Mars

Hammaga ma'lumki kosmos bilan birinchi bo'lib fir'avnlar bog'langan. Ular birinchi kemalarini Tusmus I (hozirgi kunda Mars nomi bilan mashhur) sayyorasiga uchirishgan.Sayyora yuzasini $(2n+1) \times (2n+1)$ ta kvadrat katakchalardan tashkil topgan to'plam sifatida tasvirlash mumkin, bunda har bir katakchada suv yoki quruqlik mavjud. i – qator va j – ustunda joylashgan katakchaning qiymati ($0 \le i, j \le 2 \cdot n$) unda quruqlik bo'lganda 1 ga teng bo'ladi ($s[i][j] = 1$), aks holda, suv bo'lgan holatda 0 ga teng bo'ladi ($s[i][j] = 0$).

Agar ikkita katakcha orasida har birida bir xil tomonlari bo'lgan quruqliklardan tashkil topgan katakchalar katma-ketligi bo'lsa, ular bog'langan deb ataladi. Sayyorada orol deb har qanday ikkita katakchalari bog'langan maksimal quruqlik katakchalari to'plamiga aytiladi.

Samo kemasining vazifasi sayyoradagi orollar sonini sanash edi. Ammo kemaning eski kompyuteri sababli bu vazifa oson emasdi. Kompyuterning h xotirasi bo'lib, u ma'lumotlarni o'lchami $(2n+1) \times (2n+1)$ bo'lgan ikki o'lchovli massivda saqlaydi, undagi har bir katakchadagi ununligi 100 ga teng bo'lgan satr va bu satrning har bir belgisi '0' (ASCII 48) yoki '1' (ASCII 49) bo'lishi mumkin. Dastlab, har bir katakchaning birinchi biti katakcha holatini saqlaydi, $h[i][j][0] = s[i][j]$ (barcha $0 \le i, j \le 2 \cdot n$ lar uchun). Qolgan hamma bitlar '0' (ASCII 48) ga teng.

Xotirada saqlangan ma'lumotlarni qayta ishlash uchun kompyuter faqatgina xotiraning 3×3 qismiga kirishi va ushbu qismning yuqori chap katakchasidagi qiymatni qayta yozishi mumkin. Aniqroq aytganda, kompyuter $h[i..i+2][j..j+2]$ $(0 \le i, j \le 2 \cdot (n-1))$ katakchalarga kirishi va $h[i][j]$ dagi qiymati qayta yozishi mumkin. Bu jarayon keyinchalik **process cell** (i, j) deb ataladi.

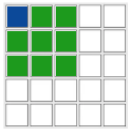
Kompyuterning cheklovlariga qarshi chiqib fir'avnlar quyidagi mexanizmni ishlab chiqdilar:

- Kompyuter xotirani n bosqichda qayta ishlaydi.
- k $(0 \le k \le n-1)$ - bosqichda quyidagicha amallar bajariladi: $m = 2 \cdot (n-k-1)$, bo'lsin, kompyuter $0 \le i, j \le m$ oralig'idagi barcha katakchalarni avval i ning o'sish tartibida, va har bir i uchun j ning o'sish tartibida qayta ishlaydi.Boshqacha qilib aytganda, kompyuter katakchalarni quyidagi tartibda qayta ishlaydi: $(0,0), (0,1), \cdots, (0,m)$, $(1,0), (1,1), \cdots, (1,m), \cdots, (m,0), (m,1), \cdots, (m,m)$.
- So'nggi bosqichda ($k = n-1$), kompyuter faqat $(0,0)$ katakchanigina qayta ishlaydi. Shu bosqichdan so'ng $h[0][0]$ – katakda yozilgan qiymat sayyoradagi orollar sonining ikkilik ko'rinishiga teng bo'lishi lozim teng bo'lishi lozim, bunda birinchi bit eng kichik qiymatga ega.

Quyidagi diagramma o'lchami 5×5 $(n = 2)$ xotirani kompyuter qanday qayta ishlashini ko'rsatib beradi. Moviy katakcha qayta yozilayotgan katakchani bildiradi, yashil rangli katakchalar esa qayta ishlanayotgan qism-massivlarni bilidiradi.

