Problema Cufere

Fişier de intrare `cufere.in`
Fişier de ieşire `cufere.out`

Alex, eroina din *Minecraft*, este foarte curajoasă și harnică. De-a lungul timpului, ea a depozitat în n cufere tot felul de obiecte fragile (de exemplu ouă) sau dure (de exemplu pietre).

Un cufăr este o cutie de lemn cu 27 de compartimente dispuse pe 3 rânduri, câte 9 pe fiecare rând. Într-un compartiment poate fi depozitat un grup de unul sau mai multe obiecte **identice**: maximum 16 obiecte fragile sau maximum 64 de obiecte dure. Pot fi mai multe compartimente care să conțină același tip de obiecte, iar unele compartimente pot fi goale.

Figura 1: Exemplu de cufăr înainte și după ordonare

Alex a etichetat atât compartimentele, cât și obiectele, cu numere construite după următoarea regulă:

- un obiect are drept etichetă un număr natural cuprins între 10 și 99, inclusiv, astfel: un număr prim, dacă este fragil, sau un număr compus, dacă este dur;
- toate obiectele identice primesc aceeași etichetă;
- un compartiment are drept etichetă un număr natural format din două valori alipite: numărul obiectelor din grupul depozitat în el, urmat de eticheta comună a acestora (de exemplu dacă eticheta compartimentului este 1994, înseamnă că în el este depozitat un grup de 19 obiecte, fiecare având eticheta 94);
- compartimentele goale sunt etichetate cu 0.

Alex vrea să **rearanjeze** obiectele din cufere, astfel încât:

- să fie valorificat spațiul, adică să fie ocupate cât mai puține cufere și, în cadrul unui cufăr, cât mai puține compartimente;
- să fie ocupate compartimentele din cuferele disponibile la rând, începând cu primul cufăr, și, în cadrul unui cufăr, începând cu primul rând și, în cadrul unui rând, de la stânga la dreapta. Cu alte cuvinte, se umple mai întâi cufărul 1, începând cu rândul 1, și pe fiecare rând de la stânga la dreapta, apoi cufărul al doilea, în aceeași manieră, și așa mai departe;
- obiectele sunt preluate în ordinea crescătoare a etichetelor și din totalul obiectelor identice se formează mai întâi grupuri cu număr maxim de obiecte, și doar ultimul grup poate fi, eventual, incomplet;
- fiecare din aceste grupuri se depozitează, pe măsura formării, în câte un compartiment al cufărului curent, iar dacă acesta se umple, se trece la cufărul următor.

După rearanjarea obiectelor, compartimentele sunt etichetate din nou, după aceeași regulă.

Cerințe

Dându-se cele n cufere, care conțin obiectele în ordinea inițială, Alex vă roagă să realizați un program care să determine:

1. pentru fiecare etichetă distinctă de obiect întâlnit în cele n cufere, numărul total al obiectelor cu acea etichetă;
2. noile etichete ale compartimentelor care compun cele n cufere, după rearanjarea obiectelor.

Date de intrare

Fișierul de intrare `cufere.in` conține pe prima linie numărul c reprezentând cerința care trebuie să fie rezolvată (1 sau 2), pe a doua linie numărul natural nenul n, cu semnificația din enunț, iar pe fiecare din următoarele $3n$ linii, câte 9 numere, reprezentând etichetele inițiale ale compartimentelor aflate pe câte un rând al unui cufăr, în ordinea în care ele se află în cufere, de la primul cufăr, până la ultimul, în cadrul fiecărui cufăr de la primul rând până la al treilea, iar în cadrul fiecărui rând de la stânga la dreapta. Numerele aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu.

Date de ieșire

Fișierul `cufere.out` va conține fie răspunsul pentru cerința 1 (dacă $c = 1$), fie răspunsul pentru cerința 2 (dacă $c = 2$).

Pentru cerința 1, pentru fiecare etichetă distinctă, în ordine strict crescătoare, se va afișa o pereche formată din eticheta respectivă și numărul obiectelor cu această etichetă. Fiecare pereche de numere va fi afișată pe câte o linie.

