

## 문제 1. 건물 4

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 2 seconds  
메모리 제한: 512 megabytes

JOI 왕국에서 올림픽 게임이 곧 열릴 것이다. 전 세계에 있는 참가자들을 환영하기 위해서 공항부터 숙박시설까지 길에 있는 건물들을 장식할 것이다. 총  $2N$  개의 건물이 있고, 1번 부터  $2N$  번 까지 번호가 붙어있다.

$K$  이사장은 건물 장식 계획을 맡았다. 그는 장식 계획의 공모를 받았고, 계획들을 관찰한 결과 계획 A와 계획 B의 두 계획을 골랐다. 계획 A에서는  $i$  번 ( $1 \leq i \leq 2N$ ) 건물의 아름다움이  $A_i$ 이다. 계획 B에서는  $i$  번 ( $1 \leq i \leq 2N$ ) 건물의 아름다움이  $B_i$ 이다.

두 계획은 모두 좋기 때문에, 둘 중 하나만 고르는 것은 어렵다. 이사장은 건물마다 계획 A와 계획 B 중 하나를 골라서 장식하기로 했다. 공평하게 장식하기 위해서  $N$  개의 건물에는 계획 A를, 나머지  $N$  개의 건물에는 계획 B를 고를 것이다. 게다가, 건물의 아름다움이 공항부터 숙박시설까지 가는 동안 올라가는 것이 참가자들에게 아름답게 보이기 때문에,  $i$  번 ( $1 \leq i \leq 2N$ ) 건물의 아름다움을  $C_i$ 라고 하면 모든  $1 \leq i \leq 2N - 1$  을 만족하는  $i$ 에 대해  $C_i \leq C_{i+1}$ 을 만족해야 한다.

건물의 수와 각 계획에 대해 건물의 아름다움이 정해졌을 때, 조건을 만족하도록 건물을 장식하는 것이 가능한지, 가능하다면 방법 하나를 출력하여라.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

$N$

$A_1 \dots A_{2N}$

$B_1 \dots B_{2N}$

### 출력 형식

조건을 만족하도록 건물을 장식하는 것이 불가능하다면, -1을 표준 출력으로 출력하여라.

가능하다면, 건물을 작성하는 길이  $2N$ 의 문자열  $S$ 를 출력하여라.  $i$  번째 ( $1 \leq i \leq 2N$ ) 문자는  $i$  번 빌딩을 계획 A로 장식한다면 A, 계획 B로 장식한다면 B이다. 답이 여럿 있으면, 아무거나 출력하여라.

### 제한

- $1 \leq N \leq 500\,000$ .
- $0 \leq A_i \leq 1\,000\,000\,000$  ( $1 \leq i \leq 2N$ ).
- $0 \leq B_i \leq 1\,000\,000\,000$  ( $1 \leq i \leq 2N$ ).

### 서브태스크 1 (11 점)

- $N \leq 2\,000$

### 서브태스크 2 (89 점)

추가 제한 조건이 없다.

### 예제

standard input	standard output
3 2 5 4 9 15 11 6 7 6 8 12 14	AABABB

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

---

각 건물에 대해 계획 A, A, B, A, B, B를 고른다. 이 경우 모든 계획은 세 번씩 골라졌다. 각 건물의 아름다움은 2, 5, 6, 9, 12, 14이므로 조건을 만족한다.

standard input	standard output
2 1 4 10 20 3 5 8 13	BBAA

건물을 장식하는 방법이 여럿 있으면, 아무거나 출력해도 좋다.

standard input	standard output
2 3 4 5 6 10 9 8 7	-1

조건을 만족하도록 장식하는 방법이 없기 때문에 -1을 출력하여라.

standard input	standard output
6 25 18 40 37 29 95 41 53 39 69 61 90 14 18 22 28 18 30 32 32 63 58 71 78	BABBABAABABA

## 문제 2. 함박 스테이크

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 3 seconds  
메모리 제한: 1024 megabytes

Just Odd Inventions라는 회사를 들어봤는가? 이 회사는 “그저 기묘한 발명”을 하는 회사이다. 이 문제에서는 줄여서 JOI 회사라고 부른다.

JOI 회사에서 신년 파티가 열렸다. 스태프는  $N$ 개의 함박 스테이크를 거대한 와이어 메쉬 (정사각형 격자가 뚫려있는 고기를 굽는 판)에 굽고 있다. 우리는 이 와이어 메쉬를  $1\ 000\ 000\ 000 \times 1\ 000\ 000\ 000$  격자라고 생각할 것이다. 왼쪽에서  $x$  번째 열, 아래에서  $y$  번째 행에 있는 격자칸을  $(x, y)$ 라고 표시할 것이다. ( $1 \leq x \leq 1\ 000\ 000\ 000$ ,  $1 \leq y \leq 1\ 000\ 000\ 000$ )

함박 스테이크는 1부터  $N$ 까지의 번호가 붙어있다.  $i$ 번 ( $1 \leq i \leq N$ ) 함박 스테이크는 왼쪽 아래가  $(L_i, D_i)$ 이고 오른쪽 위가  $(R_i, U_i)$ 인 직사각형 영역에 놓여있다. 함박 스테이크끼리 겹칠 수도 있다.

당신은 JOI 회사의 새 직원이다. 당신은 와이어 메쉬에서  $K$  개의 격자를 골라 격자 중앙에 꼬치를 꽂는 일을 해야 한다. 각 함박 스테이크에 대해, 하나 이상의 꼬치를 꽂는 것으로 함박 스테이크가 얼마나 익었는지 알 수 있다. 모든 함박 스테이크가 얼마나 익었는지 확인해야 한다. 꼬치로 하나 이상의 함박 스테이크를 꽂아도 되고, 함박 스테이크가 존재하지 않는 칸에 꼬치를 꽂아도 된다.

엄밀히 쓰면, 당신은 다음 조건을 만족하는 (서로 같을 수도 있는)  $K$  쌍의 정수  $(x_1, y_1), \dots, (x_K, y_K)$ 를 찾아야 한다:

- 모든  $i$  ( $1 \leq i \leq N$ )에 대해,  $L_i \leq x_j \leq R_i$  와  $D_i \leq y_j \leq U_i$ 를 만족하는  $j$  ( $1 \leq j \leq K$ ) 가 존재한다.
- 모든  $j$  ( $1 \leq j \leq K$ )에 대해,  $1 \leq x_j \leq 1\ 000\ 000\ 000$  과  $1 \leq y_j \leq 1\ 000\ 000\ 000$ 을 만족한다.

함박 스테이크의 위치와 꼬치의 개수가 주어졌을 때, 꼬치를 꽂는 방법을 아무거나 하나 출력하여라. 주어진 입력에서 위 조건을 만족하는 답이 항상 있음이 보장된다.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

$N\ K$

$L_1\ D_1\ R_1\ U_1$

$L_2\ D_2\ R_2\ U_2$

$\vdots$

$L_N\ D_N\ R_N\ U_N$

### 출력 형식

$K$ 개의 줄을 표준 출력으로 출력하여라.  $j$  번째 ( $1 \leq j \leq K$ ) 줄에는,  $x_j$ 와  $y_j$ 를 공백으로 구분하여 출력하여라.

답이 하나 이상 있으면, 아무거나 출력해도 좋다.

### 제한

- $1 \leq N \leq 200\ 000$ .
- $1 \leq K \leq 4$ .
- $1 \leq L_i \leq R_i \leq 1\ 000\ 000\ 000$  ( $1 \leq i \leq N$ ).
- $1 \leq D_i \leq U_i \leq 1\ 000\ 000\ 000$  ( $1 \leq i \leq N$ ).

- 문제의 조건을 만족하는  $K$  개의 격자 칸이 존재한다.

## 서브태스크 1 (1 점)

- $N \leq 2\,000$
- $K = 1$ .

## 서브태스크 2 (1 점)

- $N \leq 2\,000$
- $K = 2$ .

## 서브태스크 3 (3 점)

- $N \leq 2\,000$
- $K = 3$ .

## 서브태스크 4 (6 점)

- $N \leq 2\,000$
- $K = 4$ .

## 서브태스크 5 (1 점)

- $K = 1$ .

## 서브태스크 6 (3 점)

- $K = 2$ .

## 서브태스크 7 (6 점)

- $K = 3$ .

## 서브태스크 8 (79 점)

- $K = 4$ .

## 예제

standard input	standard output
4 2 2 1 3 3 1 2 4 3 6 1 7 4 5 3 7 5	2 2 7 4

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

---

(2, 2)에 꼬치를 꽂는 것으로, 1번과 2번 함박 스테이크가 얼마나 익었는지 확인 할 수 있고, (7, 4)에 꼬치를 꽂는 것으로 3번과 4번 함박 스테이크가 얼마나 익었는지 확인 할 수 있다.

