Message

Aisha și Basma sunt două prietene care corespondează. Aisha are un mesaj M, care este o secvență de S biți (i.e., valori de 0 și de 1), pe care ar vrea să îl trimită la Basma. Aisha comunică cu Basma trimițându-i **pachete**. Un pachet este o secvență de 31 de biți indexați de la 0 la 30. Aisha ar vrea să trimită mesajul M la Basma trimițându-i un număr de pachete.

Din păcate, Cleopatra a compromis comunicarea dintre Aisha and Basma și le poate **corupe** pachetele. Astfel, în fiecare pachet Cleopatra poate modifica biții de la exact 15 indici. Concret, se dă un șir C de lungime 31, în care fiecare element este 0 sau 1, cu semnificația următoare:

- $C[i] = 1$ semnifică faptul că bitul cu indicele i poate fi schimbat de Cleopatra. Numim acești indici **controlați** de Cleopatra.
- $C[i] = 0$ semnifică faptul că bitul cu indicele i nu poate fi schimbat de Cleopatra.

Șirul C conține exact 15 valori de unu și exact 16 zerouri. În timpul trimiterii mesajului M, mulțimea de indici controlați de Cleopatra rămâne același pentru toate pachetele. Aisha cunoaște cu precizie care sunt cei 15 indici controlați de Cleopatra. Basma știe doar că 15 indici sunt controlați de Cleopatra, dar nu știe care sunt aceștia.

Notăm cu A un pachet pe care Aisha decide să îl trimită (pe care îl vom numi **pachet original**). Notăm cu B un pachet pe care îl primește Basma (pe care îl vom numi **pachet corupt**). Pentru fiecare i, cu $0 \leq i < 31$:

- dacă Cleopatra nu controlează bitul cu indicele i ($C[i] = 0$), Basma primește bitul i cel trimis de Aisha ($B[i] = A[i]$),
- în caz contrar, dacă Cleopatra controlează bitul cu indicele i ($C[i] = 1$), atunci valoarea bitului $B[i]$ este hotărâtă de Cleopatra.

Imediat după trimiterea unui pachet, Aisha află care este pachetul corupt corespunzător.

După ce Aisha trimite toate pachetele, Basma primește toate pachetele corupte **în ordinea în care au fost trimise** și are de reconstruit mesajul original M.

Sarcina ta este să concepi și să implementezi o strategie care să permită lui Aisha să trimită mesajul M la Basma, și lui Basma să poată recupera M din pachetele corupte. În particular, trebuie să implementezi două funcții. Prima funcție implementează sarcinile lui Aisha. Ea primește mesajul M și șirul C, și trebuie să trimită niște pachete pentru a transfera mesajul către Basma. A

doua funcție implementează sarcinile lui Basma. Ea primește pachetele corupte și trebuie să reconstituie mesajul M.

Detalii de implementare

Prima funcție pe care trebuie să o implementezi este:

```
void send_message(std::vector<bool> M, std::vector<bool> C)
```

- M: un tablou de lungime S care reprezintă mesajul pe care Aisha dorește să îl trimită lui Basma.
- C: un tablou de lungime 31 specificând indicii controlați de Cleopatra.
- Această funcție poate fi apelată de **cel mult 2100 de ori** în fiecare test.

Această funcție trebuie să apeleze următoarea funcție pentru a trimite un pachet:

```
std::vector<bool> send_packet(std::vector<bool> A)
```

- A: un pachet original (un tablou de lungime 31) reprezentând biții trimiși de Aisha.
- Această funcție returnează un pachet corupt B reprezentând biții pe care îi va primi Basma.
- Această funcție poate fi apelată de cel mult 100 de ori la fiecare apel al lui `send_message`.

A doua funcție pe care trebuie să o implementezi este:

```
std::vector<bool> receive_message(std::vector<std::vector<bool>> R)
```

- R: un tablou care descrie pachetele corupte. Pachetele provin din cele trimise de Aisha într-un apel `send_message` și sunt date **în ordinea în care au fost trimise** de Aisha. Fiecare element al lui R este un șir de lungime 31, reprezentând un pachet corupt.
- Această funcție trebuie să returneze un șir de S biți care este egal cu mesajul original M.
- Această funcție trebuie apelată **de mai multe ori** în fiecare test, **exact odată** corespunzător fiecărui apel `send_message`. **Ordinea apelărilor funcției** `receive_message` nu este neapărat aceeași cu a apelările corespunzătoare ale funcției `send_message`.

Rețineți că în sistemul de evaluare funcțiile `send_message` și `receive_message` sunt apelate în **2 programe separate**.

Restricții

- $1 \leq S \leq 1024$
- C are exact 31 elemente, dintre care 16 sunt egale cu 0 și 15 sunt egale cu 1.

Subtaskuri și punctare

Dacă în vreunul dintre teste, apelul funcției `send_packet` nu respectă descrierea anterioară, sau dacă rezultatul pentru vreunul dintre apelurile `receive_message` este incorect, punctajul pentru acest test va fi 0.

În caz contrar, fie Q numărul maxim de apeluri ale funcției `send_packet` de-a lungul tuturor invocărilor `send_message` pentru toate testele. Fie, de asemenea X egal cu:

- 1, dacă $Q \leq 66$
- 0.95^{Q-66}, dacă $66 < Q \leq 100$

Atunci, punctajul se calculează astfel:

Subtask	Scor	Restricțiii
1	$10 \cdot X$	$S \leq 64$
2	$90 \cdot X$	Fără restricții suplimentare.

Rețineți că în anumite cazuri strategia graderului este **adaptivă**. Asta înseamnă că valorile returnate de funcția `send_packet` pot depinde nu doar de parametri acesteia, dar si de mulți alți factori, inclusiv datele de intrare si de ieșire ale apelurilor anterioare ale funcției si de numerele pseudo-aleatoare generate de grader. Graderul este **deterministic** în sensul în care dacă acesta este rulat de doua ori pe aceleași pachete, acesta o să facă aceleași schimbări.

Exemplu

Considerăm următorul apel.

```
send_message([0, 1, 1, 0],
             [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

Mesajul pe care Aisha încearcă să îl trimită la Basma este $[0, 1, 1, 0]$. Biții cu indici de la 0 la 15 nu pot fi schimbați de către Cleopatra, în timp ce biții cu indici de la 16 la 30 pot fi schimbați de către Cleopatra.

În cazul acestui exemplu, să presupunem că strategia Cleopatrei completează consecutiv și alternativ cu 0 și 1 biții pe care îi controlează, de exemplu poate asigna: 0 la primul indice controlat (16 în cazul nostru), 1 la al doilea indice controlat (17), 0 la al treilea indice controlat (18), etc.

Aisha poate decide să trimită doi biți din mesajul original într-un pachet, ca în continuare: ea va trimite primul bit la primii 8 indici pe care îi controlează și al doilea bit la următorii 8 indici pe care îi

controlează.

Atunci Aisha alege să trimită următorul pachet:

```
send_packet([0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
             0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

Rețineți că Cleopatra poate schimba biți cu de pe ultimii 15 indici, astfel că Aisha îi poate pune arbitrar, deoarece aceștia ar putea fi suprascriși. Cu strategia asumată de Cleopatra valoarea returnată ar fi: $[0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1]$.

Aisha decide să trimită ultimii doi biți ai lui M în pachetul al doilea într-un mod similar cu cel anterior:

```
send_packet([1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
             0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

Cu strategia asumată a Cleopatrei, funcția returnează: $[1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0]$.

Aisha poate trimite mai multe pachete dar decide să nu.

Grader-ul face atunci următorul apel de funcție:

```
receive_message([[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                  0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0],
                 [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
                  0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]])
```

Basma reconstruiește mesajul M ca mai departe. Pentru fiecare pachet ea alege primul bit care apare de două ori la rând și ultimul bit care apare de două ori la rând. Astfel, pentru primul pachet, ea alege biții $[0,1]$, și din al doilea pachet alege biții $[1,0]$. Punându-i împreună, ea reconstituie mesajul $[0,1,1,0]$, care este valoarea corectă returnată pentru acest apel `receive_message`.

Se poate arăta că cu strategia asumată a Cleopatrei și pentru mesaje de lungime 4, această abordare a lui Basma recuperează corect M, indiferent de valoarea C. Cu toate acestea, nu este corect în cazul general.

Grader local

Graderul local nu este adaptiv. În schimb, Cleopatra setează biți consecutivi pe care îi controlează cu biți alternanți 0 și 1, așa cum este descris în exemplul de mai sus.

Formatul intrării: **Prima linie a intrării conține un întreg T, reprezentând numărul de scenarii.** T scenarii urmează. Fiecare dintre ele este dat în următorul format:

```
S
M[0]  M[1]  ...  M[S-1]
C[0]  C[1]  ...  C[30]
```

Formatul ieșirii: Graderul local scrie rezultatele pentru fiecare dintre cele T scenarii în aceeași ordine în care au fost date la intrare și în formarul următor:

```
K L
D[0]  D[1]  ...  D[L-1]
```

Aici, K este numărul de apeluri ale funcției `send_packet`, D este mesajul returnat de `receive_message` și L este lungimea sa.