Pentru cerința 2, etichetele compartimentelor vor fi afișate corespunzător plasării lor în cufere, câte 9 pe fiecare linie a fișierului, de la primul cufăr până la ultimul, în cadrul fiecărui cufăr de la primul rând până la al treilea, iar în cadrul fiecărui rând de la stânga la dreapta.

Numerele aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu.

Restricții

- $c \in \{1, 2\}$
- $1 \leq n \leq 10\,000$
- Eticheta unui obiect este cuprinsă între 10 și 99, inclusiv.
- În cazul cerinței 2, se vor afișa etichetele pentru toate compartimentele, chiar dacă ele sunt goale sau provin din cufere complet goale.

#	Punctaj	Restricţii
1	40	$c = 1$
2	60	$c = 2$

Exemple

cufere.in	cufere.out
1 2 1488 1573 1437 4465 1099 1073 0 499 765 537 1173 4288 1273 2299 1555 1241 655 841 1141 237 5621 199 921 621 3465 1315 4155 1099 341 4765 6155 355 1099 6088 3988 255 4955 155 1329 1932 3099 114 3020 855 5555 1173 1388 673 2533 1488 1473 4033 2099 2065	14 1 15 13 20 30 21 71 29 13 32 19 33 65 37 21 41 34 55 241 65 152 73 79 88 182 99 107
2 2 1488 1573 1437 4465 1099 1073 0 499 765 537 1173 4288 1273 2299 1555 1241 655 841 1141 237 5621 199 921 621 3465 1315 4155 1099 341 4765 6155 355 1099 6088 3988 255 4955 155 1329 1932 3099 114 3020 855 5555 1173 1388 673 2533 1488 1473 4033 2099 2065	114 1315 3020 6421 721 1329 1932 6433 133 1637 537 1641 1641 241 6455 6455 6455 4955 6465 6465 2465 1673 1673 1673 1673 1573 6488 6488 5488 6499 4399 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Explicații

Exemplul 1

În acest exemplu se va rezolva cerința $c = 1$ și există $n = 2$ cufere. În cufere există:

- 1 obiect cu eticheta 14;
- 13 obiecte cu eticheta 15;
- 30 de obiecte cu eticheta 20;
- ...
- 107 obiecte cu eticheta 99.

Exemplul 2

În acest exemplu se va rezolva cerința $c = 2$ și există $n = 2$ cufere. După rearanjare, s-au plasat obiectele în ordinea crescătoare a etichetelor. Pentru primele trei etichete se formează câte un singur grup, aceastea fiind plasate în primele trei compartimente ale primului cufăr. Apoi, cele 71 de obiecte cu eticheta 21 (dure), sunt împărțite într-un grup de 64 (în compartimentul al patrulea), și un grup de 7 (în compartimentul al cincilea). La fel se procedează și cu celelalte obiecte, astfel încât primul cufăr este ocupat complet, primul rând al celui de-al doilea cufăr este parțial ocupat, la stânga, iar ultimele sale două rânduri sunt goale.

Problema Fibosnek

Fişier de intrare `fibosnek.in`
Fişier de ieşire `fibosnek.out`

Se consideră o matrice cu n linii și m coloane ce conține numere naturale nenule. Se definește o parcurgere ***snek*** a matricei un șir de valori obținut astfel: se parcurg elementele matricei coloană cu coloană, de la prima până la ultima, și, în cadrul fiecărei coloane, de sus în jos, de la elementul aflat pe prima linie, până la cel aflat pe ultima linie, ca în exemplu.

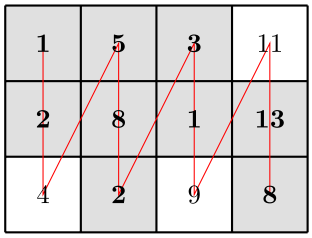

Figura 1: Exemplu de parcurgere *snek* a unei matrice cu 3 linii și 4 coloane.
Ordinea parcurgerii celulelor este: **1**, **2**, 4, **5**, **8**, **2**, **3**, **1**, 9, 11, **13**, **8**
Numerele Fibonacci au fost evidențiate.

Șirul numerelor Fibonacci este definit mai jos, unde `fib[k]` reprezintă al `k`-lea număr Fibonacci:

- `fib[1] = 1, fib[2] = 1`
- `fib[k] = fib[k - 1] + fib[k - 2]`, pentru orice `k` > 2

Se numește secvență ***fibosnek*** un termen sau o succesiune de termeni aflați pe poziții consecutive în parcurgerea *snek*, cu proprietatea că fiecare dintre ei este număr Fibonacci. Similar, se numește secvență ***non-fibosnek*** un termen sau o succesiune de termeni aflați pe poziții consecutive în parcurgerea *snek*, cu proprietatea că niciunul dintre ei nu este număr Fibonacci. Lungimea secvenței este egală cu numărul termenilor săi. Suma secvenței este egală cu suma termenilor săi.

O secvență *non-fibosnek* poate fi transformată în una *fibosnek* prin înlocuirea fiecărui număr din secvență cu un număr Fibonacci aflat cel mai aproape de el în șirul numerelor Fibonacci. Dacă există două numere Fibonacci la fel de apropiate de numărul dat, se va alege mereu cel mai mic. De exemplu, secvența (4) se transformă în secvența (3), iar secvența $(9, 11)$ în secvența $(8, 13)$.

Cerințe

Fiind date elementele matricei cu n linii și m coloane să se determine:

1. numărul de numere Fibonacci din matricea dată inițial;
2. suma celei mai lungi secvențe *fibosnek* ce poate fi obținută, știind că se poate transforma **cel mult o secvență** *non-fibosnek* în una *fibosnek* folosind procedeul explicat mai sus. Dacă se pot obține mai multe astfel de secvențe de lungime maximă, se va alege prima întâlnită în parcurgerea *snek* a matricei.

Date de intrare

Fișierul de intrare `fibosnek.in` conține pe prima linie numerele naturale c, n și m, unde c reprezintă cerința care trebuie rezolvată (1 sau 2), iar n și m au semnificația din enunț, pe următoarele n linii conține elementele matricei, parcurse în ordine, linie cu linie și în cadrul fiecărei linii, de la stânga la dreapta. Valorile aflate pe aceeași linie a fișierului sunt separate prin câte un spațiu.

Date de ieșire

Fișierul de ieșire `fibosnek.out` conține fie doar numărul determinat pentru cerința 1 (dacă $c = 1$), fie doar suma determinată pentru cerința 2 (dacă $c = 2$).

Restricții

- $c \in \{1, 2\}$
- $1 \leq n, m \leq 1\,500$
- Elementele matricei au valori în intervalul $[1, 2^{31} - 1]$.

#	Punctaj	Restricţii
1	21	$c = 1$ și $n, m \leq 1\,000$
2	20	$c = 2$ și $n, m \leq 100$
3	44	$c = 2$ și $n, m \leq 1\,000$
4	15	$c = 2$ și fără alte restricții suplimentare

Exemple

fibosnek.in	fibosnek.out
1 3 4 1 5 3 11 2 8 1 13 4 2 9 8	9
2 3 4 1 5 3 11 2 8 1 13 4 2 9 8	61
2 4 4 2 4 7 1 3 3 6 7 5 5 8 4 11 8 13 6	42

Explicații

Exemplul 1

$c = 1, n = 3, m = 4$, iar matricea corespunde celei din Fig. 1. Există 9 numere Fibonacci în matrice: $1, 5, 3, 2, 8, 1, 13, 2, 8$.

Exemplul 2

$c = 2, n = 3, m = 4$, iar matricea corespunde celei din Fig. 1. Dacă se transformă secvența *non-fibosnek* $(9, 11)$ în secvența *fibosnek* $(8, 13)$, atunci cea mai lungă secvență *fibosnek* este $(5, 8, 2, 3, 1, 8, 13, 13, 8)$, de lungime 9 și sumă 61.

