Alice, Bob, and Circuit

Cyberland Circuit Foundation-ը բաղկացած է n անդամներից: Յուրաքանչյուր անդամ ունի իր սիրելի համարը և եզակի անունը (սիրելի թվերը կարող են տարբեր չլինել):

m նամակներ են ուղարկվել անդամների միջև: Յուրաքանչյուր նամակ ունի ուղարկող և ստացող, և նամակի բովանդակությունը ուղարկողի սիրելի թիվն է:

Յուրաքանչյուր անդամ հաշվում է իր ստացած բովանդակության գումարը (ուղարկողների սիրելի համարները) և որպես արդյունք վերցնում է 65536-ի (այսինքն՝ 2^{16}) բաժանելուց մնացորդը:

Ձեր խնդիրն է պարզել բոլոր արդյունքները:

Այնուամենայնիվ, իրավիճակն այնքան էլ պարզ չէ, որքան թվում է: Alice-ը, Bob-ը և Circuit-ը որոշում են լուծել այս խնդիրը մի փոքր ավելի բարդ ձևով:

- Alice-ը գիտի բոլոր n անդամներին (նրանց անունները և սիրելի թվերը), սակայն տեղեկություն չունի նամակների մասին: Նա պետք է Circuit-ին ուղարկի երկուական տող, որի երկարությունը չի գերազանցում 10^5-ը:
- Bob-ը ունի տեղեկություն բոլոր m նամակների մասին (ուղարկողների և ստացողների անունները), բայց տեղեկություն չունի անդամների մասին: Նա պետք է Circuit-ին ուղարկի երկուական տող, որի երկարությունը չի գերազանցում 10^5-ը:
- Circuit-ը կարող է ստանալ Alice-ի և Bob-ի ուղարկած երկուական տողերը, և ելքում հաջորդաբար գեներացնի $16n$ բիթերից կազմված երկուական տողեր: Սակայն, իր հաշվողական հզորության սահմանափակության պատճառով, Circuit-ը կարող է կատարել միայն հասարակ տրամաբանական գործողություններ (օրինակ, AND, OR, NOT):

Հաջորդիվ կներկայացնենք circuit-ի աշխատանքի մանրամասները:

Circuit-ի մանրամասները

Գեյթը սխեմայի հիմնական տարրն է: Գեյթը բաղկացած է զրո կամ երկու բուլյան մուտքերից (կախված գեյթի տեսակից) և մեկ բուլյան ելքից: Գոյություն ունեն երկու տեսակի գեյթեր՝ մուտքային գեյթեր և հաշվողական գեյթեր:

- Մուտքային գեյթերը մուտք չունեն և ներկայացնում են Alice-ի և Bob-ի ուղարկած երկուական տողերի բիթերը:
  - Լինելու են $l_A + l_B$ հատ մուտքային գեյթեր, համարակալված 0-ից $(l_A + l_B - 1)$ թվերով, որտեղ $l_A-ն, l_B$-ն, համապատասխանաբար, Alice-ի և Bob-ի ուղարկած տողերի երկարություններն են:
  - $0 \le i < l_A$, համար i-րդ գեյթի ելքը Alice-ից ստացված տողի i-րդ բիթն է;
  - $0 \le i < l_B$ համար $(i + l_A)$-րդ գեյթի ելքը Bob-ից ստացված տողի i-րդ բիթն է:
- Հաշվողական գեյթերն ունեն երկու մուտք և ներկայացնում են հաշվողական պրոցեսը:
  - Հաշվողական գեյթերի համարները սկսում են $(l_A + l_B)$-ից:
  - Յուրաքանչյուր հաշվողական գեյթի համար դուք պետք է ներկայացնեք երկու անկախ գեյթերի համարներ որպես մուտք և գործողության $p (0 \le p \le 15)$ տեսակը:
    - Ցիկլիկ կախվածությունից խուսափելու համար երկու անկախ գեյթերի համարները պետք է փոքր լինեն հաշվողական գեյթի համարից:
    - Եթե երկու անկախ գեյթերի ելքերը, համապատասխանաբար, x_0 և x_1 են $(x_0, x_1 \in \{0, 1\})$, ապա հաշվողական գեյթի ելքը լինում է.

$$f(p, x_0, x_1) = \left\lfloor \frac{p}{2^{x_0 + 2x_1}} \right\rfloor \bmod 2$$

Ահա մի քանի օգտակար օրինակներ.

x_0	x_1	x_0 AND x_1 $f(8, x_0, x_1)$	x_0 OR x_1 $f(14, x_0, x_1)$	x_0 XOR x_1 $f(6, x_0, x_1)$	NOT x_0 $f(5, x_0, x_1)$
0	0	0	0	0	1
1	0	0	1	1	0
0	1	0	1	1	1
1	1	1	1	0	0

Իրականացման մանրամասներ

Նկատեք, խնդրեմ.

