문제 1. 고속도로 건설

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	1초
메모리 제한:	256MB

JOI 왕국에는 1번부터 N번까지 번호가 붙은 N 개의 도시가 있다. 1번 도시는 수도이다. 각 도시에는 **활기**라고 부르는 값이 있어서, i번 ($1 \leq i \leq N$) 도시의 활기의 초깃값은 C_i이다.

JOI 왕국의 도로는 두 개의 서로 다른 도시를 양방향으로 잇는다. JOI 왕국은 처음에는 도로가 없었다. JOI 왕국은 이제부터 $N-1$ 개의 도로를 건설하려고 한다. j 번째 ($1 \leq j \leq N-1$) 도로 건설은 다음과 같이 진행된다.

- A_j번 도시와 B_j번 도시를 지정한다. 미리 건설된 도로를 사용해서 1번 도시에서 A_j번 도시로 갈 수는 있지만, 1번 도시에서 B_j번 도시로 갈 수는 없다.
- A_j번 도시와 B_j번 도시를 잇는 도로를 짓는다. 건설 비용은 다음 조건을 만족하는 (s, t) 쌍의 개수이다.

  > s번 도시와 t번 도시는 1번 도시와 A_j번 도시를 잇는 최단 경로상에 있고, 1번 도시에서 A_j번 도시로 갈 때 s번 도시를 t번 도시보다 먼저 방문하며, s번 도시의 활기는 t번 도시의 활기보다 크다.

  여기서, 1번 도시와 A_j번 도시를 잇는 경로는 1번 도시와 A_j번 도시를 포함한다. 1번 도시와 A_j번 도시를 잇는 최단 경로는 유일함에 유의하여라.
- 1번 도시와 A_j번 도시를 잇는 최단 경로상에 있는 모든 도시의 활기를 B_j번 도시의 활기로 바꾼다.

각각 도로 건설에 걸리는 비용을 알고 싶다.

도시와 도로의 건설 계획에 대한 정보가 주어졌을 때, 각각의 도로 건설에 드는 비용을 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 정수 N이 주어진다. 이는 JOI 왕국에 N 개의 도시가 있다는 것을 의미한다.
- 둘째 줄에는 N 개의 정수 C_1, C_2, $\cdots$, C_N이 공백으로 구분되어 주어진다. 이는 i번 ($1 \leq i \leq N$) 도시의 활기의 초깃값이 C_i라는 것을 의미한다.
- 다음 $N-1$ 개의 줄의 j 번째 ($1 \leq j \leq N-1$) 줄에는 두 개의 정수 A_j, B_j 가 공백으로 구분되어 주어진다. 이는 j 번째 도로 건설에 A_j번 도시와 B_j번 도시가 지정됨을 의미한다.

출력 형식

표준 출력으로 $N-1$ 개의 줄을 출력한다. $N-1$ 개의 줄의 j 번째 ($1 \leq j \leq N-1$) 줄에는 j 번째 도로 건설에 드는 비용을 출력하여라.

제한

- $1 \leq N \leq 100\ 000$.
- $1 \leq C_i \leq 1\ 000\ 000\ 000$ ($1 \leq i \leq N$).
- $1 \leq A_j \leq N$ ($1 \leq j \leq N-1$).

- $1 \leq B_j \leq N$ $(1 \leq j \leq N-1)$.
- j 번째 도로 건설보다 전에 지어진 도로를 사용해서 1번 도시에서 A_j번 도시로 갈 수는 있지만, 1번 도시에서 B_j번 도시로 갈 수는 없다. $(1 \leq j \leq N-1)$

서브태스크 1 (7 점)

- $N \leq 500$.

서브태스크 2 (9 점)

- $N \leq 4\ 000$.

서브태스크 3 (84 점)

추가 제한조건이 없다.

예제

standard input	standard output
5 1 2 3 4 5 1 2 2 3 2 4 3 5	0 0 0 2

이 입력 예제에서는, 다음과 같이 도로 건설이 진행된다.

- 첫 번째 도로 건설을 만족하는 s번 도시와 t번 도시는 없기 때문에 건설 비용은 0이다. 1번 도시와 2번 도시를 잇는 도로를 지은 후에 1번 도시의 활기를 2로 바꾼다.
- 두 번째 도로 건설도 건설 비용은 0이다. 2번 도시와 3번 도시를 잇는 도로를 지은 후에 1번 도시와 2번 도시의 활기를 3으로 바꾼다.
- 세 번째 도로 건설도 건설 비용은 0이다. 2번 도시와 4번 도시를 잇는 도로를 지은 후에 1번 도시와 2번 도시의 활기를 4로 바꾼다.
- 네 번째 도로 건설은 $(s,t) = (1,3),(2,3)$이 조건을 만족하므로 건설 비용은 2이다. 3번 도시와 5번 도시를 잇는 도로를 지은 후에 1번 도시와 2번 도시와 3번 도시의 활기를 5로 바꾼다.

standard input	standard output
10 1 7 3 4 8 6 2 9 10 5 1 2 1 3 2 4 3 5 2 6 3 7 4 8 5 9 6 10	0 0 0 1 1 0 1 2 3

문제 2. 울타리

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	1초
메모리 제한:	256MB

JOI 씨는 IOI 나라의 거대한 토지를 소유하고 있다. IOI 나라의 토지는 좌표평면으로 표현되며, 직교하는 X 축과 Y축이 정해져 있다. X 좌표가 x, Y 좌표가 y인 점을 (x, y)로 표현한다. JOI 씨가 소유하고 있는 토지는 X 좌표와 Y 좌표 모두 -10^{100} 이상 10^{100} 이하인 영역이다. 이 토지에서 X 좌표와 Y 좌표 모두 $-S$ 이상 S 이하인 영역은 목초지이며, 소를 기르고 있다.

JOI 씨는 소가 도망가지 않도록 목초지를 몇 개의 울타리로 감싸기로 했다. 울타리는 길이가 양의 실수인 선분으로 표현된다. 여기서 목초지를 울타리로 감싼다는 것은, 목초지 내부의 어떤 점에서부터도 울타리가 존재하는 지점 (울타리의 경계도 포함한다) 을 지나지 않고 JOI 씨의 토지 밖으로 나가는 것이 불가능하다는 것을 의미한다. JOI 씨의 토지에는 처음에 몇 개의 울타리가 놓여있으므로 이 울타리를 사용하여 목초지를 감싸도 된다. 처음에 존재하는 어떤 두 울타리에 대해 이 두 울타리의 공통점이 존재한다면, 이 점은 적어도 한 울타리의 끝점이다.

JOI 씨는 울타리를 새로 몇 개 배치하기로 했다. 이때, 목초지의 내부나 JOI 씨의 토지의 이외를 지나도록 울타리를 배치하는 것은 불가능하지만, 목초지의 경계에 배치하는 것은 상관없다. 이 조건을 만족하면 어떤 길이의 울타리를 어떤 방향으로 배치해도 상관없다. 길이 l의 ($l > 0$) 울타리를 한 개 추가하는 데에는 비용이 l만큼 든다. 여기서 울타리끼리 교차하거나, 울타리의 끝점이 다른 끝점과 일치해도 상관없다. 또한, 울타리의 끝점이 다른 울타리에 포함되어 있어도 상관없다.