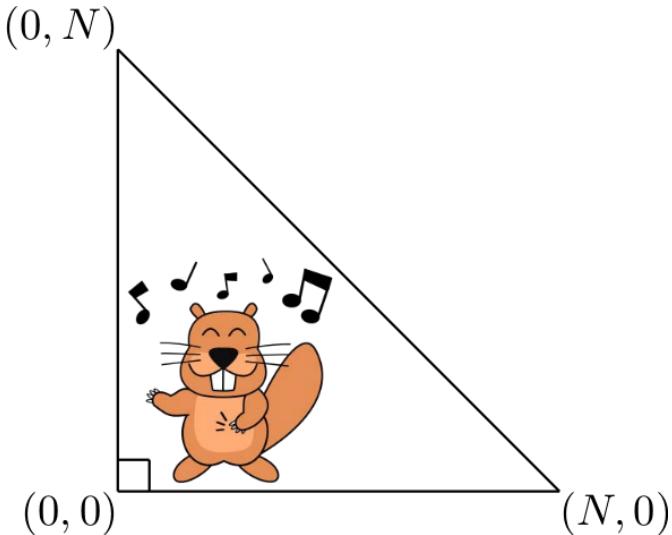
(2, 2)와 (7, 4) 이외에도 (3, 3) 과 (6, 4)에 꼬치를 꽂아도 확인할 수 있다.

standard input	standard output
3 3	1 1
1 1 1 1	1 2
1 2 1 2	1 3
1 3 1 3	

## 문제 3. 청소

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 18 seconds  
메모리 제한: 2048 megabytes

비타로의 방은 밑변의 길이가  $N$ 인 직각이등변삼각형 모양이다. 비타로의 방의 위치는  $0 \leq x \leq N$ ,  $0 \leq y \leq N$ ,  $x + y \leq N$ 을 만족하는  $(x, y)$ 로 표현된다. 직각인 꼭짓점이 원점이다. 두 밑변은  $x$ 축과  $y$ 축이다.



어느 날, 비타로는 방이 먼지로 가득 찬 것을 발견했다. 처음에 방에는  $M$ 개의 먼지가 있다.  $i$ 번 ( $1 \leq i \leq M$ ) 먼지는  $(X_i, Y_i)$ 에 놓여 있다. 둘 이상의 먼지가 같은 곳에 있을 수도 있다.

이제, 비타로는 빗자루를 이용하여 집을 청소하려고 한다. 우리는 빗자루를 방 안에 있는 선분이라고 생각한다. 이 선분의 길이를 **너비**라고 한다. 비타로는 다음 두 방법으로만 빗자루질한다.

- 비타로는 빗자루를 한 점을 원점에 놓고  $y$ 축과 평행하게 놓는다. 그리고 빗자루를  $x$ 축 양의 방향으로 움직이면서 빗자루의 한 점이  $x$ 축 위에 있고  $y$ 축과 평행하도록 움직일 수 있을 때까지 움직인다. 빗자루의 너비가  $l$ 이라면,  $x < N - l$ 과  $y \leq l$ 을 만족하는  $(x, y)$ 에 있는 먼지는  $(N - l, y)$ 로 움직일 것이다. ( $(N - l, y)$ 에 다른 먼지가 있을 수도 있다.) 이 방법을 **H 쓸기**라고 한다.
- 비타로는 빗자루를 한 점을 원점에 놓고  $x$ 축과 평행하게 놓는다. 그리고 빗자루를  $y$ 축 양의 방향으로 움직이면서 빗자루의 한 점이  $y$ 축 위에 있고  $x$ 축과 평행하도록 움직일 수 있을 때까지 움직인다. 빗자루의 너비가  $l$ 이라면,  $x \leq l$ 과  $y < N - l$ 을 만족하는  $(x, y)$ 에 있는 먼지는  $(x, N - l)$ 로 움직일 것이다. ( $(x, N - l)$ 에 다른 먼지가 있을 수도 있다.) 이 방법을 **V 쓸기**라고 한다.

비타로의 방에서  $Q$ 개의 일이 차례로 일어날 것이다.  $j$  번째 ( $1 \leq j \leq Q$ ) 일은 다음 중 하나이다.

- $P_j$  번째 먼지의 좌표를 계산한다.
- 너비  $L_j$ 인 빗자루를 사용하여, H 쓸기를 한다.
- 너비  $L_j$ 인 빗자루를 사용하여, V 쓸기를 한다.
- $(A_j, B_j)$ 에 새로운 먼지가 추가된다. 이 일 전에  $c$  개의 먼지가 있었다면, 이 먼지는  $(c + 1)$  번째 먼지이다.

방의 밑변의 길이, 방에 있는 먼지의 좌표와 일어난 일들이 주어졌을 때, 먼지의 좌표를 계산하여라.

## 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 수는 정수이다.

$N \ M \ Q$

$X_1 \ Y_1$

:

$X_M \ Y_M$

(Query 1)

:

(Query Q)

공백으로 구분 된 두 개 혹은 세 개의 수가 (Query  $j$ ) ( $1 \leq j \leq Q$ ) 로 주어진다.  $T_j$ 가 첫 번째 수라고 하자. 이 줄의 의미는 다음과 같다.

- $T_j = 1$ 이면, 두 정수  $T_j, P_j$ 가 공백으로 구분되어 주어진다. 이는  $j$  번째 일이 비타로가  $P_j$ 의 위치를 계산하는 것을 의미한다.
- $T_j = 2$ 이면, 두 정수  $T_j, L_j$ 가 공백으로 구분되어 주어진다. 이는  $j$  번째 일이 비타로가 너비  $L_j$ 인 빗자루를 사용하여 H 쓸기를 하는 것을 의미한다.
- $T_j = 3$ 이면, 두 정수  $T_j, L_j$ 가 공백으로 구분되어 주어진다. 이는  $j$  번째 일이 비타로가 너비  $L_j$ 인 빗자루를 사용하여 V 쓸기를 하는 것을 의미한다.
- $T_j = 4$ 이면, 세 정수  $T_j, A_j, B_j$ 가 공백으로 구분되어 주어진다. 이는  $j$  번째 일이 새 먼지가  $(A_j, B_j)$ 에 추가 된 것을 의미한다.

## 출력 형식

$T_j = 1$ 인 일이 생길 때마다, 하나의 줄을 표준 출력으로 출력하여라.  $j$  번째 일이 생길 때에  $P_j$ 의  $x$ 좌표와  $y$  좌표를 공백으로 구분하여 출력하여라.

## 제한

- $1 \leq N \leq 1\,000\,000\,000$ .
- $1 \leq M \leq 500\,000$ .
- $1 \leq Q \leq 1\,000\,000$ .
- $0 \leq X_i \leq N$  ( $1 \leq i \leq M$ ).
- $0 \leq Y_i \leq N$  ( $1 \leq i \leq M$ ).
- $X_i + Y_i + i \leq N$  ( $1 \leq i \leq M$ ).
- $1 \leq P_j \leq (j$  번째 일이 생길 때 먼지 개수) ( $1 \leq j \leq Q$ ).
- $0 \leq L_j \leq N - 1$  ( $1 \leq j \leq Q$ ).
- $0 \leq A_j \leq N$  ( $1 \leq j \leq Q$ ).
- $0 \leq B_j \leq N$  ( $1 \leq j \leq Q$ ).
- $A_j + B_j \leq N$  ( $1 \leq j \leq Q$ ).
- $T_j = 1$ 인 일이 적어도 하나 존재한다. ( $1 \leq j \leq Q$ ).

## 서브태스크 1 (1 점)

- $M \leq 2\,000$ .
- $Q \leq 5\,000$ .

## 서브태스크 2 (10 점)

- $T_j = 1, 2, 4$ .

## 서브태스크 3 (11 점)

- $T_j = 1, 2, 3$ .
- $X_j \leq X_{j+1}$  ( $1 \leq j \leq M - 1$ ).
- $Y_j \geq Y_{j+1}$  ( $1 \leq j \leq M - 1$ ).

## 서브태스크 4 (53 점)

- $T_j = 1, 2, 3$ .

## 서브태스크 5 (25 점)

추가 제한조건이 없다.

### 예제

standard input	standard output
6 2 10	1 3
1 1	3 2
4 0	3 3
4 2 3	6 0
3 3	
1 1	
4 1 2	
2 3	
2 0	
1 4	
3 2	
1 3	
1 2	

- 처음에 첫 번째 먼지가 (1, 1), 두 번째 먼지가 (4, 0)에 있다. (그림 1)이 방의 상태이다.
- 첫 번째 일에서 세 번째 먼지가 (2, 3)에 놓인다. (그림 2)가 방의 상태이다.
- 두 번째 일에서 너비 3의 빗자루로 V 쓸기를 한다. 그 후, 첫 번째 먼지가 (1, 3)으로 옮겨졌다. (그림 3)이 방의 상태이다.
- 세 번째 일에서 첫 번째 먼지의 좌표인 (1, 3)을 계산한다.
- 네 번째 일에서 네 번째 먼지가 (1, 2)에 놓인다. (그림 4)가 방의 상태이다.