- Բոլոր զանգվածների ինդեքսները սկսվում են 0-ից: Օրինակ, եթե `a`-ն `n` երկարության զանգված է, ապա թույլատրելի արժեքներ են `a[0]`-ից մինչև `a[n-1]`, այս տիրույթից դուրս գալու դեպքում կարող է հանգեցնել սահմաններից-դուրս-գալու սխալի:
- Բոլոր տողերն ավարտվում են `\0` զրոյական սիմվոլով:

Դուք պետք է իրականացնեք հետևյալ ֆունկցիաները.

Alice

```
int alice(const int n, const char names[][5], const unsigned short numbers[], bool outputs_alice[]);
```

Direction	Value	Length	Meaning	Constraint
Input	n	1	n	$0 \le n \le 700$
	names	n	Յուրաքանչյուր անդամի անունը:	Բոլոր անուններն իրարից տարբեր են, բաղկացած են միայն անգլերեն փոքրատառերից և ունեն առավելագույնը 4 երկարություն:
	numbers	n	Յուրաքանչյուր անդամի սիրելի թիվը:	Բոլոր թվերը 0-ից 65535 սահմաններում են:
Output	outputs_alice	l_A	Circuit-ին ուղարկվող երկուական տողը:	
	(Return value)	1	l_A	**Պետք է համոզված լինեք, որ l_A-ն չի գերազանցում 10^5-ը և, երբ n-ը նույնն է, l_A-ն պետք է ֆիքսված լինի:**

Bob

```
int bob(const int m, const char senders[][5], const char recipients[][5], bool outputs_bob[]);
```

Direction	Value	Length	Meaning	Constraint
Input	m	1	m	$0 \le m \le 1000$
	senders	m	Ուղարկողի անունը յուրաքանչյուր նամակում:	Բոլոր անունները հանդիպում են Alice-ի մուտքային տվյալներում:
	recipients	m	Ստացողի անունը յուրաքանչյուր նամակում:	
Output	outputs_bob	l_B	Circuit-ին ուղարկվող երկուական տողը:	
	(Return value)	1	l_B	**Պետք է համոզված լինեք, որ l_B-ն չի գերազանցում 10^5 -ը, և երբ m-ը նույնն է, l_B-ն պետք է ֆիքսված լինի:**

Circuit

Ապահովելու համար, որ Circuit-ի հաշվարկման պրոցեսը նման լինի ընդհանրապես սխեմայի հաշվման պրոցեսի, դուք չպետք է ուղղակիորեն ստանաք Alice-ի և Bob-ի Circuit-ին ուղարկած երկուական տողերը: Դուք միայն գիտեք այդ երկու տողերի երկարությունները և ելքում տալիս եք սխեմայի կառուցվածքը:

```
int circuit(const int la, const int lb, int operations[], int operands[][2], int outputs_circuit[][16]);
```

Direction	Value	Length	Meaning	Constraint
Input	la	1	l_A	
	lb	1	l_B	
Output	operations	l	Սխեմայի յուրաքանչյուր գեյթի կողմից կիրառվող գործողության տեսակը։	0-ից 15 տիրույթին պատկանող ամբողջ թիվ։
	operands	l	Սխեմայի յուրաքանչյուր գեյթի կողմից օգտագործվող օպերանդը։	Այս թիվը պետք է փոքր լինի ընթացիկ գեյթից։
	outputs_circuit	n	Սխեմայի ելքի գեյթի համարը։	`outputs_circuit[i][j]` ցույց է տալիսi-րդ անդամի վերջնական արդյունքի j-րդ բիթը (համարակալումը սկսած ամենափոքր նշանակությամբ բիթից)։ Անդամները համարակալված են ըստ Alice-ի մուտքային տվյալների։
	(Return value)	1	l, որը ցույց է տալիս գեյթերի ընդհանուր քանակը (ներառյալ մուտքային գեյթերը)։	**Պետք է հազմոված լինեք, որ** $l \leq 2 \times 10^7$

Թեկուզև դուք կարող եք ձևափոխել $l_A + l_B$-ից փոքր համարներով գեյթերի ինֆորմացիան `operations` և `operands` զանգվածներում, գրեյդերն անտեսելու է նման փոփոխությունները։

Օրինակ

Դիտարկենք հետևյալ կանչերը.

```
alice(3, {"alic", "bob", "circ"}, {10000, 20000, 30000}, outputs_alice);
bob(5, {"alic", "bob", "bob", "circ", "circ"}, {"circ", "circ", "alic", "circ", "circ"}, outputs_bob);
```

Նրանք ներկայացնում են հետևյալ սցենարը.