JOI 씨는 가능한 한 적은 비용으로 목초지를 울타리로 감싸려고 한다.

JOI 씨의 목초지의 크기와 처음 놓여있는 울타리의 정보가 주어졌을 때, 목초지를 울타리로 감싸는 데 필요한 비용의 최솟값을 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 두 개의 정수 N, S가 공백으로 구분되어 주어진다. 이는 토지에서 X 좌표와 Y 좌표 모두 $-S$ 이상 S 이하인 영역은 목초지이며, JOI 씨가 소유하고 있는 토지에 처음에 N 개의 울타리가 놓여있다는 것을 의미한다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N-1$) 줄에는 네 개의 정수 A_i, B_i, C_i, D_i가 공백으로 구분되어 주어진다. 이는 i 번째 울타리가 두 개의 선분 (A_i, B_i)와 (C_i, D_i)를 잇는 선분이라는 것을 의미한다.

출력 형식

목초지를 울타리로 감싸는 데 필요한 비용의 최솟값을 첫째 줄에 출력하여라. 출력하는 소수점 이하의 개수에 제한은 없지만, 답과의 절대오차가 0.01 이하여야 한다.

제한

- $1 \leq N \leq 100$.
- $1 \leq S \leq 200$.
- $-200 \leq A_i \leq 200$ $(1 \leq i \leq N)$.
- $-200 \leq B_i \leq 200$ $(1 \leq i \leq N)$.
- $-200 \leq C_i \leq 200$ $(1 \leq i \leq N)$.
- $-200 \leq D_i \leq 200$ $(1 \leq i \leq N)$.

- $(A_i, B_i) \neq (C_i, D_i)$ $(1 \leq i \leq N)$.
- 입력으로 주어지는 울타리는 목초지의 내부를 지나지 않는다.
- 입력으로 주어지는 서로 다른 두 울타리에 대해 이 두 울타리의 공통점이 존재한다면, 이 점은 적어도 한 울타리의 끝점이다.

서브태스크 1 (18 점)

- $N = 1$.

서브태스크 2 (33 점)

- $N \leq 6$.

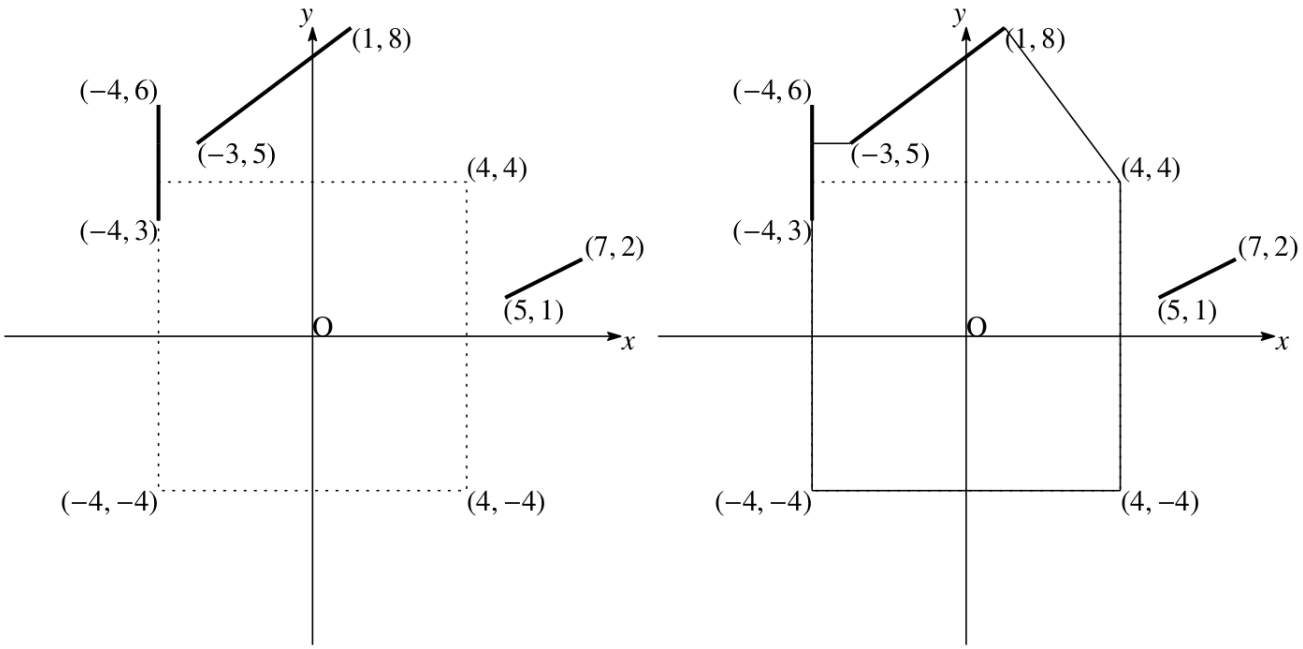
서브태스크 3 (49 점)

추가 제한조건이 없다.

예제

standard input	standard output
3 4 -3 5 1 8 -4 3 -4 6 5 1 7 2	29.0000000000

이 입력 예제에서는 처음에 왼쪽 아래의 지도처럼 울타리가 배치되어 있다. 중앙에 점선으로 표시된 정사각형은 목초지의 외곽이다.



오른쪽 아래의 실선으로 표시된 새로운 울타리를 배치하면 목초지를 감쌀 수 있다. 이때, 비용은 29이고 최소이다. 즉, 이 입력 예제의 경우 29.0000000000 이외에도 29나 28.999를 출력해도 정답으로 판정한다.

standard input	standard output
1 2 -3 -3 -3 -2	16.0000000000

처음에 배치된 울타리를 사용하지 않고 목초지를 울타리로 감쌀 수 있다는 것에 유의하여라.

standard input	standard output
4 3 4 -1 3 4 -4 2 -2 4 -4 0 -5 6 0 -6 5 -2	14.1392801789
10 80 175 95 60 -146 -106 57 18 185 190 -68 177 -142 84 -195 127 -179 34 143 126 69 -92 133 -190 80 -157 -66 -119 -161 -85 -124 129 -171 141 181 175 175 107 -38 150 148	238.4778364511

문제 3. 텐트

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	2초
메모리 제한:	512MB

JOI 군은 캠프장을 운영하고 있다. 이 캠프장은 H 행 W 열로 나뉘어 있다. 캠프장의 세로 방향은 남북 방향으로, 가로 방향은 동서 방향으로 놓여있다. 북쪽에서 i 번째 행, 서쪽에서 j 번째 열에 ($1 \leq i \leq H$, $1 \leq j \leq W$) 있는 구획을 (i, j)번 구획이라고 쓴다.

JOI 군은 몇몇 구획에 텐트를 배치할 것이다. 각 텐트는 정확히 하나의 구획을 차지해야 한다. 하나의 구획에 두 개 이상의 텐트가 있을 수 없다.

각 텐트는 동서남북 중 한 방향에 입구가 있다. 텐트의 입구 방향은 다음 조건을 만족해야 한다.