Exemplul 3

Se transformă secvența *non-fibosnek* $(11, 4)$ în secvența *fibosnek* $(13, 3)$ și se obține secvența *fibosnek* $(2, 3, 5, 13, 3, 3, 5, 8)$ de lungime 8 și sumă 42. Deși mai există o secvență *fibosnek* de lungime 8 ce se poate obține prin transformarea secvenței *non-fibosnek* $(7, 6)$, aceasta nu a fost aleasă deoarece nu este prima secvență ce poate fi obținută.

Problema Partitură

Fişier de intrare `partitura.in`
Fişier de ieşire `partitura.out`

Mihai s-a decis în sfârșit să compună o melodie. Fără să știe de unde să înceapă, a scris pe o foaie n note muzicale. Fiecare notă muzicală este definită de două valori reprezentând durata și înălțimea acesteia astfel:

- **durata** este exprimată printr-o fracție de forma $\frac{1}{2^x}$, unde x este un număr natural nenul;
- **înălțimea** este exprimată printr-un număr natural nenul y.

Durata unui grup de note este egală cu suma duratelor notelor din grup. Pentru a compune o melodie corect din punct de vedere muzical, el trebuie să distribuie toate notele în grupuri disjuncte, astfel încât durata fiecărui grup să fie 1. Mihai definește **scorul unui grup** de note ca fiind suma înălțimilor tuturor notelor din grup, ridicată la pătrat. De asemenea, el definește **scorul unei melodii** ca fiind suma scorurilor tuturor grupurilor de note formate pentru acea melodie.

Mihai vrea să afle care este scorul maxim al unei melodii pe care îl poate obține după gruparea tuturor notelor date.

$$\frac{1}{2} + \frac{1}{2} = 1$$
$$\frac{1}{4} + \frac{1}{2} + \frac{1}{4} = 1$$
$$\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$$
$$\frac{1}{4} + \frac{1}{8} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} = 1$$

Figura 1: Exemple de durate ale unor note care pot forma un grup.

Cerință

Dându-se n note sub forma a n perechi de numere, x și y, să se afișeze scorul maxim ce poate fi obținut după gruparea tuturor notelor date în grupuri disjuncte.

Date de intrare

Fișierul de intrare `partitura.in` va conține pe prima linie un număr natural n, reprezentând numărul de note, iar pe următoarele n linii se vor afla câte două numere naturale x și y separate prin câte un spațiu, cu semnificația din enunț, pentru fiecare din cele n note.

Date de ieșire

Fișierul de ieșire `partitura.out` va conține un singur număr natural reprezentând scorul maxim cerut.

Restricții

- $1 \leq n \leq 300\,000$
- $1 \leq x \leq 18$
- $1 \leq y \leq 10\,000$
- Se garantează că se pot distribui toate notele date în grupuri de durată 1.

#	Punctaj	Restricţii
1	20	$n \leq 4$, $x = 1$
2	22	$x = 1$
3	17	Pentru toate notele, x are aceeași valoare
4	41	Fără restricții suplimentare

Exemple

partitura.in	partitura.out
5 2 3 3 2 2 1 2 2 3 5	169
6 1 3 2 2 1 4 2 2 2 2 2 2	113

Explicații

Exemplul 1

Pentru a determina scorul maxim al unei melodii, singura soluție posibilă se obține prin formarea unui singur grup. Acesta este format din toate notele și are durata $\frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^2} + \frac{1}{2^2} + \frac{1}{2^3} = 1$ și scorul $(3 + 2 + 1 + 2 + 5)^2 = 169$. Scorul melodiei este, de asemenea, 169.

Exemplul 2

Pentru a determina scorul maxim al unei melodii, o soluție posibilă se obține prin formarea a două grupuri. Primul grup este format din prima, a doua și a patra notă și are durata $\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^2} = 1$ și scorul $(3 + 2 + 2)^2 = 49$. În al doilea grup este format din a treia, a cincea și a șasea notă și are durata $\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^2} = 1$ și scorul $(4 + 2 + 2)^2 = 64$. Scorul melodiei este $49 + 64 = 113$ și este maximul care se poate obține pentru aceste note.