- 다섯 번째 일에서 너비 3의 빗자루로 H 쓸기를 한다. 그 후, 첫 번째 먼지가  $(3, 3)$  으로, 세 번째 먼지가  $(3, 3)$ 으로, 네 번째 먼지가  $(3, 2)$ 로 옮겨졌다. (그림 5)가 방의 상태이다.
- 여섯 번째 일에서, 너비 0의 빗자루로 H 쓸기를 한다. 그 후, 두 번째 먼지가  $(6, 0)$  으로 옮겨졌다. (그림 6)이 방의 상태이다.
- 일곱 번째 일에서, 비타로는 네 번째 먼지의 좌표인  $(3, 2)$ 를 계산한다.
- 여덟 번째 일에서, 비타로는 너비 2의 빗자루로 V 쓸기를 한다. 어떤 먼지도 옮겨지지 않았다. (그림 7)이 방의 상태이다.
- 아홉 번째 일에서, 비타로는 세 번째 먼지의 좌표인  $(3, 3)$ 을 계산한다.
- 열 번째 일에서, 비타로는 두 번째 먼지의 좌표인  $(6, 0)$ 을 계산한다.

이 예제는 서브태스크 1과 5의 조건을 만족한다.

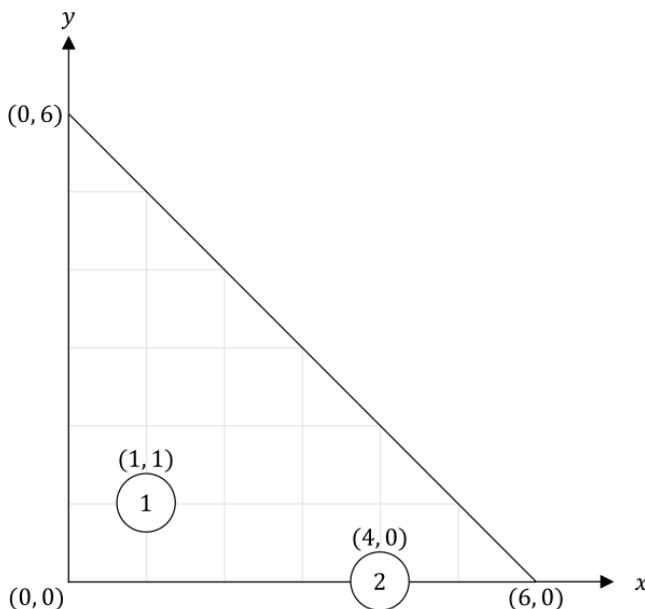


그림 1

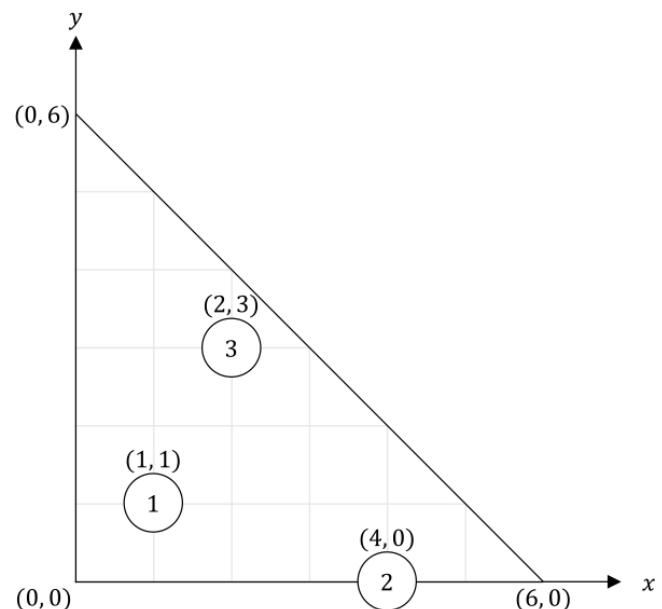


그림 2

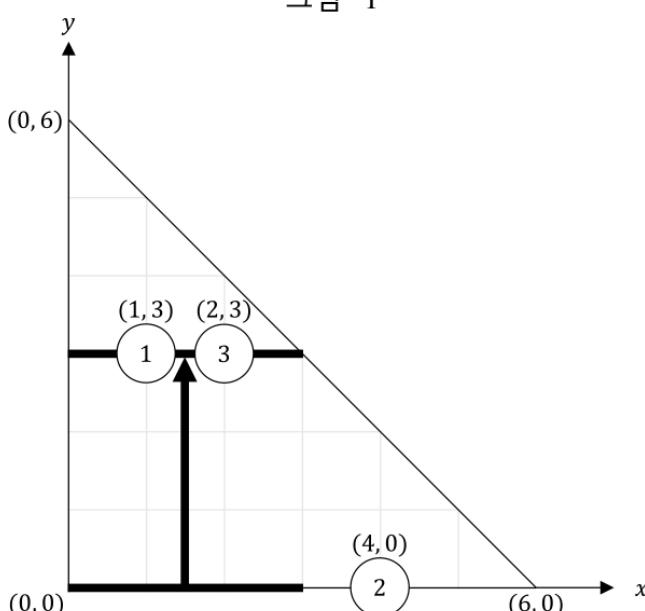


그림 3

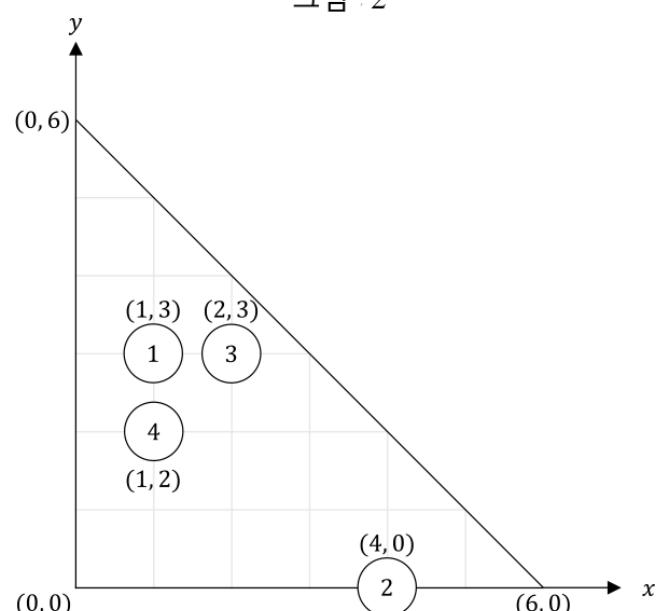


그림 4

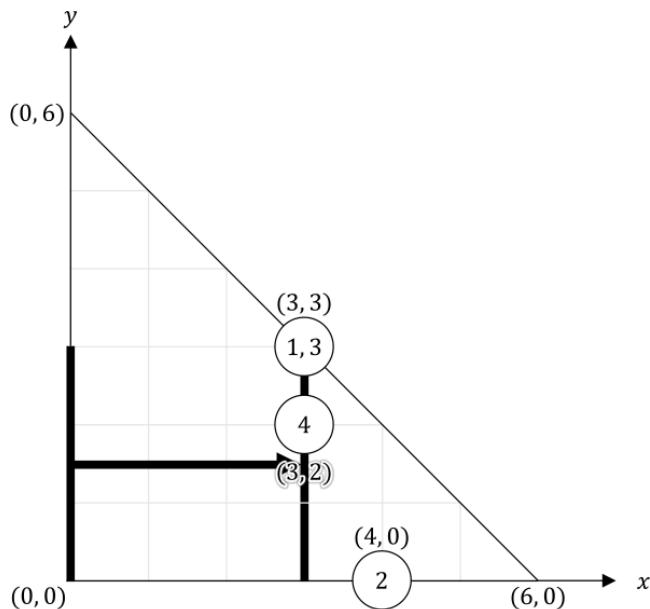


그림 5

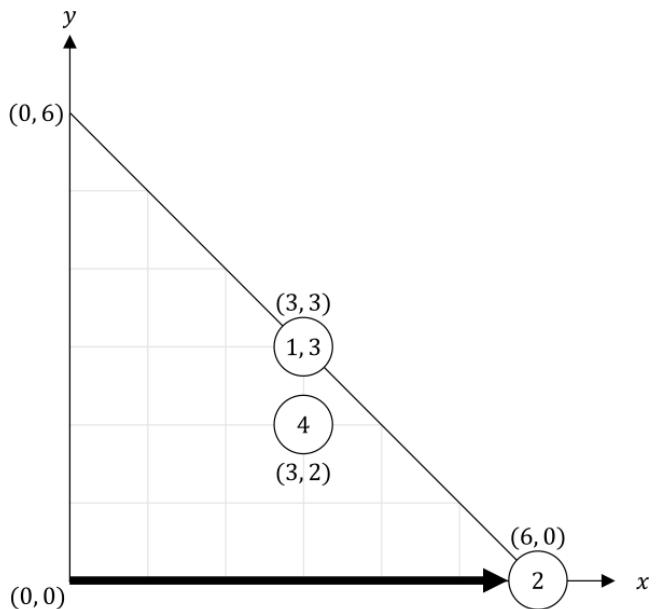


그림 6

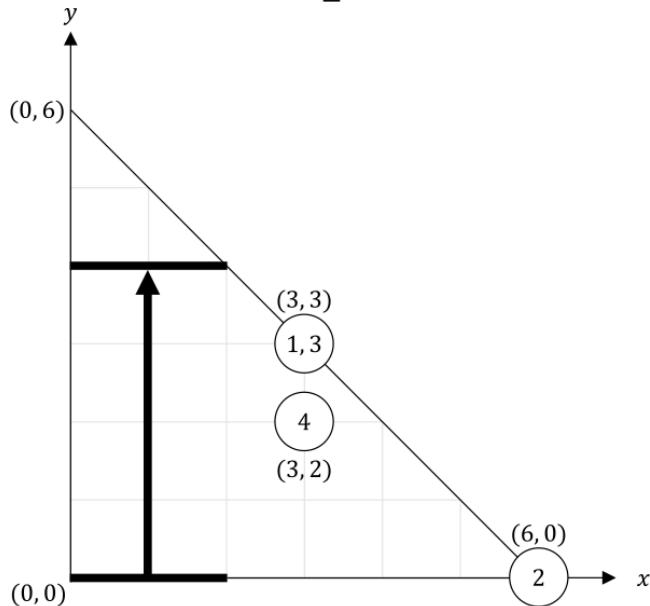


그림 7

standard input	standard output
9 4 8	3 6
2 3	4 3
3 1	7 1
1 6	6 3
4 3	
2 6	
1 3	
2 2	
1 4	
2 3	
1 2	
2 4	
1 1	

이 예제는 서브태스크 1, 2, 4와 5의 조건을 만족한다.

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

standard input	standard output
8 1 8	4 1
1 5	3 5
4 4 1	3 2
2 6	
1 2	
2 3	
4 2 2	
2 5	
1 1	
1 3	

이 예제는 서브태스크 1, 2와 5의 조건을 만족한다.

standard input	standard output
7 4 9	4 2
1 5	5 1
2 2	1 6
4 2	5 2
5 0	
2 6	
2 3	
1 2	
3 6	
1 4	
3 1	
1 1	
2 2	
1 3	

이 예제는 서브태스크 1, 3, 4와 5의 조건을 만족한다.

## 문제 4. 카멜레온의 사랑

시간 제한: 2 second  
메모리 제한: 512 megabytes

JOI 동물원에는  $2N$  마리의 카멜레온이 있고 1부터  $2N$ 까지의 번호가 붙어있다. 그 중,  $N$  마리의 카멜레온은 성별이 X이다. 다른  $N$  마리의 카멜레온은 성별이 Y이다.

각 카멜레온에는 원본 색이 있다. 카멜레온의 원본 색에 대해서는 다음과 같은 사실이 알려져 있다.

- 성별 X인 카멜레온  $N$  마리의 원본 색은 모두 다르다.
- 성별이 X인 각 카멜레온에 대해, 원본 색이 같은 성별이 Y인 카멜레온이 유일하게 존재한다.

지금 JOI 동물원에는 새로운 사랑이 짜트는 계절이다. 각 카멜레온은 다른 카멜레온을 사랑한다. 카멜레온의 사랑에 대해서는 다음과 같은 사실이 알려져 있다.

- 각 카멜레온은 성별이 자신과 다른 카멜레온을 정확히 한 마리 사랑한다.
- 어떤 카멜레온과 그 카멜레온이 사랑하는 카멜레온의 색은 다르다.
- 같은 카멜레온을 사랑하는 서로 다른 두 카멜레온은 존재하지 않는다.

당신은 카멜레온 몇 마리를 모아서 미팅을 주선하려고 한다. 미팅에 참석한 카멜레온  $s$ 에 대해,  $s$ 가 카멜레온  $t$ 를 사랑한다고 하자.  $s$ 의 피부색은 다음과 같이 정해진다.

- $t$ 가 미팅에 참석하면,  $s$ 의 피부색은  $t$ 의 원본 색이다.
- $t$ 가 미팅에 참석하지 않으면,  $s$ 의 피부색은  $s$ 의 원본 색이다.

카멜레온의 피부색은 어떤 미팅에 참석하느냐에 따라 바뀔 수 있다. 당신이 주선하는 미팅에 대해, 미팅에 참석한 카멜레온의 피부색이 총 몇 종류인지 알 수 있다.

당신은 미팅을 20 000번 이하로 주선해서 원본 색이 같은 카멜레온 쌍을 모두 알고 싶다.

카멜레온의 수가 주어질 때, 미팅을 20 000번 이하로 주선해서 원본 색이 같은 카멜레온 쌍을 모두 정하는 프로그램을 작성하여라.

### 구현 명세

당신은 파일 하나를 제출해야 한다.

이 파일의 이름은 `chameleon.cpp`이다. 파일은 다음 함수를 구현해야 한다. 또한, `chameleon.h`를 include 해야 한다.

- `void Solve(int N)`  
이 함수는 각 테스트 케이스마다 정확히 한 번 불린다.
  - 인자  $N$ 은 성별이 X인 카멜레온의 수  $N$ 을 나타낸다.

당신의 프로그램은 다음 함수를 호출 할 수 있다.

- `int Query(const std::vector<int> &p)`  
당신은 이 함수를 호출함으로써 미팅을 주선할 수 있다.
  - \* 인자  $p$ 는 미팅에 참여하는 카멜레온의 목록이다.
  - \* 이 함수의 반환값은 미팅에 참석한 카멜레온의 피부색이 몇 종류인지이다.
  - \*  $p$ 에 있는 각 원소는 1 이상  $2N$  이하의 정수여야 한다. 이를 만족하지 않은 경우에는 오답 [1] 이 된다.

- \*  $p$ 에 있는 각 원소는 서로 달라야 한다. 이를 만족하지 않은 경우에는 **오답 [2]**이 된다.
- \* 당신은 이 함수를 20 000번 이상 호출해서는 안된다. 이를 만족하지 않은 경우에는 **오답 [3]**이 된다.

– `void Answer(int a, int b)`

이 함수를 사용하여, 같은 원본 색을 가진 카멜레온의 쌍을 답할 수 있다.

- \* 인자  $a$ 와  $b$ 는  $a$  번째 카멜레온과  $b$  번째 카멜레온의 원본 색이 같다는 것을 의미한다.
- \*  $1 \leq a \leq 2N$ 과  $1 \leq b \leq 2N$ 을 만족해야 한다. 이를 만족하지 않을 경우 **오답 [4]**이 된다.
- \*  $a$ 와  $b$ 로 주어진 수는 2번 이상 나타나서는 안 된다. 이를 만족하지 않은 경우에는 **오답 [5]**이 된다.
- \* 서로 원본 색을 가진 카멜레온을 지정할 경우 **오답 [6]**이 된다.
- \* 함수 `Answer`는 정확히  $N$ 번 호출될 필요가 있다. 함수 `Solve`의 실행 종료 시에 함수 `Answer`의 호출 횟수가  $N$ 번이 아닐 경우 **오답 [7]**이 된다.

## 참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안 된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안 된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 `grader.cpp`이다. 당신의 프로그램을 테스트 하기 위해서, `grader.cpp`, `chameleon.cpp`, `chameleon.h`를 같은 디렉토리 안에 놓고, 컴파일 하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=gnu++14 -O2 -o grader grader.cpp chameleon.cpp`

컴파일이 성공적이면, 파일 `grader`가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여야라. 샘플 그레이더는 하나의 프로세스에서 실행되며, 입력을 표준 입력으로부터 받고, 출력을 표준 출력에 출력한다.

## 입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

$N$

$Y_1 \dots Y_{2N}$

$C_1 \dots C_{2N}$

$L_1 \dots L_{2N}$

$Y_i$ 는 ( $1 \leq i \leq 2N$ )  $i$  번째 카멜레온의 성별을 나타내고, 0 혹은 1이다. 0이면 성별이 X이고, 1이면 성별이 Y이다.

$C_i$ 는 ( $1 \leq i \leq 2N$ )  $i$  번째 카멜레온의 원본 색을 나타내고, 1 이상  $N$  이하의 정수이다.

$L_i$ 는 ( $1 \leq i \leq 2N$ )  $i$  번째 카멜레온이 사랑하는 카멜레온의 번호이다.

$X_i$ 와  $Y_i$  ( $0 \leq i \leq N - 2$ )는  $X_i$ 번 조각과  $Y_i$ 번 조각이 같은 종류의 광물임을 의미한다.

## 출력 형식

프로그램이 정상적으로 종료되었다면, 샘플 그레이더는 다음과 같은 정보를 표준 출력에 출력한다. (따옴표는 출력하지 않는다.)

- 정답으로 판단된 경우, Query함수의 호출 횟수를 “Accepted: 100”과 같은 형식으로 출력한다.
- 오답으로 판단된 경우, 오답의 종류를 “Wrong Answer [1]”과 같은 형식으로 출력한다.

프로그램이 다양한 오답의 종류에 속해 있으면, 샘플 그레이더는 그중 하나만 출력할 것이다.