0 – bosqichda kompyuter qism-massivlarni quyidagi tartibda qayta ishlaydi:

Birinchi bosqichda kompyuter faqat bitta qism-massivnigini qayta ishlaydi:



Sizning vazifangiz Tutmus sayyorasidagi orollarni sonini ko’rsatilganidek amalga oshirishning yo’lini topishdir.

Bajarish tafsilotlari

Siz quyidagi protsedurani amalga oshirishingiz kerak:

```
string process(int64[][] a, int i, int j, int k, int n)
```

- a: o’chami 3×3 bo’lgan qayta ishlanayotgan qism-massiv, xususan $a = h[i..i+2][j..j+2]$. a ning har bir elementi satr bo'lib, uzunligi 100 ga teng va har bir belgisi '0' (ASCII 48) yoki '1' (ASCII 49) ga teng bo'lishi mumkin.
- i, j: ayni paytda qayta ishlanayotgan katakchaning qator va ustun raqami.
- k: hozirgi bosqich raqami.
- n: jami bosqichlar soni, va sayyora yuzi: $(2n+1) \times (2n+1)$
- Bu protsedura uzunligi 100 ga teng bo'lgan satr qaytarishi zarur. Qaytarilgan qiymat xotiraning at $h[i][j$ katakchasiga saqlanadi.
- Bu protsedurani so’nggi bor $k = n - 1$ bo’lganda chaqiriladi. Bu chaqirish mobaynida, protsedura sayyoradagi orollar sonining ikkilik ko'rinishini qaytarishi zarur. Bunda 0 - indeks eng kam qiymatli bitni, 1 - bit ikkinchi eng kam qiymatli bitni bildiradi.
- Bu protsedura har qanday statik va global o’zgaruvchilardan mustaqil bo’lishi, va qaytariladigan qiymat faqatgina unga o’tkaziladigan qiymatlarga bog’liq bo’lishi zarur.

Har bir testlar to’plami T ta mustaqil ssenariylardan iborat (ya’ni har bir sayyora turli xil). Sizning dasturningiz qo’llanishi har bir ssenariy uchun mustaqil bo’lishi zarur, chunki `process` protseduralari ketma-ket tartibda kelmasligi mumkin. Ammo har bir ssenariy uchun `process` lar chaqirilishi masala shartidagidek bo’lishi ta’minlangan.

Bundan tashqari, har bir test uchun dasturingizning bir nechta nusxalari bir vaqtning o'zida ishga tushiriladi. Xotira va CPU vaqt chegaralari bu barcha holatlar uchun birlashtiriladi. Ushbu holatlar o'rtasida ma'lumotlarni tarmoqdan tashqariga uzatishga qaratilgan har qanday urinish qoida buzarlik hisoblanadi va chetlashtirish uchun sabab bo'ladi.

Xususan, jarayon davomida saqlangan istalgan statik yoki global o’zgaruvchi keyingi protseduraga o’tishi ta’minlanmagan.

Chegaralar

- $1 \le T \le 10$
- $1 \le n \le 20$
- $0 \le s[i][j] \le 1$ ($0 \le i, j \le 2 \cdot n$ oralig'idagi har bir katakcha uchun)
- $h[i][j]$ ning uzunligi 100 ga teng ($0 \le i, j \le 2 \cdot n$ oralig'idagi har bir katakcha uchun)
- $h[i][j]$ ning har bir belgisi '0' (ASCII 48) yoki '1' (ASCII 49) bo'lishi mumkin ($0 \le i, j \le 2 \cdot n$ oralig'idagi har bir katakcha uchun)

`process` protsedurasiga beriladigan har bir so'ruv uchun:

- $0 \le k \le n - 1$
- $0 \le i, j \le 2 \cdot (n - k - 1)$

Qism-masalalar

Maslaning maxsus reyting sxemasi mavjud. Ba'zi qism-masalarni yechish boshqa qism-masalalar ballarini ham olib kelishi mumkin.