- 만약 (i_1, j)번 구획과 (i_2, j)번 구획에 ($1 \leq i_1 < i_2 \leq H$, $1 \leq j \leq W$) 모두 텐트가 놓여 있으면, (i_1, j)번 구획에 있는 텐트의 입구 방향은 남쪽으로, (i_2, j)번 구획에 있는 텐트의 입구 방향은 북쪽으로 놓여있어야 한다.
- 만약 (i, j_1)번 구획과 (i, j_2)번 구획에 ($1 \leq i \leq H$, $1 \leq j_1 < j_2 \leq W$) 모두 텐트가 놓여 있으면, (i, j_1)번 구획에 있는 텐트의 입구 방향은 동쪽으로, (i, j_2)번 구획에 있는 텐트의 입구 방향은 서쪽으로 놓여있어야 한다.

JOI 군이 캠프장에 위와 같은 조건을 만족하도록 적어도 하나의 텐트를 배치하는 경우의 수를 알고 싶다. 어떤 구역에 텐트가 배치된 방법이 (텐트가 배치되어 있는지와 텐트의 입구 방향) 다르면, 텐트를 배치하는 두 방법은 서로 다른 방법이다.

문제의 조건을 만족하도록 적어도 하나의 텐트를 배치하는 방법의 수를 1 000 000 007로 나눈 나머지를 출력하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 두 개의 정수 H, W가 공백으로 구분되어 주어진다. 이는 JOI 군이 운영하는 캠프장이 H 행 W 열로 이루어져 있다는 것을 의미한다.

출력 형식

문제의 조건을 만족하도록 캠프장에 적어도 하나의 텐트를 배치하는 방법의 수를 1 000 000 007로 나눈 나머지를 출력하여라.

제한

- $1 \leq H \leq 3\,000$.
- $1 \leq W \leq 3\,000$.

서브태스크 1 (48 점)

- $1 \leq H \leq 300$.
- $1 \leq W \leq 300$.

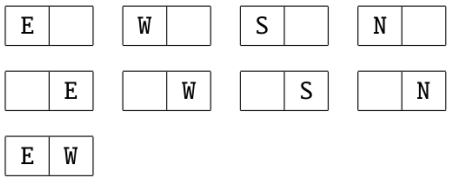
서브태스크 2 (52 점)

추가 제한조건이 없다.

예제

standard input	standard output
1 2	9

동쪽, 서쪽, 남쪽, 북쪽으로 입구가 놓인 텐트를 각각 'E', 'W', 'S', 'N'으로 표현하자. 조건을 만족하도록 텐트를 배치하는 방법은 아래와 같이 9가지가 있다.



standard input	standard output
4 3	3252
100 100	561068619

문제 4. 수행

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	0.6초
메모리 제한:	256MB

타임머신을 손에 넣은 JOI군은 약 1200년 전의 일본으로 여행을 갔다. 거기서 JOI군은 고보대사라는 이름으로 알려진 승려 구카이를 만났다. 구카이는 새로운 수행방법을 생각하고 있었다.

구카이는 다음과 같은 방법으로 수행을 하려는 중이다.

- 구카이는 N 개의 문장으로 된 불경을 읽는다. N 개의 문장에는 순서가 있어서, 구카이는 그 순서대로 읽어야 한다.
- 각각의 문장은 1 이상 N 이하의 정수 하나가 붙어있다. 단, 서로 다른 문장에 같은 정수가 붙은 경우는 없다.
- 정수 i $(1 \leq i \leq N)$ 이 붙은 문장은, 하루를 N등분 했을 때 i 번째 시간에 읽어야 한다. 각각의 문장은 매우 짧기 때문에 이 시간 안에 문장을 모두 읽는 것이 가능하다.

구카이는 어떤 날부터 하루가 시작할 때 수행을 시작해서 가장 빨리 이 수행을 끝내고 싶다. 문장에 붙은 정수에 따라 수행에 며칠이 걸리는지가 달라진다. JOI군은 가장 빨리 수행을 끝낼 때 정확히 K 일이 걸리도록 문장에 정수를 붙이는 방법이 몇 가지 있는지를 구해달라는 구카이의 부탁을 받았다.

불경의 문장 수 N과 K가 주어질 때, 불경을 모두 읽는 데 가장 빠른 방법으로 읽으면 정확히 K 일이 걸리도록 문장에 정수를 붙이는 방법의 가짓수를 1 000 000 007로 나눈 나머지를 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 두 개의 정수 N, K가 공백으로 구분되어 주어진다. 이는 불경이 N 개의 문장으로 되어 있고, 불경을 모두 읽는 데 가장 빠른 방법으로 읽으면 정확히 K 일이 걸리도록 문장에 정수를 붙이는 방법의 가지수를 구해야 한다는 것을 의미한다.

출력 형식

가장 빨리 수행을 끝낼 때 정확히 K 일이 걸리도록 문장에 정수를 붙이는 가짓수를 1 000 000 007로 나눈 나머지를 출력하여라.

제한

- $1 \leq N \leq 100\ 000$.
- $1 \leq K \leq N$.

서브태스크 1 (4 점)

- $N \leq 10$.

서브태스크 2 (20 점)

- $N \leq 300$.

서브태스크 3 (25 점)

- $N \leq 3\ 000$.

서브태스크 4 (51 점)

추가 제한조건이 없다.

예제

standard input	standard output
3 2	4

불경을 모두 읽는 데 가장 빠른 방법으로 읽으면 정확히 2일이 걸리는 경우는 아래 4가지이다.

- 첫 번째 문장에 1, 두 번째 문장에 3, 세 번째 문장에 2가 붙어 있다. 이때, 첫째 날에 (1과 3이 붙어 있는) 처음 두 개의 문장을 읽고, 둘째 날에 (2가 붙어있는) 마지막 문장을 읽는다.
- 첫 번째 문장에 2, 두 번째 문장에 1, 세 번째 문장에 3이 붙어 있다.
- 첫 번째 문장에 2, 두 번째 문장에 3, 세 번째 문장에 1이 붙어 있다.
- 첫 번째 문장에 3, 두 번째 문장에 1, 세 번째 문장에 2가 붙어 있다.

standard input	standard output
10 5	1310354

문제 5. 도로망 정비

출력 파일: `Output Only`

IOI 왕국에는 N 개의 도시가 있다. 도시에는 1번부터 N번까지의 번호가 붙어있다. 또한, IOI 왕국에는 $N-1$ 개의 양방향 도로가 있어서, 어떤 두 개의 서로 다른 도시를 골라도 몇 개의 도로를 경유하면 서로 오갈 수 있다. 도로에도 1번부터 $N-1$번까지의 번호가 붙어 있고, i번 도로는 A_i번 도시와 B_i번 도시를 잇는다.

두 도시 사이의 거리는, 두 도시를 잇는 최단 경로의 도로 개수이다. IOI 왕국의 총 거리를 모든 가능한 두 도시 쌍의 거리의 합으로 정의한다.

IOI 왕국의 왕은 K 개의 도로를 추가로 지어서 왕국의 총 거리를 줄이려 하고 있다.

왕국의 보좌관인 당신은 좋은 계획을 만들어 왕국을 도와야 한다.

IOI 왕국에 이미 있는 도로의 정보와 추가로 건설해야 하는 도로의 수 K가 주어진다. K개의 도로를 건설하는 방법을 하나 출력하여라. 왕국의 총 거리가 작으면 작을수록 더 높은 득점을 얻을 수 있다.

입력 형식

이 문제의 입력 데이터는 총 6개이다.

