 до 100 і створити програму з розгалуженнями типу IF THEN ELSE або CASE / SWITCH, які залежно від числа на вході повертають потрібний результат. Але цей підхід досить трудомісткий і скоріше за все учасник, який пішов таким шляхом, зробить принаймні декілька помилок у підрахунку або під час занесення результатів у програму.

Натомість можна помітити, що назви всіх двоцифрових чисел, починаючи з числа 20, утворюються шляхом сполучення слова, що позначає кількість десятків («двадцять», «сорок», «дев'яносто») і слова, що позначає кількість одиниць («один», «вісім»). При цьому у назвах чисел, кратних 10, компонент одиниць відсутній. Тому слід окремо внести у програму (наприклад, як масив) кількості складів у назвах чисел 1, 2, 3, …, 18, 19, 20, 30, 40, …, 90, 100, а решту відповідей давати як суму двох значень, занесених для числа, що позначає кількість десятків, і числа, що позначає кількість одиниць заданого у вхідному файлі значення *n*.

Петрик П'яточкін купує книги

Теорія

Якщо Петрик замовить *k* окремих доставок, то за умови оптимального розподілу книг між доставками хлопцю доведеться сплатити $kd + \max\{0, n-km\}\cdot t$ грн: *kd* грн за *k* доставок, а також $(n-km)t$ грн за $n-km$ понаднормово завезених книг, якщо $n-km>0$. Уведімо позначення

$$f(k) = kd + \max\{0, n-km\}\cdot t,\ \ k\in\mathbb{N},$$

і розгляньмо різницю $f(k+1)-f(k)$:

$$f(k+1)-f(k)=\begin{cases} d, & n-km<0,\\ d-(n-km)t, & 0\leqslant n-km<m,\\ d-mt, & n-km\geqslant m.\end{cases}$$

Запишімо це трохи інакше, позначивши через $n \bmod m$ остачу від ділення *n* на *m*:

$$f(k+1)-f(k)=\begin{cases} d, & k>\dfrac{n}{m},\\ d-(n \bmod m)t, & \dfrac{n}{m}-1<k\leqslant\dfrac{n}{m},\\ d-mt, & k\leqslant\dfrac{n}{m}-1.\end{cases}$$

Зрозуміло, що якщо $\dfrac{n}{m}<1$, тобто $n<m$, то замовляти більше ніж одну доставку Петрику сенсу немає. Тому відповіддю в цьому випадку є значення $f(1)$. Далі вважатимемо, що $\dfrac{n}{m}\geqslant 1$.

Якщо $d-mt\geqslant 0$, то $f(k+1)\geqslant f(k)$ незалежно від значення *k*. Тому мінімальне значення функція *f* набуває за найменшого *k*, тобто при $k=1$, а шукана сума грошей — це $f(1)$.

Якщо $d-mt<0$, але $d-(n \bmod m)t>0$, то $f(k+1)<f(k)$ при $k\leqslant\left[\dfrac{n}{m}\right]-1$ та $f(k+1)>f(k)$ при $k\geqslant\left[\dfrac{n}{m}\right]$. Тому мінімальне значення функція *f* набуває при $k=\left[\dfrac{n}{m}\right]$, а шукана сума грошей — це $f\left(\left[\dfrac{n}{m}\right]\right)$.

А якщо $d-(n \bmod m)t\leqslant 0$, то $f(k+1)\leqslant f(k)$ при $k\leqslant\left[\dfrac{n}{m}\right]$ та $f(k+1)>f(k)$ при $k\geqslant\left[\dfrac{n}{m}\right]+1$. Тому мінімальне значення функція *f* набуває при $k=\left[\dfrac{n}{m}\right]+1$, а шукана сума грошей — це $f\left(\left[\dfrac{n}{m}\right]+1\right)$.

Практика

На практиці досить написати програму, яка порівнює три числа — значення $f(1)$, $f\left(\left[\frac{n}{m}\right]+1\right)$ і, якщо $\left[\frac{n}{m}\right]\neq 0$, $f\left(\left[\frac{n}{m}\right]\right)$ — та виводить найменше з них. Усі обчислення слід проводити обережно, щоб не вийти за межі стандартного типу даних. Зокрема число $f\left(\left[\frac{n}{m}\right]+1\right)$ може перевищувати $2\cdot 10^9$, але в цьому випадку воно точно не буде найменшим із трьох чисел (бо, замовивши, приміром, одну доставку, Петрик заплатить менше ніж $d+nt\leqslant n(d+t)<2\cdot 10^9$ грн).

Алгоритм, який не враховує монотонності функції *f* при значеннях аргументу від 1 до $\left[\frac{n}{m}\right]$ та шукає найменше серед усіх чисел $f(1)$, $f(2)$, …, $f\left(\left[\frac{n}{m}\right]\right)$, $f\left(\left[\frac{n}{m}\right]+1\right)$, набирає лише частковий бал: значення $\left[\frac{n}{m}\right]$ попри обмеження, накладені в умові задачі, може досягати мільярда.

Петрик П’яточкін пише олімпіаду

Задачу можна переформулювати в термінах теорії графів. Нехай учасники олімпіади — це вершини графа, а ребро між двома вершинами проведене тоді й лише тоді, коли два відповідні учасники знайомі між собою. Нам необхідно розділити всі вершини графа на дві частини так, щоб кожне ребро графа сполучало вершини, що належать до різних частин. Граф, для якого це вдається зробити, називається дводольним, а відповідні його частини — долями.

Задачу можна розв’язати з допомогою пошуку <u>в глибину</u> або <u>в ширину</u>. Присвоївши вершині 1 (Петрику) кабінет № 1 і запустивши будь-який із типів пошуку з цієї вершини, будемо кожній новій пройденій вершині графа присвоювати кабінет № 2, якщо ми прийшли у неї з вершини, якій було присвоєно кабінет № 1, і навпаки. При цьому якщо дана вершина сполучена ребром хоча б з однією іншою вершиною, якій раніше присвоїли той самий кабінет, що й даній вершині, то розбити граф на дві частини неможливо.

Після завершення пошуку в глибину або в ширину, якщо вдалося присвоїти кабінети всім вершинам, маємо потрібний розподіл учасників у компоненті зв’язності графа, яка містить вершину 1, а інакше виводимо нулі. Якщо граф складається з єдиної компоненти зв’язності, виводимо знайдений розподіл. Якщо ж у графі є дві або більше компонент зв’язності, тобто якщо після завершення пошуку, запущеного з вершини 1, одна чи кілька вершин залишились невідвіданими, то слід також вивести нулі. Справді: якщо потрібний розподіл вершин усіх компонент зв’язності існує, то він не єдиний, адже розподіли різних компонент можна як завгодно комбінувати. А згідно з умовою задачі якби розподіл існував, то він мав би бути єдиним. Отже, у випадку двох і більше компонент зв’язності умова задачі гарантує відсутність потрібного розподілу учасників.

Насамкінець зауважимо, що обмеження $m<450\,000$ має суто технічний характер і пов’язане з необхідністю обмежити розмір вхідного файла.