1. (6 ball) n ≤ 2
2. (8 ball) n ≤ 4
3. (7 ball) n ≤ 6
4. (8 ball) n ≤ 8
5. (7 ball) n ≤ 10
6. (8 ball) n ≤ 12
7. (10 ball) n ≤ 14
8. (24 ball) n ≤ 16
9. (11 ball) n ≤ 18
10. (11 ball) n ≤ 20

Misollar

Misol 1

$n = 1$ va s quyidagicha bo'lgan misolga nazar soling:

```
'1' '0' '0'
'1' '1' '0'
'0' '0' '1'
```

Bu misolda, sayyoraning yuzi 3×3 katakchalardan va 2 ta orollardan tashkil topgan. Bu holatda process protsedurasi faqatgina bir bosqichda chaqiriladi.

0 - bosqichda, grader `process` protsedurasini faqat bir marta chaqiradi:

```
process([["100","000","000"],["100","100","000"],["000","000","100"]],0,0,0,1)
```

E'tibor bering h ning har bir katakchasining faqatgina birinchi 3 ta biti ko'rsatilgan holos.

Bu protsedura `"0100..."`(qolgan barcha bitlar 0 ga teng) qiymatini qaytarishi lozim. Bunda ikkilikdagi ....0010 o'nlikdagi 2 ga teng. E'tibor bering 96 ta nollar tashlab ketildi va `...` bilan almashtirildi.

Misol 2

$n = 2$ va s quyidagicha bo'lgan misolga nazar soling:

```
'1' '1' '0' '1' '1'
'1' '1' '0' '0' '0'
'1' '0' '1' '1' '1'
'0' '1' '0' '0' '0'
'0' '1' '1' '1' '1'
```

Bu misolda, sayyoraning yuzi 5×5 katakchalar va 4 ta orollardan iborat. Bu holatda `process` protsedurasi ikki bosqichda chaqiriladi.

0 - bosqichda, grader `process` protsedurasini 9 marta chaqiradi:

```
process([["100","100","000"],["100","100","000"],["100","000","100"]],0,0,0,2)
process([["100","000","100"],["100","000","000"],["000","100","100"]],0,1,0,2)
process([["000","100","100"],["000","000","000"],["100","100","100"]],0,2,0,2)
process([["100","100","000"],["100","000","100"],["000","100","000"]],1,0,0,2)
process([["100","000","000"],["000","100","100"],["100","000","000"]],1,1,0,2)
process([["000","000","000"],["100","100","100"],["000","000","000"]],1,2,0,2)
process([["100","000","100"],["000","100","000"],["000","100","100"]],2,0,0,2)
process([["000","100","100"],["100","000","000"],["100","100","100"]],2,1,0,2)
process([["100","100","100"],["000","000","000"],["100","100","100"]],2,2,0,2)
```

Tasavvur qiling yuqoridagi so'rovlar quyidagi qiymatlarni qaytaradi `"011", "000", "000", "111", "111", "011", "110", "010", "111"`. Shunday qilib, 0 tugagach, h quyidagi qiymatlarni o'zida saqlaydi:

```
"011", "000", "000", "100", "100"
"111", "111", "011", "000", "000"
"110", "010", "111", "100", "100"
"000", "100", "000", "000", "000"
"000", "100", "100", "100", "100"
```

1 - bosqichda grader `process` protsedurasini faqat bir marta chaqiradi:

```
process([["011","000","000"],["111","111","011"],["110","010","111"]],0,0,1,2)
```

Va nihoyat funksiya `"0010000...."` (qolgan barcha bitlar 0 ga teng) qiymatini qaytarishi lozim. Bunda ikkilikdagi ....0000100 o'nlikdagi 4 ga teng. E'tibor bering 93 ta nollar tashlab ketildi va `...` bilan almashtirildi.

Sample grader

Sample grader kiruvchi ma'lumotlarni quyidagicha qabul qiladi:

- 1 - qator: T
- i - blok ($0 \leq i \leq T-1$): i - ssenariyni namoyon etib beruvchi to'plam .
  - 1 - qator: n
  - $2+j$ ($0 \leq j \leq 2 \cdot n$) qatorlar: $s[j][0]\ s[j][1]\ \ldots\ s[j][2 \cdot n]$

Sample grader javoblarni quyidagi formatda chiqaradi.

- $1+i$ ($0 \leq i \leq T-1$) qatorlarda: i - holat uchun `process` protsedurasi qaytargan so'nggi qiymat.