각 입력 데이터는 다음 형식의 파일로 주어진다.

- 첫째 줄에는 세 개의 정수 N, K, W_0가 공백으로 구분되어 주어진다. 이는 IOI 왕국이 N 개의 도시로 이루어져 있고, K 개의 도로를 새로 건설할 계획이라는 것을 의미한다. W_0는 채점을 위해 사용되는 정수이다.
- 다음 $N-1$ 개의 줄의 i 번째 $(1 \leq i \leq N-1)$ 줄에는, 두 개의 정수 A_i, B_i가 공백으로 구분되어 주어진다. 이는 i 번째 도로가 A_i번 도시와 B_i번 도시를 잇는다는 것을 의미한다.

출력 형식

K개의 줄을 출력하여라. j 번째 $(1 \leq j \leq K)$ 줄에는 두 개의 정수 X_j, Y_j $(1 \leq X_j \leq N, 1 \leq Y_j \leq N)$을 공백으로 구분하여 출력하여라. 이는 X_j번 도시와 Y_j번 도시를 잇는 도로를 건설하는 것을 의미한다.

각 입력 데이터에 대응하는 출력을 제출한다. 이때, 출력 형식이 올바른가 검사한다.

제한

- $1 \leq N \leq 1000$.
- $1 \leq A_i < B_i \leq N$ $(1 \leq i \leq N-1)$.
- $(A_i, B_i) \neq (A_k, B_k)$ $(1 \leq i < k \leq N-1)$.
- 어떤 두 개의 서로 다른 도시를 골라도 몇 개의 도로를 경유하면 서로 오갈수 있다.

배점

각 테스트 케이스에 대해, 당신의 득점은 다음과 같이 계산된다.

만약, 당신의 출력이 문제의 조건을 만족하지 않는 경우, 당신의 득점은 0점이 된다. 조건을 만족하는 경우 출력 방법대로 도로를 건설한 이후 왕국의 총 거리를 W라 하고, 배점을 P라 한다. 이 때,

$$S = 1.0 - \frac{W}{W_0}$$

이라고 하자. 이 때, 입력 데이터에 대응하는 득점은

$$\min(P, P \times 20^S)$$

이 된다.

6개의 입력 데이터 득점의 합계를 소수점 이하에서 반올림 한 정수가 이 문제의 점수가 된다.

각 입력 데이터의 N, K, W_0, P의 값은 다음과 같다.

입력 데이터	N	K	W_0	P
1	20	4	512	10
2	1000	100	2650000	18
3	1000	300	1755000	18
4	1000	100	2900000	18
5	1000	100	2690000	18
6	1000	300	1745000	18

예제

Sample Input	Sample Output
4 1 8 1 2 2 3 3 4	1 4

원래 있는 도로와 함께 1번 도시와 4번 도시를 잇는 도로를 건설하는 것으로, 총 거리가 8이 되었다.

이 입력 예제에 대해 $P = 10$이라 하자. 이때, $S = 0$이므로, 당신의 득점은 10점이다.

Sample Input	Sample Output
4 1 8 1 2 2 3 3 4	1 2

이 경우, 총 거리는 10이다.

이 입력 예제에 대해 $P = 10$이라 하자. 이때, $S = -0.25$이므로, 당신의 득점은 $4.728\cdots$ 점이다.

문제 6. 최악의 기자 3

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	2초
메모리 제한:	256MB

IOI 2018의 개회식에는 N 명의 선수가 일렬로 서 행진한다. 선수가 행진하는 도로는 수직선으로 표현된다. 선수는 전원 수직선 위의 양의 방향을 향해 행진하고 있다. 처음에 앞에서 i 번째 $(1 \leq i \leq N)$ 선수는 좌표 $-i$에 있다. 좌표 0에는 기수 IOI 양이 있다.

모든 선수에는 **게으름**이라는 값이 정해져 있다. 왼쪽에서 i 번째 선수의 게으름은 D_i이다. 선수들은 보통 다음의 규칙에 따라 행동한다.

- 앞에서 i 번째 선수는 자신의 바로 앞에 참가자와의 (선수 혹은 IOI 양) 거리가 $D_i + 1$ 이상 떨어져 있으면, 바로 앞 참가자와 거리가 1 차이 나는 위치까지 이동한다. 그렇지 않을 경우, 이동하지 않는다.

IOI 양은 단위 시각 마다 1만큼 양의 방향으로 움직인다. 선수들은 모두 위에 쓰인 조건을 만족하면 움직인다.

당신은 개회식을 취재하러 온 기자이다. 당신은 사진을 찍어야 했지만, 개회식 도중에 잠에 빠져버렸다. 어쩔 수 없이 개회장의 사진을 찍어 그 사진에 참가자의 사진을 합성하려고 했다.

사진이 합성되었다는 것을 들키지 않고 합성하는 시간을 예측하기 위해서, 당신은 다음 Q개의 값을 알고 싶다.

- 시각 T_j에, L_j 이상 R_j 이하의 좌표에 위치한 참가자의 수 $(1 \leq j \leq Q)$

각 선수의 게으름과 Q개의 질문의 정보가 주어졌을 때, 각각에 질문에 대해 조건을 만족하는 참가자의 수를 출력하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 정수 N, Q가 공백으로 구분되어 주어진다. 이는 선수의 수와 질문의 수를 의미한다. 선수의 수를 셀 때, IOI 양은 세지 않음에 주의하여라.
- 다음 N개의 줄의 i 번째 $(1 \leq i \leq N)$ 줄에는 정수 D_i가 주어진다. 이는 i 번째 선수의 게으름을 의미한다.
- 다음 Q개의 줄의 j 번째 $(1 \leq j \leq Q)$ 줄에는 정수 T_j, L_j, R_j가 주어진다. 이는 j 번째 질문의 정보를 의미한다.

출력 형식

표준 출력에 Q 개의 줄을 출력하여라. j 번째 $(1 \leq j \leq Q)$ 줄에는 j 번째 질문에 대한 답을 정수로 출력하여라.

제한

- $1 \leq N \leq 500\ 000$.
- $1 \leq Q \leq 500\ 000$.
- $1 \leq D_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.
- $1 \leq T_j \leq 1\ 000\ 000\ 000$ $(1 \leq j \leq Q)$.
- $1 \leq L_j \leq R_j \leq 1\ 000\ 000\ 000$ $(1 \leq j \leq Q)$.

서브태스크 1 (7 점)

- $D_i = 1$ $(1 \leq i \leq N)$.

서브태스크 2 (12 점)

- $N \leq 1\ 000$.
- $Q \leq 1\ 000$.
- $T_j \leq 1\ 000$. $(1 \leq j \leq Q)$.
- $1 \leq L_j \leq R_j \leq 1\ 000$ $(1 \leq j \leq Q)$.

서브태스크 3 (81 점)

추가 제한조건이 없다.

예제

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

이 입력 예제에서, 선수와 IOI 양은 다음과 같이 행진한다.

수직선의 좌표 중 L이상 R이하인 점 전체를 $[L, R]$로 표현한다.

- 처음에, IOI 양은 좌표 0에, 1, 2, 3번째 선수는 좌표 -1, -2, -3에 있다.
- 시각 1에, IOI 양이 좌표 1로 행진한다. 행진하는 선수는 없고 1, 2, 3번째 선수는 좌표 -1, -2, -3에 있다. 구간 $[2, 4]$에 아무도 없으므로, 첫 번째 질문에는 0을 출력한다.
- 시각 2에, IOI 양이 좌표 2로 행진한다. IOI 양과 첫 번째 선수의 거리가 3이 되었기 때문에, 첫 번째 선수는 좌표 1로 행진한다. 1, 2, 3번째 선수는 좌표 1, -2, -3에 있다. 구간 $[2, 4]$에 IOI 양 혼자 있으므로, 두 번째 질문에는 1을 출력한다.
- 시각 3에, IOI 양이 좌표 3으로 행진한다. 행진하는 선수는 없고 1, 2, 3번째 선수는 좌표 1, -2, -3에 있다. 구간 $[2, 4]$에 IOI 양 혼자 있으므로, 세 번째 질문에는 1을 출력한다.
- 시각 4에, IOI 양이 좌표 4로 행진한다. IOI 양과 첫 번째 선수의 거리가 3이 되었기 때문에, 첫 번째 선수는 좌표 3으로 행진한다. 1, 2, 3번째 선수는 좌표 3, -2, -3에 있다. 구간 $[2, 4]$에 IOI 양과 첫 번째 선수가 있으므로, 네 번째 질문에는 2를 출력한다.
- 시각 5에, IOI 양이 좌표 5로 행진한다. 행진하는 선수는 없고 1, 2, 3번째 선수는 좌표 3, -2, -3에 있다. 구간 $[2, 4]$에 첫 번째 선수 혼자 있으므로, 다섯 번째 질문에는 1을 출력한다.

- 시각 6에, IOI 양이 좌표 6으로 행진한다. IOI 양과 첫 번째 선수의 거리가 3이 되었기 때문에, 첫 번째 선수는 좌표 3으로 행진한다. 또한, 첫 번째 선수와 두 번째 선수의 거리가 7이 되었기 때문에, 두 번째 선수는 좌표 4로 행진한다. 또한, 두 번째 선수와 세 번째 선수의 거리가 7이 되었기 때문에, 두 번째 선수는 좌표 3으로 행진한다. 1, 2, 3번째 선수는 좌표 5, 4, 3에 있다. 구간 $[2, 4]$에 두 번째 선수와 세 번째 선수가 있으므로, 여섯 번째 질문에는 2를 출력한다.

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

이 입력 예제는 서브태스크 1의 조건을 만족한다,

standard input	standard output
6 6 11 36 28 80 98 66 36 29 33 190 171 210 18 20 100 1000 900 1100 92 87 99 200 100 300	1 6 0 5 2 7

문제 7. 항공 노선도

시간 제한: 2초
메모리 제한: 1024MB

Alice는 JOI 왕국에 살고 있다. 그는 IOI 공화국에 사는 Bob을 초대하게 되어 JOI 왕국 내의 항공 노선도를 Bob에게 보내게 되었다. JOI 왕국은 0번 섬부터 $N-1$번 섬까지 총 N 개의 섬으로 이루어진 섬나라이다. JOI 왕국에는 M 개의 항공 노선이 있어서 $i+1$ 번째 $(0 \leq i \leq M-1)$ 항공 노선은 A_i번 섬과 B_i번 섬을 양방향으로 잇는다. 같은 쌍의 섬을 잇는 서로 다른 두 개의 항공 노선은 존재하지 않는다. JOI 왕국에서 IOI 공화국으로 정보를 보내기 위해서는, JOI 왕국이 운영하는 전보를 써야 한다. 하지만 전보를 쓸 때, 정점 번호와 간선 번호가 무작위로 섞일 것이다.

정확히 전보는 다음과 같은 순서로 보내지게 된다. G를 Alice가 보내는 그래프라고 하자. (V를 G의 정점 개수, U를 G의 간선 개수라고 하자.)

- Alice는 G의 정점 개수 V와 간선 개수 U를 지정한다. 또한, G의 각 정점에 $0, 1, \cdots, V-1$의 번호를 G의 각 간선에 $0, 1, \cdots, U-1$의 번호를 각각 붙인다.
- Alice는 인자 $C_0, C_1, \cdots, C_{U-1}$ 및 $D_0, D_1, \cdots, D_{U-1}$을 지정한다. 이 인자는 G의 간선을 의미한다. 즉, 각 j $(0 \leq j \leq U-1)$ 에 대해, G의 간선 j는 정점 C_j와 D_j를 연결한다.
- JOI 왕국에 의해 G의 정점 번호가 무작위로 섞인다. 우선 JOI 왕국은 $0, 1, \cdots, V-1$로 이루어진 순열 $p[0], p[1], \cdots, p[V-1]$을 생성한다. 다음, $C_0, C_1, \cdots, C_{U-1}$을 $p[C_0], p[C_1], \cdots, p[C_{U-1}]$로, $D_0, D_1, \cdots, D_{U-1}$을 $p[D_0], p[D_1], \cdots, p[D_{U-1}]$로 바꾼다.
- 이어서, JOI 왕국에 의해 G의 간선 번호가 무작위로 섞인다. 우선 JOI 왕국은 $0, 1, \cdots, U-1$로 이루어진 순열 $q[0], q[1], \cdots, q[U-1]$을 생성한다. 다음, $C_0, C_1, \cdots, C_{U-1}$을 $C_{q[0]}, C_{q[1]}, \cdots, C_{q[U-1]}$로, $D_0, D_1, \cdots, D_{U-1}$을 $D_{q[0]}, D_{q[1]}, \cdots, D_{q[U-1]}$로 바꾼다.
- V, U의 값과 바뀐 후의 인자 $C_0, C_1, \cdots, C_{U-1}$ 및 $D_0, D_1, \cdots, D_{U-1}$가 Bob에게 전달된다.

단, 이 전보로 보낼 수 있는 그래프는 단순 그래프 뿐이다. 단순 그래프는 다중간선이나 루프가 없는 그래프이다. 즉, 각 i, j $(0 \leq i < j \leq U-1)$에 대해 $(C_i, D_i) \neq (C_j, D_j)$ 와 $(C_i, D_i) \neq (D_j, C_j)$를 만족하며 또한 모든 i $(0 \leq i \leq U-1)$ 에 대해 $C_i \neq D_i$ 를 만족하는 그래프를 보내는 것이 가능하다.

Alice는 가능한 한 적은 정점 수의 그래프로 Bob에게 JOI 왕국 내의 항공 노선도를 보내고 싶다.

Alice와 Bob의 통신을 구현하기 위해 다음 2개의 프로그램을 작성하여라.

1. 첫 번째 프로그램은 JOI 왕국의 섬의 수 N, JOI 왕국의 항공 노선의 수 M, JOI 왕국의 항공 노선을 의미하는 수열 A, B가 주어질 때, Alice가 보내는 그래프 G의 정보를 만든다.
2. 두 번째 프로그램은 Bob이 받은 그래프 G가 주어질 때, JOI 왕국의 항공 노선도를 복원한다.

구현 명세

당신은 파일 두 개를 제출해야 한다.

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

- `void Alice(int N, int M, int A[], int B[])`

  이 함수는 각 테스트케이스마다 정확히 한 번 불린다.

  - 인자 `N`은 JOI 왕국의 섬의 수를 의미한다.
  - 인자 `M`은 JOI 왕국의 항공 노선의 수 M을 의미한다.
  - 인자 `A[]`, `B[]`는 길이 M의 배열이며, JOI 왕국의 항공 노선을 의미한다.

함수 Alice 안에서는 다음의 함수를 호출해서 보낼 그래프 G의 정보를 설정한다.

- `void InitG(int V, int U)`
  이 함수를 호출하는 것으로 G의 정점 개수와 간선 개수를 설정한다.
  * 인자 V는 G의 정점 개수를 의미한다. V는 1 이상 1500 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답** [1]이 된다.
  * 인자 U는 G의 간선 개수를 의미한다. U는 0 이상 $V(V-1)/2$ 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답** [2]이 된다.
- `void MakeG(int pos, int C, int D)`
  이 함수는 G의 간선을 설정한다.
  * 인자 pos는 G의 정점 번호를 의미한다. pos는 0 이상 $U-1$ 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답** [3]이 된다. 또한, 같은 인자 pos로 함수를 두 번 이상 호출할 수 없다. 같은 인자로 두 번 이상 호출한 경우 **오답** [4]이 된다.
  * 인자 C와 D는 pos번 간선의 양쪽의 정점을 의미한다. C와 D는 0 이상 $V-1$ 이하의 정수여야 한다. 또한, C $\neq$ D를 만족해야 한다. C 혹은 D가 이 조건을 만족하지 않은 경우, **오답** [5]가 된다.

  여기서 U와 V는 initG에서 설정된 값이다.

함수 Alice 안에서는 함수 InitG를 한 번 호출한 후, 함수 MakeG를 정확히 U번 호출해야 한다. 함수 InitG가 두 번 이상 호출된 경우 **오답** [6]이 된다. 함수 InitG가 호출되기 전에 함수 MakeG가 호출된 경우 **오답** [7]이 된다. 함수 Alice의 실행이 완료될 때 함수 InitG가 호출되지 않은 경우 혹은 MakeG의 호출 횟수가 U가 아니었던 경우 **오답** [8]이 된다. 함수 Alice의 실행이 종료될 때 지정한 그래프 G가 단순 그래프가 아닌 경우, **오답** [9]이 된다.

Alice의 호출이 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

둘째 파일의 이름은 Bob.cpp이다. 이 파일은 Bob의 일을 나타내고 다음 함수를 구현해야 한다. 또한, Bob.h를 include해야 한다.

- `void Bob(int V, int U, int C[], int D[])`
  이 함수는 각 테스트케이스마다 정확히 한 번 불린다.
  - 인자 V은 그래프 G의 정점 개수를 의미한다.
  - 인자 U은 그래프 G의 간선 개수를 의미한다.
  - 인자 C[], D[]는 길이 U의 배열이며, 그래프 G를 의미한다.

  함수 Bob 안에서는 다음의 함수를 호출해서 복원한 JOI 왕국의 항공 노선도의 정보를 복원한다.

  - `void InitMap(int N, int M)`
    이 함수를 호출하는 것으로 JOI 왕국의 섬의 수와 JOI 왕국의 항공 노선의 수를 복원한다.
    * 인자 N은 복원한 JOI 왕국의 섬의 수를 의미한다. N은 실제 JOI 왕국의 섬의 수와 일치해야 한다. 일치하지 않을 경우, **오답** [10]이 된다.
    * 인자 M은 복원한 JOI 왕국의 항공 노선의 수를 의미한다. N은 실제 JOI 왕국의 항공 노선의 수와 일치해야 한다. 일치하지 않을 경우, **오답** [11]이 된다.
  - `void MakeMap(int A, int B)`
    이 함수를 호출하는 것으로 JOI 왕국의 항공 노선을 복원한다.
    * 인자 A와 B는 JOI 왕국의 **A**번 섬과 **B**번 섬을 잇는 항공 노선이 존재한다는 것을 의미한다. A와 B는 0 이상 $N-1$ 이하의 정수여야 한다. 또한, A $\neq$ B를 만족해야 한다. A 혹은 B가 이 조건을 만족하지 않은 경우 **오답** [12]가 된다. JOI 왕국의 섬 A와 섬 B를 연결하는 항공 노선이 없을 경우 **오답** [13]이 된다. 또한, 과거에 호출한 것과 같은 항공 노선을 호출해서는 안 된다. MakeMap(A, B)를 호출할 때 이미 MakeMap(A, B) 또는 MakeMap(B, A)를 호출한 경우 **오답** [14]이 된다.

여기서 N은 `initMap`에서 설정된 값이다.

함수 `Bob` 안에서는 함수 `InitMap`을 한 번 호출한 후, 함수 `MakeMap`을 정확히 M번 호출해야 한다. 함수 `InitMap`이 두 번 이상 호출된 경우 **오답** [**15**]이 된다. 함수 InitMap이 호출되기 전에, 함수 `MakeMap`이 호출된 경우 **오답** [**16**]이 된다. 함수 `Bob`의 실행이 완료 될 때, 함수 `InitMap`이 호출되지 않은 경우 혹은 `MakeMap`의 호출 횟수가 M이 아니었던 경우 **오답** [**17**]이 된다. 여기서 M은 `initMap`에서 설정된 값이다.

`Bob`의 호출이 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

채점은 다음과 같은 방식으로 진행된다. 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

(1) JOI 왕국의 항공 노선도의 정보를 인자로 하여 함수 `Alice`를 한 번 호출한다.

(2) 함수 `Alice`안에서 설정된 그래프를 G라고 하자. G의 정점 번호와 간선 번호를 뒤섞은 그래프를 의미하는 정보를 인자로 하여 함수 `Bob`을 한 번 호출한다.

(3) 채점이 종료된다.

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 위하여 `static`으로 선언되어야 한다. 채점될 때는, Alice과 Bob에 해당하는 두 프로세스로 나누어서 실행될 것이다. Alice의 프로세스와 Bob의 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`, `Alice.cpp`, `Bob.cpp`, `Alicelib.h`, `Boblib.h`를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=c++14 -O2 -o grader grader.cpp Alice.cpp Bob.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.

- 첫째 줄에는 정수 N, M이 주어진다. 이는 JOI 왕국의 섬의 수가 N이며, JOI 왕국의 항공 노선의 수가 M이라는 것을 의미한다.
- 다음 M 개의 줄에는 M 개의 항공 노선의 정보가 주어진다. M개의 줄 중 $i+1$ 번째 $(0 \leq i \leq M-1)$ 줄에는 두 개의 정수 A_i, B_i가 주어진다. 이는 JOI 왕국의 항공 노선의 정보를 의미한다.

출력 형식

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

- 프로그램의 실행 중에 오답으로 판단된 경우, 오답의 종류를 "`Wrong Answer [1]`"과 같은 형식으로 출력하고, 실행이 종료된다.
- `Alice`와 `Bob` 모두 오답이 아닌 경우, "`Accepted`"를 출력하고 또한 V의 값이 출력된다.

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

제한

- $1 \leq N \leq 1\,000$.
- $0 \leq M \leq N(N-1)/2$.
- $0 \leq A_i \leq N-1$ $(0 \leq i \leq M-1)$.
- $0 \leq B_i \leq N-1$ $(0 \leq i \leq M-1)$.
- $A_i \neq B_i$ $(0 \leq i \leq M-1)$.
- $(A_i, B_i) \neq (A_j, B_j)$ 이며, $(A_i, B_i) \neq (B_j, A_j)$ $(0 \leq i < j \leq M-1)$.

서브태스크 1 (22 점)

- $N \leq 22$

서브태스크 2 (15 점)

- $N \leq 40$

서브태스크 3 (63 점)

추가 제한조건이 없다.

배점

- 서브태스크 1 혹은 2에서는, 각 서브태스크의 모든 테스트 케이스를 맞은 경우, 만점이다.
- 서브태스크 3에서는, 모든 테스트 케이스를 맞은 경우, $V - N$의 최댓값을 MaxDiff로 해서, 다음 기준으로 채점한다.
  - $101 \leq$ MaxDiff이면, 0점.
  - $21 \leq \text{MaxDiff} \leq 100$이면, $13 + \left\lfloor \dfrac{100 - \text{MaxDiff}}{4} \right\rfloor$점. 여기서, x를 넘지 않는 최대의 정수를 $\lfloor x \rfloor$로 표현한다.
  - $13 \leq \text{MaxDiff} \leq 20$이면, $33 + (20 - \text{MaxDiff}) \times 3$점.
  - MaxDiff ≤ 12이면, 63점.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력 1	함수 호출의 예			
	호출	반환값	호출	반환값
	`Alice(...)`			
			`InitG(4,3)`	
				(없음)
			`MakeG(0,0,1)`	
				(없음)
			`MakeG(1,0,2)`	
				(없음)
			`MakeG(2,0,3)`	
`4 3`				(없음)
`0 1`		(없음)		
`0 2`	`Bob(...)`			
`0 3`			`InitMap(4,3)`	
				(없음)
			`MakeMap(0,1)`	
				(없음)
			`MakeMap(0,2)`	
				(없음)
			`MakeMap(0,3)`	
				(없음)
		(없음)		

이 때, `Alice(...)`, `Bob(...)`에 주어진 인자는 다음과 같다.

인자	Alice(...)	Bob(...)
N	4	
M	3	
V		4
U		3
A	{0,0,0}	
B	{1,2,3}	
C		{2,2,2}
D		{3,0,1}

예제 입력 2

`5 7`
`0 1`
`0 2`
`1 3`
`1 4`
`3 4`
`2 3`
`2 4`

문제 8. 비타로의 파티

시간 제한: 2초
메모리 제한: 512MB

비버가 사는 마을 N개가 높이가 감소하는 순으로 1번부터 N번까지 번호가 붙어있다. 어떤 두 마을도 같은 높이에 있지 않다. 서로 다른 마을을 단방향으로 잇는 수로 M 개가 존재한다. i 번째 $(1 \leq i \leq M)$ 수로는 S_i 번 마을에서 E_i번 마을로 흐른다. 이 수로는 높은 마을에서 낮은 마을로 흐른다. 수로를 거슬로 올라갈 수는 없다.

비버 비타로는 N 명의 친구가 있고 N 개의 마을에 각각 한 마리씩 살고 있다.

비타로는 친구를 초대해서 파티를 Q 번 할 것이다. j 번째 $(1 \leq j \leq Q)$ 파티에는 Y_j 마리의 비버가 너무 바빠서 참석하지 못한다. j 번째 파티는 T_j번 마을에서 열리기 때문에, 수로를 통해서 T_j번 마을로 올 수 없는 친구들도 참석할 수 없다. 다른 친구들은 파티에 참석한다.

각 친구는 수로를 통해서 파티가 개최되는 마을로 이동한다. 이동하는 경로가 여러 개 존재할 수도 있다. 하지만 비타로의 친구들은 수로를 너무 좋아해서, 여러 개의 경로가 있는 경우에는 거치는 수로의 개수가 최대가 되는 경로로 이동한다.

비타로는 각각의 파티에 대해, 가장 많은 수의 수로를 거친 친구가 몇 개의 수로를 경유했는지가 궁금했다.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 세 정수 N, M, Q가 공백으로 구분되어 주어진다. 이는 마을이 N 개 있고, 수로가 M 개 있으며, 비타로가 Q 번의 파티를 연다는 것을 의미한다.
- 다음 M 개의 줄에는 수로의 정보가 주어진다. M 개의 줄의 i 번째 $(1 \leq i \leq M)$ 줄에는, 정수 S_i, E_i가 공백으로 구분되어 주어진다. 이는 S_i번 마을 부터 E_i번 마을까지 단방향인 수로가 흐르고 있다는 것을 의미한다.
- 다음 Q개의 줄에는 비타로가 여는 파티의 정보가 주어진다. Q 개의 줄의 j 번째 $(1 \leq j \leq Q)$ 줄에는, 정수 T_j, Y_j와 Y_j개의 정수 $C_{j,1}, C_{j,2}, \cdots, C_{j,Y_j}$가 공백으로 구분되어 주어진다. 이는, j 번째 파티는 T_j번 마을에서 열리고, $C_{j,1}, C_{j,2}, \cdots, C_{j,Y_j}$번 마을에 사는 친구는 너무 바빠서 참석할 수 없다는 것을 의미한다.

출력 형식

표준 출력으로 Q개의 줄을 출력하여라. j 번째 $(1 \leq j \leq Q)$ 줄은 j 번째 파티에 대해 가장 많은 수의 수로를 경유한 친구가 거친 수로의 개수를 출력하여라. 단, 파티에 친구가 한 명도 참석하지 않을 경우 대신 `-1`을 출력하여라.

제한

- $1 \leq N \leq 100\ 000$.
- $0 \leq M \leq 100\ 000$.
- $1 \leq Q \leq 100\ 000$.
- $1 \leq S_i < E_i \leq N$ $(1 \leq i \leq M)$.
- $(S_i, E_i) \neq (S_j, E_j)$ $(1 \leq i < j \leq M)$.
- $1 \leq T_j \leq N$ $(1 \leq j \leq Q)$.
- $0 \leq Y_j \leq N$ $(1 \leq j \leq Q)$.

- $1 \leq C_{j,1} < C_{j,2} < \cdots < C_{j,Y_j} \leq N$ $(1 \leq j \leq Q)$.
- $Y_1 + Y_2 + \cdots + Y_Q \leq 100\ 000$.

서브태스크 1 (7 점)

- $N \leq 1\ 000$.
- $M \leq 2\ 000$.
- $Q = 1$.

서브태스크 1 (7 점)

- $Q = 1$.

서브태스크 3 (86 점)

추가 제한조건이 없다.

예제

test	answer
5 6 3 1 2 2 4 3 4 1 3 3 5 4 5 4 1 1 5 2 2 3 2 3 1 4 5	1 3 0

첫 번째 파티에 참석한 친구 (2번, 3번, 4번 마을에 사는 친구) 중에서는 2번 마을에 사는 친구와 3번 마을에 살고있는 친구가 파티에 개최되는 4번 마을까지 거친 수로의 수가 제일 많다. 이는 1개이므로, 1을 출력한다.

두 번째 파티에 참석한 친구 (1번, 4번, 5번 마을에 사는 친구) 중에서는 1번 마을에 사는 친구가 파티에 개최되는 5번 마을까지 거친 수가 제일 많다. 이는 3개이므로, 3을 출력한다.