## 제한

모든 입력 데이터는 다음의 조건을 만족한다.  $Y, C, L$ 의 의미는 입력 형식을 참고하여라.

- $2 \leq N \leq 500$ .
- $0 \leq Y_i \leq 1$  ( $1 \leq i \leq 2N$ ).
- 각  $j$  ( $1 \leq j \leq N$ ) 에 대해,  $Y_i = 0$ 과  $C_i = j$ 를 만족하는 유일한  $i$ 가 ( $1 \leq i \leq 2N$ ) 존재한다.
- 각  $j$  ( $1 \leq j \leq N$ ) 에 대해,  $Y_i = 1$ 과  $C_i = j$ 를 만족하는 유일한  $i$ 가 ( $1 \leq i \leq 2N$ ) 존재한다.
- $1 \leq L_i \leq 2N$  ( $1 \leq i \leq 2N$ ).
- $Y_i \leq Y_{L_i}$  ( $1 \leq i \leq 2N$ ).
- $C_i \leq C_{L_i}$  ( $1 \leq i \leq 2N$ ).
- $L_k \neq L_l$  ( $1 \leq k < l \leq 2N$ ).

## 서브태스크 1 (4 점)

- $L_{L_i} = i$ . ( $1 \leq i \leq 2N$ ).

## 서브태스크 2 (20 점)

- $N \leq 7$ .

## 서브태스크 3 (20 점)

- $N \leq 50$ .

## 서브태스크 4 (20 점)

- $Y_i = 0$  ( $1 \leq i \leq N$ ).

## 서브태스크 5 (36 점)

추가 제한조건이 없다.

## 예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

예제 입력	예제 함수 호출		
	호출	호출	반환값
4 1 0 1 0 0 1 1 0 4 4 1 2 1 2 3 3 4 3 8 7 6 5 2 1	Solve(4)		
		Query([])	0
		Query([6, 2])	2
		Query([8, 1, 6])	2
		Query([7, 1, 3, 5, 6, 8])	4
		Query([8, 6, 4, 1, 5])	3
		Answer(6, 4)	
		Answer(7, 8)	
		Answer(2, 1)	
		Answer(3, 5)	

대회 홈페이지의 아카이브에서 받을 수 있는 파일 중, sample-02.txt는 서브태스크 1의 조건을, sample-03.txt는 서브태스크 4의 조건을 만족한다.

## 문제 5. 조이터에서 친구를 만드는건 재밌어

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 3 seconds  
메모리 제한: 1024 megabytes

조이터는 당신의 추억을 친구와 남길 수 있는 소셜 미디어 서비스이다.

조이터에서는 당신은 다른 사용자를 팔로우할 수 있다. 예를 들어, 사용자  $a$ 가 사용자  $b$ 를 팔로우 하면, 사용자  $a$ 는 사용자  $b$ 의 게시글을 자신의 타임라인에서 읽을 수 있다. 이 경우에 사용자  $b$ 가 사용자  $a$ 를 팔로우 할 수도, 아닐 수도 있다. 하지만, 사용자  $a$ 가 특정 사용자  $b$ 를 한 번 넘게 팔로우 할 수는 없다.

1번, 2번, …,  $N$ 번 사용자 총  $N$ 명의 사용자가 조이터를 시작했다. 처음에, 아무도 서로를 팔로우하고 있지 않다.

이제부터 총  $M$ 일 동안  $i$ 번째 ( $1 \leq i \leq M$ ) 날에  $A_i$ 번 사용자가  $B_i$ 번 사용자를 팔로우한다.

조이터는 교류 이벤트를  $M$ 일 동안 열려고 한다. 교류 이벤트는 다음과 같은 방법으로 이루어진다.

1. 사용자 한 명을 고른다. 이 사용자를  $x$ 라고 하자.
2. 사용자  $x$ 가 현재 팔로우하고 있는 사용자 중 한 명을 고른다. 이 사용자를  $y$ 라고 하자.
3. 다음 조건을 만족하도록 사용자  $z$ 를 고른다:  $z$ 는  $x$ 와 다르며,  $x$ 가  $z$ 를 팔로우하고 있지 않고,  $y$ 가  $z$ 를 팔로우하고,  $z$ 가  $y$ 를 팔로우한다.
4.  $x$ 가  $z$ 를 팔로우한다.
5. 조건을 만족하는  $(x, y, z)$ 가 없을 때까지 위 과정을 반복한다.

교류 이벤트가 언제 진행될지는 결정되지 않았다. 그래서, 각  $M$ 일에 대해, 어떤 사용자가 다른 사용자를 팔로우한 이후 교류 이벤트가 개최되었을 때, 교류 이벤트가 종료된 시점에  $N$ 명이 팔로우하고 있는 사람 수의 총합의 최댓값을 구하고 싶다. 단, 교류 이벤트는 시작한 날에 종료되는 것으로 한다.

사용자의 수와  $M$ 일간의 팔로우 정보가 주어졌을 때, 각 일에 교류 이벤트가 열린 직후에  $N$ 명이 팔로우하는 사람 수의 총합의 최댓값을 구하는 프로그램을 작성하여라.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

$N\ M$

$A_1\ B_1$

:

$A_M\ B_M$

### 출력 형식

$M$ 개의 줄을 표준 출력으로 출력하여라.  $i$ 번째 ( $1 \leq i \leq M$ ) 줄에는,  $i$ 번째에 어떤 사용자가 다른 사용자를 팔로우한 이후 교류 이벤트가 개최되었을 때, 교류 이벤트가 종료된 시점에  $N$ 명이 팔로우하는 사람 수의 총합의 최댓값을 출력하여라.

### 제한

- $2 \leq N \leq 100\ 000$ .
- $1 \leq M \leq 300\ 000$ .
- $1 \leq A_i \leq N$  ( $1 \leq i \leq M$ ).

- $1 \leq B_i \leq N$  ( $1 \leq i \leq M$ ).
- $A_i \neq B_i$  ( $1 \leq i \leq M$ ).
- $(A_i, B_i) \neq (A_j, B_j)$  ( $1 \leq i < j \leq M$ ).

## 서브태스크 1 (1 점)

- $N \leq 50$ .

## 서브태스크 2 (16 점)

- $N \leq 2\,000$ .

## 서브태스크 3 (83 점)

추가 제한조건이 없다.

### 예제

standard input	standard output
4 6	1
1 2	2
2 3	4
3 2	4
1 3	5
3 4	9
4 3	

- 첫째 날에 1번 사용자가 2번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 1이다.
- 둘째 날에 2번 사용자가 3번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 2이다.
- 셋째 날에 3번 사용자가 2번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최 하면, 예를 들면 1번 사용자가 3번 사용자를 팔로우 할 수 있다. 이 때 팔로우 수 합계는 4가 되고, 이 값이 최댓값이다.
- 넷째 날에 1번 사용자가 3번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 4이다.
- 다섯째 날에 3번 사용자가 4번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 5이다.
- 여섯째 날에 4번 사용자가 3번 사용자를 팔로우한다. 이 후 교류 이벤트를 개최 하면, 예를 들면 1번 사용자가 3번 사용자를 팔로우하고, 2번 사용자가 4번 사용자를 팔로우하고, 4번 사용자가 2번 사용자를 팔로우한다. 이 때 팔로우 수 합계는 9가 되고, 이 값이 최댓값이다.

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

standard input	standard output
6 10	1
1 2	2
2 3	3
3 4	4
4 5	5
5 6	7
6 5	11
5 4	17
4 3	25
3 2	30
2 1	

## 문제 6. 유적 3

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 4 seconds  
메모리 제한: 512 megabytes

JOI 교수는 IOI 왕국 역사 연구의 일인자이다. IOI 왕국에 관련된 신사를 조사 중일 때 그는 건설된  $N$  개의 기둥을 발견했다. 또한, IOI 왕국의 고대인들이 쓴 것으로 추정되는 고문서를 발견했다. 이 문서는 기둥에 대한 정보가 쓰여 있었다. 문서에 쓰인 내용은 다음과 같다.

- 기둥 건설 직후에  $2N$ 개의 기둥이 있었고, 1번부터  $2N$ 번까지의 번호가 붙어있다.
- 기둥 건설 직후에 각  $k$ 에 ( $1 \leq k \leq N$ ) 대해, 높이가  $k$ 인 기둥이 정확히 두 개 있었다.
- 지진이  $N$ 번 일어났다. 지진 이후에 몇몇 기둥은 무너졌고 높이가 1 감소했다. 다른 기둥들은 고대인들이 보호했고 기둥이 무너지지 않았기 때문에 높이도 변하지 않았다.
- 지진이 일어났을 때 각  $k$ 에 ( $1 \leq k \leq N$ ) 대해, 높이가  $k$ 인 기둥이 정확히 하나가 보호되었다. 만약에 높이가  $k$ 인 기둥이 두 개 이상 있었을 경우, 가장 번호가 높은 기둥이 보호되었다. 다른 말로,  $i$ 번째 ( $1 \leq i \leq 2N$ ) 기둥의 높이가 지진 전에  $h_i$ 인 경우,  $i$  번째 기둥은  $h_i \geq 1$ 이고 모든  $j > i$ 에 대해,  $h_j \neq h_i$ 를 만족한 경우 보호되었다.
- $N$ 번의 지진이 일어난 이후  $N$ 개의 기둥이 남았다. (즉 기둥의 높이가 1 이상인 기둥이 정확히  $N$ 개 있다.)

$2N$ 개 기둥의 높이를 복원할 수 있다면 세기의 대발견이 될 수 있다고 생각한 JOI 교수는, 기둥을 좀 더 면밀히 조사했다. 그는  $N$ 번의 지진이 일어난 이후로 남은 기둥의 번호가  $A_1, A_2, \dots, A_N$ 이라는 것을 알아냈다.

JOI 교수는 기둥이 건설 되었을 때  $2N$ 개 기둥의 높이로 가능한 것이 몇 가지인지를 알고 싶다. 당신은 JOI 교수의 제자로, 이 수를 계산하는 프로그램을 작성하는 요구를 받았다.

$N$ 번의 지진 이후에 남은 기둥의 번호가 주어졌을 때,  $2N$ 개의 기둥의 높이로 가능한 가짓수를 1 000 000 007로 나눈 나머지를 구하여라.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 수는 정수이다.

$N$

$A_1 \dots A_N$

### 출력 형식

표준 출력에 하나의 줄을 출력하여라. 이 줄은 답을 1 000 000 007로 나눈 나머지이다.

### 제한

- $1 \leq N \leq 600$ .
- $1 \leq A_i \leq 2N$  ( $1 \leq i \leq N$ ).
- $A_i < A_{i+1}$  ( $1 \leq i \leq N - 1$ ).

### 서브태스크 1 (6 점)

- $N \leq 13$ .

## 서브태스크 2 (52 점)

- $N \leq 60$ .

## 서브태스크 3 (42 점)

추가 제한조건이 없다.

### 예제

standard input	standard output
3	5
3 4 6	

예를 들어, 기둥의 높이가  $(2, 2, 3, 3, 1, 1)$ 이었다고 하자. 각  $k$ 에 ( $1 \leq k \leq 3$ ) 대해, 높이가  $k$ 인 기둥이 정확히 두 개 존재하므로, 고문서에 쓰인 것과 일치한다.

- 첫 번째 지진 이후 2, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는  $(1, 2, 2, 3, 0, 1)$ 이 되었다.
- 두 번째 지진 이후 3, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는  $(0, 1, 2, 3, 0, 1)$ 이 되었다.
- 세 번째 지진 이후 3, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는  $(0, 0, 2, 3, 0, 1)$ 이 되었다.

세 지진 이후 3, 4, 6번 기둥이 남았고 입력에 주어진 정보와 같다. 추가로 기둥의 높이가  $(2, 3, 2, 3, 1, 1)$ ,  $(2, 3, 3, 2, 1, 1)$ ,  $(3, 2, 2, 3, 1, 1)$ ,  $(3, 2, 3, 2, 1, 1)$ 일 때도 답으로 가능하다.

그러므로, 총 5가지의 기둥 높이가 고문서에 쓰인 것과 입력으로 주어진 정보와 일치한다.

standard input	standard output
1	0
1	

이 예제에서,  $(1, 1)$ 이 고문서와 일치하는 유일한 높이이다. 첫 번째 진 이후 기둥의 높이는  $(0, 1)$ 이 된다.

그러므로, 고문서에 쓰인 것과 입력을 모두 만족시키는 높이는 존재하지 않는다.

standard input	standard output
10	147003663
5 8 9 13 15 16 17 18 19 20	

건설되었을 때 가능한  $2N$ 개의 기둥 높이로 111 147 004 440가지가 가능하다. 111 147 004 440을 1 000 000 007로 나눈 나머지는 147 003 663이다. 그러므로 147 003 663을 출력한다.

## 문제 7. 별자리 3

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 2 seconds  
메모리 제한: 512 megabytes

JOI군은 야경 사진을 찍었다. 이 사진은 가로  $N$ 개, 세로  $N$ 개의 픽셀로 되어있는 사진이다. 왼쪽에서  $x$  번째 열, 아래에서  $y$  번째 행 ( $1 \leq x \leq N, 1 \leq y \leq N$ )에 있는 픽셀을 픽셀  $(x, y)$ 라고 부른다.

이 사진의 각 픽셀은 빌딩, 밤하늘 혹은 별이다. 색은 각각 하얀색, 검은색, 노란색이다. 각  $i$ 에 ( $1 \leq i \leq N$ ) 대해  $i$ 번째 열에 있는 픽셀 중 아래에서  $A_i$ 행 까지는 하얀색 픽셀이다. 별이 찍힌 노란색 픽셀은  $M$ 개가 있고, 그 중  $j$  번째 ( $1 \leq j \leq M$ ) 픽셀은 픽셀  $(X_j, Y_j)$ 이다. 이 이외의 픽셀은 모두 밤하늘을 찍은 검은색 픽셀이다.

이 사진의 어느 직사각형 영역이 다음 두 조건을 만족한다면 별자리를 찍은 것이 된다.

- 직사각형 영역 내에 하얀색 픽셀이 존재하지 않는다.
- 직사각형 영역 내에 노란색 픽셀이 두 개 이상 존재한다.

JOI군은 별자리를 보는 것이 지쳤다. 몇몇 노란색 픽셀을 검은색으로 칠하는 것으로 어떤 직사각형 영역도 별자리를 찍은 것이 되지 않도록 하고 싶다. 하지만 너무 많은 노란색 픽셀을 없애 버린 경우 사진의 부자연스러움이 올라간다. 구체적으로는  $j$  번째 ( $1 \leq j \leq M$ ) 노란색 픽셀을 검은색으로 만들면 사진의 부자연스러움이  $C_j$  증가한다. 처음 사진의 부자연스러움은 0이다.

사진의 정보와 각 노란색 픽셀을 없앴을 때 증가하는 부자연스러움이 주어졌을 때, 어떤 직사각형 영역도 별자리를 찍은 것이 되지 않도록 하면서 노란색 픽셀을 지운 사진의 부자연스러움의 최솟값을 구하는 프로그램을 작성하여라.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

$N$   
 $A_1 \dots A_N$   
 $M$   
 $X_1 \ Y_1 \ C_1$   
 $\vdots$   
 $X_M \ Y_M \ C_M$

### 출력 형식

어떤 직사각형 영역도 별자리를 찍은 것이 되지 않도록 하면서 노란색 픽셀을 지울 때, 사진의 부자연스러움의 최솟값을 표준 출력으로 첫째 줄에 출력하여라.

### 제한

- $1 \leq N \leq 200\,000$ .
- $1 \leq A_i \leq N$  ( $1 \leq i \leq N$ ).
- $1 \leq M \leq 200\,000$ .
- $1 \leq X_j \leq N$  ( $1 \leq j \leq M$ ).
- $1 \leq Y_j \leq N$  ( $1 \leq j \leq M$ ).
- $1 \leq C_j \leq 1\,000\,000\,000$  ( $1 \leq j \leq M$ ).

- $A_{X_j} < Y_j$  ( $1 \leq j \leq M$ ).
- $(X_j, Y_j) \neq (X_k, Y_k)$  ( $1 \leq j < k \leq M$ ).

## 서브태스크 1 (14 점)

- $N \leq 300$ .
- $M \leq 300$ .

## 서브태스크 2 (21 점)

- $N \leq 2\,000$ .
- $M \leq 2\,000$ .

## 서브태스크 3 (65 점)

추가 제한조건이 없다.

### 예제

standard input	standard output
5 1 3 4 2 3 3 1 5 3 4 3 2 2 4 2	2

이 입력 예제에서, 픽셀 (1, 5)을 왼쪽 아래의 정점, 픽셀 (2, 4)를 오른쪽 아래의 정점으로 하는 직사각형 영역은 별자리를 찍은 것이다. 세 번째 노란색 픽셀을 검은색으로 만들면 사진의 부자연스러움은 2 증가하고 어떤 직사각형 영역도 별자리를 찍은 것이 되지 않는다. 이것이 최솟값이므로 2를 출력한다.

이 입력 예제의 사진은 (그림 1)에 대응된다.

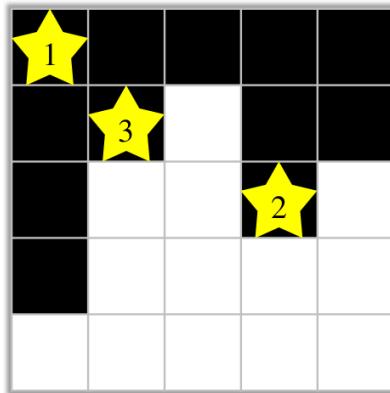


그림 1

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

standard input	standard output
7 5 6 2 3 6 7 6 5 7 7 5 3 3 7 3 7 10 1 7 6 4 7 8	16

이 입력 예제에서, 세 번째와 네 번째 노란색 픽셀을 검은색으로 만들면 된다.

standard input	standard output
8 6 8 5 7 3 4 2 1 10 8 2 9 6 6 7 8 3 18 5 8 17 8 5 3 5 5 3 5 4 8 1 8 13 1 7 5 7 4 13	44

## 문제 8. 수확

입력 파일: standard input  
출력 파일: standard output  
시간 제한: 3 seconds  
메모리 제한: 512 megabytes

IOI 농장은 사과를 기르는 거대한 농장이다. 이 농장은 큰 원 모양의 호수 둘레에 위치한 것으로 유명하다.

IOI 농장에는 1번부터  $N$ 번까지 번호가 붙어있는  $N$ 명의 직원이 있다. 또한, IOI 농장에는  $M$ 개의 사과나무가 심겨 있고, 1번부터  $M$ 번까지 번호가 붙어 있다. 호수의 둘레는  $L$  미터이다.

처음에,  $i$ 번 ( $1 \leq i \leq N$ ) 직원은 호수의 가장 북쪽에 위치한 점으로부터 시계방향으로  $A_i$  미터 떨어진 곳에서 기다리고 있다.  $A_i$ 는 ( $1 \leq i \leq N$ ) 서로 다르다.  $j$ 번 ( $1 \leq j \leq M$ ) 사과나무는 호수의 가장 북쪽에 위치한 점으로부터 시계방향으로  $B_j$  미터 떨어진 곳에 심겨 있다.  $B_j$ 는 ( $1 \leq j \leq M$ ) 모두 다르다. 게다가, 어떤 사과나무도 직원이 있는 곳에 심겨 있지 않다.

IOI 농장에 심어진 사과를 품종개량을 하기 위한 결과, 하나의 나무에는 한 번에 하나의 사과만 열린다. 그리고 사과가 수확되었을 때 새로운 사과는 정확히  $C$ 초 이후에 생긴다. 시각 0 초에 모든 사과나무에는 사과가 열려 있고 모든 직원이 시계방향으로 걷기 시작한다. 모든 직원의 속도는 초당 1 미터이다. 만약 직원이 사과나무에 도착했을 때 사과나무에 사과가 열려 있으면 직원은 그 사과를 항상 수확할 것이다(만약 사과가 열리는 시간과 동시에 직원이 사과나무에 도착한다면 직원은 그 사과도 항상 수확한다). 우리는 사과를 수확하는 데 걸리는 시간을 무시할 것이다.

K 이사장은 IOI 농장의 주식 지분을 가지고 있다. 당신이 IOI 농장의 관리인이기 때문에 K 이사장은 직원의 효율성에 대해서 보고하기를 바란다. 정확히는 K 이사장은 다음  $Q$ 개의 값을 알고 싶다.

각  $k$ 에 ( $1 \leq k \leq Q$ ) 대해,  $V_k$ 번 직원이 시각  $T_k$  초까지 수확한 사과의 개수 (시각  $T_k$ 에 수확한 사과가 존재한다면, 이도 포함한다.)

직원의 수, 사과나무의 수, 호수의 둘레, 새로운 사과가 열리는 데 필요한 시간, 직원의 위치, 사과나무의 위치와  $Q$ 개의 쿼리에 대한 정보가 주어졌을 때, 각 쿼리에 대해 수확한 사과의 개수를 출력하여라.

### 입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 수는 정수이다.

$N \ M \ L \ C$

$A_1 \dots A_N$

$B_1 \dots B_M$

$Q$

$V_1 \ T_1$

$\dots$

$V_Q \ T_Q$

### 출력 형식

표준 출력에  $Q$ 개의 줄을 출력하여라.  $k$  번째 ( $1 \leq k \leq Q$ ) 줄은,  $k$  번째 쿼리에 대한 답이다.

### 제한

- $1 \leq N \leq 200\,000$ .
- $1 \leq M \leq 200\,000$ .
- $N + M \leq L \leq 1\,000\,000\,000$ .

- $1 \leq C \leq 1\,000\,000\,000$ .
- $0 \leq A_i < L$  ( $1 \leq i \leq N$ ).
- $A_i < A_{i+1}$  ( $1 \leq i \leq N - 1$ ).
- $0 \leq B_j < L$  ( $1 \leq j \leq M$ ).
- $B_j < B_{j+1}$  ( $1 \leq j \leq M - 1$ ).
- $A_i \neq B_j$  ( $1 \leq i \leq N$ ,  $1 \leq j \leq M$ ).
- $1 \leq Q \leq 200\,000$ .
- $1 \leq V_k \leq N$  ( $1 \leq k \leq Q$ ).
- $1 \leq T_k \leq 1\,000\,000\,000\,000\,000\,000 = 10^{18}$  ( $1 \leq k \leq Q$ ).

## 서브태스크 1 (5 점)

- $N \leq 3\,000$ .
- $M \leq 3\,000$ .
- $Q \leq 3\,000$ .

## 서브태스크 2 (20 점)

- $T_k \geq 1\,000\,000\,000\,000\,000 = 10^{15}$  ( $1 \leq k \leq Q$ ).

## 서브태스크 3 (75 점)

추가 제한조건이 없다.

### 예제

standard input	standard output
3 2 7 3	2
1 4 6	1
0 5	1
3	
1 7	
2 3	
3 8	

- 시각 1 초에, 2번 직원은 2번 사과나무에서, 3번 직원은 1번 사과나무에서 사과를 수확한다.
- 시각 3 초에, 2번 직원은 1번 사과나무에 도착한다. 그때 사과가 열리지 않았으므로, 직원은 사과를 수확하지 않는다.
- 시각 4 초에, 1번 직원은 2번 사과나무에서 사과를 수확한다.
- 시각 6 초에, 1번 직원은 1번 사과나무에서 사과를 수확한다. 3번 직원은 2번 사과나무에 도착하지만, 그 때 사과가 열리지 않았으므로, 직원은 사과를 수확하지 않는다.
- 시각 8 초에, 2번 직원은 2번 사과나무에서 사과를 수확한다. 3번 직원은 1번 사과나무에 도착하지만, 그 때 사과가 열리지 않았으므로, 직원은 사과를 수확하지 않는다.

제 19회 일본 정보올림피아드 (JOI 2019/2020)  
여름 캠프 / 선발 고사, 2019년 3월 19–23일, (도쿄 코마바, 요요기)

시각 7 초까지 1번 직원이 수화한 사과의 개수가 2개이므로, 첫째 줄에는 2를 출력한다.

standard input	standard output
5 3 20 6	146
0 4 8 12 16	7035
2 11 14	7
9	7359360
4 1932	202
2 93787	10320
1 89	0
5 98124798	628
1 2684	18
1 137598	
3 2	
3 8375	
4 237	
8 15 217 33608	33230868503053
0 12 71 96 111 128 152 206	3
4 34 42 67 76 81 85 104 110 117 122	5
148 166 170 212	1
14	123542793648997
2 223544052420046341	8
3 86357593875941375	165811220737767
4 892813012303440034	8
1 517156961659770735	7
7 415536186438473633	1
6 322175014520330760	1
7 557706040951533058	7
6 640041274241532527	7535161012043
5 286263974600593111	132506837660717
8 349405886653104871	
1 987277313830536091	
5 989137777159975413	
2 50689028127994215	
7 445686748471896881	

## 문제 9. 길고양이

시간 제한: 2 seconds  
메모리 제한: 512 megabytes

Anthony는 JOI 도시에 사는 개미이다. JOI 도시에는 0번 부터  $N - 1$ 번까지 번호가 붙어 있는  $N$ 개의 마을이 있다. Anthony는 0번 마을에 살고 있다. 또한, 0번 부터  $M - 1$ 번까지 번호가 붙어 있는  $M$ 개의 도로가 있다.  $i$ 번 ( $0 \leq i \leq M - 1$ ) 도로는 마을  $U_i$ 와  $V_i$ 를 양방향으로 잇고 있다. 여러 개의 도로가 같은 쌍의 마을을 연결하는 경우는 없다. 어떤 마을에서도 다른 모든 마을까지 몇 개의 도로를 거치면 이동하는 것이 가능하다.

Anthony의 친구인 고양이 Catherine이 JOI 도시에 놀러 올 예정이다. Catherine은 도로의 정보를 모르기 때문에 길을 잃는 일이 잦았다. Anthony는 도로에 표식을 표시하기로 생각했다. 표식에는 0번부터  $A - 1$ 번까지 번호가 붙어 있는  $A$  종류의 표식이 있다.

이제, Catherine은 JOI 도시의 한 마을에 도착했다. Catherine은 0번 마을을 제외한 마을에 있을 경우,

(존재한다면) 직전에 지났던 도로를 제외하고, 지금 있는 마을에 인접한 도로 중 어떤 표식이 몇 개 있는가

를 셀 수 있고, 다음의 어떤 도로로 가는지 고르는 것이 가능하다. **직전에 지났던 도로를 제외하고는, 도로를 표식의 종류로만 구분할 수 있다.** 도로를 올바르게 고르는 것으로, 0번 마을까지 시간을 너무 많이 소비하지 않고 가고 싶다. 구체적으로는, 0번 마을까지 가는데 사용하는 도로의 최소 개수가  $d$  개인 경우, 0번 마을까지 최대  $d + B$  개만 사용하여 도착하고 싶다.