- Alice-ը գիտի, որ կա 3 անդամ, `alic` անունով անդամի սիրելի թիվը 10000- ն է, և այլն։ Հնարավոր է, որ `alice()`-ի ելքը լինի այսպիսին.
  - `alice()`-ի վերադարձի արժեքը 2 է, այսինքն $l_A = 2$։
  - `alice()` ֆունկցիայի ներսում վերագրենք `outputs_alice[0] = 1, outputs_alice[1] = 0`, այսինքն արդյունարար երկուական տողը լինում է 10։
- Bob-ը գիտի, որ կա 5 նամակ, առաջին նամակը ուղղված է `alic`-ից `circ`-ին, և այլն։ Հնարավոր է, որ `bob()`-ի ելքը լինի այսպիսին.
  - `bob()`-ի վերադարձի արժեքը 3 է, նշանակում է $l_B = 3$.
  - `bob()` ֆունկցիայի ներսում վերագրենք `outputs_bob[0] = 1, outputs_bob[1] = 1, outputs_bob[2] = 0`, այսինքն արդյունարար երկուական տողը լինում է 110։

`alice()`-ի և `bob()`-ի ելքային տվյալների հիման վրա կարվի հետևյալ կանչը.

```
circuit(2, 3, operations, operands, outputs_circuit);
```

Այս ֆունկցիայի համար կոռեկտ ելքային տվյալները կլինեն այսպիսին.

- `circuit()`-ի վերադարձի արժեքըs 7 է, որը նշանակում է, որ մենք ավելացրել ենք երկու հաշվողական գեյթեր, որոնց համարներն են 5 և 6։
- `circuit()`-ի ներսում `operations`-ին, `operands`-ին և `outputs_circuit`-ին արժեքներ ենք տալիս հետևյալ կերպ.
  - `operations = {-1, -1, -1, -1, -1, 8, 14}`, որտեղ օգտագործում ենք `-1`-երը նշելու համար, որ անտեսում ենք մուտքային գեյթերի վերաբերյալ ինֆորմացիան։

- `operands = {{-1, -1}, {-1, -1}, {-1, -1}, {-1, -1}, {-1, -1}, {0, 4}, {2, 5}};`
- `outputs_circuit = {{5, 5, 5, 5, 5, 6, 5, 5, 5, 6, 6, 6, 5, 5, 6, 5}, ...}`: Այս զանգվածի բիթերը շատ են, ամբողջությամբ կարող եք տեսնել խնդրին կցված `abc.cpp` ֆայլում:

Ըստ ելքային տվյալների հաշվման գործընթացն այսպիսին է.

- Ավելացվում է 8 տեսակի հաշվողական գեյթ, որի մուտքը ստացվում է 0 և 4 գեյթերից: 0 գեյթի ելքը Alice-ից ստացված տողի 0-րդ բիթն է, որը 1 է: 4 գեյթի ելքը Bob-ից ստացված տողի 2-րդ բիթն է, որը 0 է: Հետևաբար, 5 գեյթի ելքը լինում է $f(8,0,1) = 0 \text{ AND } 1 = 0$:
- Ավելացվում է 14 տեսակի հաշվողական գեյթ, որի մուտքը ստացվում է 2 և 5 գեյթերից: 2 գեյթի երլքը Bob-ից ստացված մուտքային տողի 0-րդ բիթն է, որը 1 է: 5 գեյթի ելքը 0 է: Հետևաբար, 6 գեյթի ելքը լինում է $f(14,1,0) = 1 \text{ OR } 0 = 1$.
- `output_circuit[0]` -ը ներկայացնում է `alic`-ի արդյունքը, որն է $(0100111000100000)_2 = 20000$: Քանի որ `alic`-ը նամակ ստանում է միայն `bob`-ից, `alic`-ի վերջնական արդյունքը 20000 է:
- `bob`-ի վերջնական արդյունքը պետք է լինի 0, քանի որ նա ոչ մի նամակ չի ստանում: `circ`-ի վերջնական արդյունքը կլինի $(10000 + 20000 + 30000 + 30000) \bmod 65536 = 24464$:

Կցված `abc.cpp` ծրագիրն այս թեստը անց է կացնում, բայց մենք չենք երաշխավորում, որ այն կարող է բոլոր թեստերն անցկացնել:

Սահմանափակումներ

Բոլոր թեստերի համար

- $0 \le n \le 700, 0 \le m \le 1000$.
- Բոլոր անունները տարբեր են, բաղկացած են միայն անգլերեն փոքրատառերից, և նրանցից յուրաքանչյուրի երկարությունն առավելագույնը 4 է:
- Յուրաքանչյուր անդամի սիրելի թիվը պատկանում է 0-ից 65535 տիրույթին:
- Բոլոր ուղարկողների և ստացողների անունները հանդիպում են Alice-ի մուտքային `names` զանգվածում:
- `alice()`-ի և `bob()`-ի հիշողության սահմանափակումը 2048 MiB է, իսկ ժամանակի սահմանափակումը 0.02 վայրկյան:
- `circuit()`-ի հիշողության սահմանափակումը 2048 MiB է, իսկ ժամանակի սահմանափակումը՝ 7 վայրկյան:

Վերջնական գնահատման համար `alice()` և `bob()` ֆունկցիաները կարող են միևնույն թեստի համար կանչվել մի քանի անգամ: 0.02 ժամանակի սահմանափակումը վերաբերում է յուրաքանչյուր կանչին:

Ենթախնդիրներ

A տեսակի ենթախնդիրներ (12 միավոր)

1,2,3 ենթախնդիրներն A տեսակի են, որտեղ $n = 1$:

Յուրաքանչյուր ենթախնդիր բացի այդ ունի այսպիսի սահմանափակումներ.

- Ենթախնդիր 1 (4 միավոր) $m = 0$.
- Ենթախնդիր 2 (4 միավոր) $0 \le m \le 1$.
- Ենթախնդիր 3 (4 միավոր) $0 \le m \le 1000$.

B տեսակի ենթախնդիրներ (54 միավոր)

4,5,6 ենթախնդիրները B տեսակի են, որտեղ

- $0 \le n \le 30, \frac{n}{2} \le m \le n^2$
- Նույն ուղարկողով և ստացողով երկու նամակ չկա:
- Բոլոր անդամների անունները հանդիպում են Bob-ի մուտքային տվյալներում (այսինքն, յուրաքանչյուր անդամ կամ ուղարկում է մեկ նամակ, կամ ստանում է մեկ նամակ):

Ենթախնդիրներից յուրաքանչյուրն ունի լրացուցիչ այսպիսի սահմանափակումներ.

- Ենթախնդիր 4 (24 միավոր) $n = 26$, Բոլոր անդամների անունները մեկ տառանոց են և Alice-ի մուտքային տվյալներում նրանք հանդիպում են `a`-ից `z` հերթականությամբ:
- Ենթախնդիր 5 (24 միավոր) $n = 26$.
- Ենթախնդիր 6 (6 միավոր) Լրացուցիչ սահմանափակումներ չկան:

C տիպի ենթախնդիրներ (34 միավոր)

7,8,9 ենթախնդիրները C տիպի են, որտեղ $0 \le n \le 700, 0 \le m \le 1000$:

Ենթախնդիրներից յուրաքանչյուրն ունի այսպիսի լրացուցիչ սահմանափակումներ.

- Ենթախնդիր 7 (18 միավոր) $n = 676$, բոլոր անդամների անունները երկու տառանոց են և Alice-ի մուտքային տվյալներում նարնք հանդիպում են բառարանային կարգով (այսինքն, `aa`, `ab`, `ac`, ..., `az`, `ba`, ..., `bz`, `ca`, ..., `zz`):
- Ենթախնդիր 8 (10 միավոր) $n = 676$
- Ենթախնդիր 9 (6 միավոր) Լրացուցիչ սահմանափակումներ չկան:

Գրեյդերի նմուշ

Գրեյդերի նմուշը մուտքային տվյալները կարդում է հետևյալ ձևաչափով.

- Տող 1. $n\ m$
- Տող $2 + i (0 \le i \le n - 1)$. $names_i\ numbers_i$
- Տող $2 + n + i (0 \le i \le m - 1)$. $senders_i\ recipients_i$.

Գրեյդերի նմուշն արտածում է հետևյալ ձևաչափով.

- Եթե ծրագիրը հաջողությամբ է ավարտվում, գրեյդերի նմուշը արտածում է n տող, յուրաքանչյուրը պարունակում է մեկ ամբողջ թիվ, որը ցույց է տալիս յուրաքանչյուր անդամի համար ձեր իրրականացրած ֆունկցիաների հաշված վերջնական արդյունքը:
- Հակառակ դեպքում գրեյդերը ստանդարտ ելքում ոչինչ չի արտածում, այլ տպում է հաղորդագրություն սխալի մասին `abc.log` ֆայլում:
- Բացի այդ գրեյդերի նմուշը արտածում է l_A, l_B, l արժեքները և ֆունկցիաներից յուրաքանչյուրի աշխատելու ժամանակը `abc.log` ֆայլում:

Գրեյդերի նմուշը չի ստուգում հիշողության սահմանափակումը և այն պայմանը, որ միևնույն n / m համար l_A / l_B պետք է լինեն հվասար: