XOR Sort

Vi se dă un număr întreg S și un șir A de N numere întregi mai mari sau egale cu 0, indexat de la 1. Aveți voie să aplicați următoarea operație pe el: alegeți un indice i ($1 \leq i \leq N$), alegeți un vecin j ($1 \leq j \leq N$, fie $j = i - 1$ sau $j = i + 1$) și îl înlocuiți pe A_i cu ($A_i \oplus A_j$) unde $\oplus$ este XOR-ul pe biți al numerelor. Definim XOR-ul pe biți la finalul enunțului.
Scopul vostru este să îl sortați pe A:

- dacă $S = 1$ atunci se cere ca șirul final să fie crescător, i.e. $A_i < A_{i+1}$ pentru $1 \leq i < N$
- dacă $S = 2$ atunci se cere ca șirul final să fie nedescrescător, i.e. $A_i \leq A_{i+1}$ pentru $1 \leq i < N$

Găsiți orice secvență de operații care îndeplinește scopul.
Nu este neapărat să minimizați numărul de operații, cât timp numărul lor nu trece de 40000.

Input

Primul rând conține două valori întregi: N și S.
Al doilea rând conține N numere întregi: elementele lui A.

Output

Primul rând al outputului ar trebui să conțină un număr întreg K ($0 \leq K \leq 40000$) – numărul de operații.
Următoarele K linii ar trebui să conțină două numere întregi fiecare, descriind operațiile în ordine cronologică: primul număr este un indice i al elementului care se modifică, iar al doilea număr este un indice j al celuilalt număr implicat în operație.

Constraints

- $1 \leq S \leq 2$
- $2 \leq N \leq 1000$
- $0 \leq A_i < 2^{20}$

Subtasks

1. (25 puncte) 2 ≤ N ≤ 150, S=1, Elementele lui A sunt distincte
2. (35 puncte) 2 ≤ N ≤ 200, S=1, Elementele lui A sunt distincte
3. (40 puncte) 2 ≤ N ≤ 1000, S=2

Examples

Input	Output
5 1 3 2 8 4 1	3 1 2 4 3 5 4
5 2 4 4 2 0 1	3 3 2 4 3 5 4

Explicație la primul output:
[3, 2, 8, 4, 1] -> [**1**, 2, 8, 4, 1] -> [1, 2, 8, **12**, 1] -> [1, 2, 8, 12, **13**]

Explicație la al doilea output:
[4, 4, 2, 0, 1] -> [4, 4, **6**, 0, 1] -> [4, 4, 6, **6**, 1] -> [4, 4, 6, 6, **7**]

XOR-ul dintre biții a și b este 0 dacă a = b și 1 altfe.

XOR-ul pe biți între numerele întregi a și b este operația XOR făcută bit cu bit asupra biții lui a și b:

$75 \oplus 29 = 86$

$1001011 \oplus 0011101 = 1010110$

În C/C++/Java se folosește operatorul “^”pentru a face XOR.