도로의 정보가 주어졌을 때, 도로에 표식을 표시하는 Anthony의 전략을 구현한 프로그램과 도로의 정보를 받아서 길을 걷는 Catherine의 전략을 구현한 프로그램을 작성하여라.

### 구현 명세

당신은 파일 두 개를 제출해야 한다.

첫째 파일의 이름은 `Anthony.cpp`이다. 이 파일은 Anthony의 일을 나타내고, 다음 함수를 구현해야 한다. 또한, `Anthony.h`를 include해야 한다.

- `std::vector<int> Mark(int N, int M, int A, int B,  
                          std::vector<int> U, std::vector<int> V)`

이 함수는 최초에 정확히 한 번 불린다.

- 인자  $N$ 은 마을의 수  $N$ 을 나타낸다.
- 인자  $M$ 은 도로의 수  $A$ 를 나타낸다.
- 인자  $A$ 는 표식의 개수  $A$ 를 나타낸다.
- 인자  $B$ 는 길을 걷는 횟수의 여유  $B$ 를 나타낸다.
- 인자  $U$ ,  $V$ 는 길이  $M$ 의 배열이다.  $U[i]$ 와  $V[i]$ 는  $i$ 번 도로가 연결하는 두 마을의 번호  $U_i$ ,  $V_i$ 를 의미한다. ( $0 \leq i \leq M - 1$ )
- 반환값  $x$ 는 길이  $M$ 의 배열이어야 한다. 길이가  $M$ 이 아닐 경우 오답 [1]이 된다.  $x[i]$ 는 ( $0 \leq i \leq M - 1$ ) 도로  $i$ 에 붙어 있는 표식의 번호를 의미한다.  $0 \leq x[i] \leq A - 1$ 을 만족해야 한다.  $0 \leq x[i] \leq A - 1$ 이 아닐 경우 오답 [2]이 된다.

둘째 파일의 이름은 `Catherine.cpp`이다. 이 파일은 Catherine의 일을 나타내고, 다음 함수를 구현해야 한다. 또한, `Catherine.h`를 include해야 한다.

- `void Init(int A, int B)`

이 함수는 최초에 정확히 한 번 불린다.

- 인자  $A$ 는 표식의 개수  $A$ 를 나타낸다.

– 인자  $B$ 는 길을 걷는 횟수의 여유  $B$ 를 나타낸다.

- `int Move(std::vector<int> y)`

이 함수는 Catherine이 0번 마을 이외의 마을을 방문할 때마다 불린다.

- 인자  $y$ 는 배열이고, (존재한다면) 직전에 지났던 도로를 제외하고, 지금 있는 마을에 인접한 도로 중  $j$  번째 ( $0 \leq j \leq A - 1$ ) 표식이  $y[j]$ 개 있다는 것을 의미한다.
- 반환값  $z$ 는  $-1 \leq z \leq A - 1$ 을 만족해야 한다.  $-1 \leq z \leq A - 1$ 을 만족하지 않을 경우 **오답 [3]**이 된다.  $z = -1$ 인 경우 직전에 왔던 도로로 돌아간다는 것을 의미하고,  $0 \leq z \leq A - 1$ 인 경우 직전에 지났던 도로 이외의  $z$ 번 표식이 붙어있는 도로를 고른다는 것을 의미한다. 함수 `Move`가 처음으로 호출 될 때,  $z = -1$ 인 경우 **오답 [4]**이 된다.  $0 \leq z \leq A - 1$ 이고  $y[z] = 0$ 인 경우 **오답 [5]**이 된다.

Catherine은 어떤 마을에서 직전에 지났던 도로 이외의 길을 고르는 경우 도로는 `Move`의 반환값에 해당하는 표식이 있는 도로 중 하나를 따라간다. 이 선택이 무작위가 아닐 수 있음에 유의하여라.

Catherine이 0번 마을에  $d + B$ 개의 도로를 사용한 이후 (즉, 함수 `Move`가  $d + B$  번 호출 된 이후)에 도착하지 못할 경우, **오답 [6]**이 된다.

## 참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다. 제출된 파일들은 같이 컴파일되어 하나의 실행 파일이 된다. 모든 글로벌 변수나 함수는 충돌을 피하기 위해 이름이 없는 namespace에 구현되어야 한다. 채점 될 때는, Anthony와 Catherine에 해당하는 두 프로세스로 나누어서 실행될 것이다. Anthony의 프로세스와 Catherine의 프로세스는 전역변수를 공유할 수 없다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안 된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안 된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 `grader.cpp`이다. 당신의 프로그램을 테스트하기 위해서, `grader.cpp`, `Anthony.cpp`, `Catherine.cpp`, `Anthony.h`, `Catherine.h`를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=gnu++14 -O2 -o grader grader.cpp Anthony.cpp Catherine.cpp`

컴파일이 성공적이면, 파일 `grader`가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여라. 샘플 그레이더는 하나의 프로세스에서 실행되며, 입력을 표준 입력으로부터 받고, 출력을 표준 출력에 출력한다.

## 입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

$N \ M \ A \ B \ S$

$U_0 \ V_0$

:

$U_{M-1} \ V_{M-1}$

여기서  $S$ 는 Catherine이 처음으로 도착하는 말의 번호이다.

## 출력 형식

프로그램이 정상적으로 종료되었다면, 샘플 그레이더는 다음과 같은 정보를 표준 출력 및 표준 에러에 출력한다. (따옴표는 출력하지 않는다.)

- 오답 [1], [2], [3], [4] 혹은 [5]로 판단 된 경우, 오답의 종류를 “Wrong Answer [1]”과 같은 형식으로 출력한다.
- Catherine이 0번 마을에  $N + B$ 번 이동 이내에 도착하지 못한 경우, “Wrong Answer; Number of moves > N+B”를 출력한다.
- 아닌 경우, 이동 횟수(Move의 호출 횟수)를 “Number of moves = 4”와 같은 형식으로 출력한다. 샘플 그레이더는 오답 [6] 인지 아닌지를 검사하지 않는다.

프로그램이 다양한 오답의 종류에 속해 있으면, 샘플 그레이더는 그중 하나만 출력할 것이다.

샘플 그레이더에서, Catherine이 자기가 직전에 지났던 도로 이외의 도로를 선택 할 때, 선택한 도로는 Move의 반환값에 해당하는 표식이 있는 도로 중 하나가 동일한 확률로 선택되며, 이는 주어진 시드에 따른 유사난수생성기에 의해 결정된다. 다른 시드로 프로그램을 실행하고 싶은 경우, 그레이더를 다음과 같은 형식으로 실행하여라:

```
./grader 2020
```

여기서, 첫 번째 인자는 유사난수생성기의 시드로 사용되는 정수이다.

## 제한

- $2 \leq N \leq 20\,000$ .
- $1 \leq M \leq 20\,000$ .
- $1 \leq S \leq N - 1$ .
- $0 \leq U_i < V_i \leq N - 1$  ( $0 \leq i \leq M - 1$ ).
- $(U_i, V_i) \neq (U_j, V_j)$  ( $0 \leq i < j \leq M - 1$ ).
- 어떤 마을에서도 다른 모든 마을까지 몇 개의 도로를 거치면 이동하는 것이 가능하다.

## 서브태스크 1 (2 점)

- $A = 4$ .
- $B = 0$ .
- $M = N - 1$ .

## 서브태스크 2 (2 점)

- $A = 4$ .
- $B = 0$ .

## 서브태스크 3 (2 점)

- $A = 3$ .
- $B = 0$ .
- $M = N - 1$ .

## 서브태스크 4 (9 점)

- $A = 3$ .
- $B = 0$ .

## 서브태스크 5 (5 점)

- $A = 2$ .
- $B = 2N$ .
- $M = N - 1$ .
- $6 \leq N \leq 500$ .

## 서브태스크 6 (71 점)

- $A = 2$ .
- $B = 12$ .
- $M = N - 1$ .

## 서브태스크 5 (9 점)

- $A = 2$ .
- $B = 6$ .
- $M = N - 1$ .

### 예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력
7 6 2 6 1
0 2
0 4
1 2
1 3
1 5
4 6

Anthony			
호출	반환값	호출	반환값
Mark(7,6,2,6,[0,0,1,1,1,4],[2,4,2,3,5,6])	[1,0,0,1,0,1]		
	Init(2, 6)		
	Move([2, 1])	0	
	Move([0, 0])	-1	
	Move([1, 1])	0	
	Move([0, 1])	1	

이 예제 입력에서, Catherine은 1, 5, 1, 2, 0 순으로 마을을 방문한다.  $d = 2$ 이고, Catherine은 4번 움직였다.

이 입력은 서브태스크 7의 조건을 만족한다.

대회 홈페이지의 아카이브에서 받을 수 있는 파일 중, sample-02.txt는 서브태스크 4의 조건을 만족한다.