세 번째 파티에 참석하는 친구는, 파티가 개최되는 2번 마을에 사는 친구밖에 없다. 이 친구는 수로를 경유하지 않으므로, 0을 출력한다.

test	answer
12 17 10 1 2 2 3 3 4 1 5 2 6 3 7 4 8 5 6 6 7 7 8 5 9 6 10 7 11 8 12 9 10 10 11 11 12 6 3 1 7 12 3 7 1 2 3 4 5 6 7 11 3 1 3 5 9 2 1 9 8 4 1 2 3 4 1 1 1 12 0 10 3 1 6 10 11 8 2 3 5 6 7 9 10 11 8 7 2 3 4 5 6 7 8	1 -1 3 1 3 -1 5 2 4 4

문제 9. 보안 게이트

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	5초
메모리 제한:	1536MB

당신은 Just Odd Invention이라는 회사를 아는가? 이 회사의 일은 "그저 기묘한 발명 (just odd invetion)"을 하는 것이다. 줄여서 JOI 회사라고 부른다.

JOI 회사에서는 기밀정보를 지키기 위해 입구에 보안 게이트를 설치했다. 회사를 출입하기 위해서는 반드시 보안 게이트를 거칠 필요가 있다. 또한, 한 번에 두 명 이상의 사람이 보안 게이트를 지날 수는 없다.

이 보안 게이트는 사람이 게이트를 통과할 때 사람이 회사로 들어왔는지 혹은 나왔는지에 대한 정보를 기록한다. 이제 JOI 회사의 사원 IOI군이 어떤 날의 게이트 기록을 보았다. 기록은 문자열 S로 표현된다. S의 i 번째 문자가 '('인 경우, 게이트를 i 번째로 통과한 사람이 회사로 들어갔다는 의미이고, S의 i 번째 문자가 ')'인 경우, 게이트를 i 번째로 통과한 사람이 회사에서 나왔다는 의미이다. IOI군은 이날의 시작 혹은 마지막 날의 JOI 회사 안에 아무도 없다는 것을 알고 있다. '(' 과 ')'로만 이루어져 있는데, 기록으로 등장할 수 없는 문자열이 있을 수 있음에 유의하여라. 예를 들어, `())(` 혹은 `(()`같은 문자열은 기록으로 등장할 수 없는데, JOI 회사 안에 있는 사람의 수가 음수가 되어야 하거나, 이날의 시작 혹은 마지막 날의 JOI 회사 안에 사람이 있었다는 의미이기 때문이다.

IOI군이 기록을 확인한 순간 S는 JOI 회사에 퍼져나간 바이러스로 인해 바뀌었다! 조사 이후에, 그는 수정이 다음과 같은 절차로 이루어져 있다고 가정했다:

- 처음에 S의 연속된 구간에 있는 문자열이 모두 다음과 같이 바뀌었다. 그 구간에 포함된 문자 전부에 대해 각 문자가 '('이라면 ')'로 바뀌고, ')'이라면 '('로 바뀌었다. 변화 후의 문자열을 S'이라고 하자. 여기서 변화한 구간의 길이가 0일 수도 있다, 즉, $S = S'$일 수도 있다.
- 다음에 S'의 0개 이상의 문자가 'x'로 바뀌었다. 변화 후의 문자열은 S''이다.

IOI군은 S의 정보를 기억하고 있지 않기 때문에, S''으로부터 S를 복원하려고 생각하고 있다. 이를 위해 IOI군은 우선 S'으로 가능한 문자열의 경우의 수를 세고 싶다. (S가 아님에 주의하여라.)

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 정수 N이 주어진다. 이는 S''의 길이가 N이라는 의미이다.
- 다음 줄에는 각 문자가 '(', ')' 혹은 'x'인 문자열 S''이 주어진다.

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 출력은 S'으로 가능한 문자열의 경우의 수를 1 000 000 007 로 나눈 나머지여야 한다. 만약 해당하는 S'이 없으면 0을 출력하여라.

제한

- $1 \leq N \leq 300$.

서브태스크 1 (4 점)

- $N \leq 100$.
- S''의 x의 개수는 최대 4개이다.

서브태스크 2 (8 점)

- $N \leq 100$.
- S''의 x의 개수는 최대 12개이다.

서브태스크 3 (18 점)

- $N \leq 100$.
- S''의 x의 개수는 최대 20개이다.

서브태스크 4 (43 점)

- $N \leq 100$.

서브태스크 5 (27 점)

추가 제한조건이 없다.

예제

standard input	standard output
4 x))x	3

이 입력 예제에서, $S' =$))(은 S가 될 수 있는 문자열이 존재하지 않기 때문에 불가능하다. 다음 세 문자열이 S'이 될 수 있다.

- $S' =$ ())(. 예를 들어, $S =$ ()()
- $S' =$ ()). 예를 들어, $S =$ ()()
- $S' =$)). 예를 들어, $S =$ (())

이 세 문자열만 S'이 될 수 있으므로, 3을 출력한다.

standard input	standard output
10 xx(xx()x(x	45
5 x))x(	0
10 xxxxxxxxxx	684

문제 10. 사탕

입력 파일: standard input
출력 파일: standard output
시간 제한: 5초
메모리 제한: 512MB

책상 위에 N 개의 사탕이 있다. 각 사탕은 **맛**이란 값이 붙어 있다. 왼쪽에서 i 번째 ($1 \leq i \leq N$) 사탕의 맛은 A_i이다.

JOI양은 N개의 사탕 중 몇 개를 먹기로 결심했다. JOI양은 자신이 먹는 사탕의 맛의 합을 최대화하고 싶다. 하지만 사탕을 평범하게 선택하는 것이 재미 없다고 생각한 JOI양은 연속된 두 개의 사탕을 고르면 안 된다는 규칙을 만들었다.

JOI양은 얼마나 몇 개의 사탕을 먹을지 아직 결정하지 않았기 때문에, 각 j ($1 \leq j \leq \lceil \frac{N}{2} \rceil$)에 대해, j개의 사탕을 먹을 때 가능한 맛의 합의 최댓값을 구하고 싶어한다. 여기서 $\lceil x \rceil$은, x보다 크거나 같은 최소의 정수를 의미한다.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 정수 N이 주어진다. 이는 책상 위에 N 개의 사탕이 있다는 의미이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 A_i가 주어진다. 이는 왼쪽에서 i 번째 사탕의 맛이 A_i임을 의미한다.

출력 형식

표준 출력으로 $\lceil \frac{N}{2} \rceil$개의 줄을 출력하여라. j 번째 ($1 \leq j \leq \lceil \frac{N}{2} \rceil$) 줄은 j 개의 사탕을 먹을 때 가능한 맛의 합의 최댓값이다.

제한

- $1 \leq N \leq 200\ 000$.
- $1 \leq A_i \leq 1\ 000\ 000\ 000$ ($1 \leq i \leq N$).

서브태스크 1 (8 점)

- $N \leq 2\ 000$.

서브태스크 2 (92 점)

추가 제한조건이 없다.

예제

standard input	standard output
5 4 5 1 7 6	7 12 10

이 입력 예제에서는 5개의 사탕이 있고, 맛은 왼쪽에서부터 각각 3, 5, 1, 7, 6이다.

JOI양은 다음과 같이 사탕을 먹는다.

- 한 개의 사탕을 먹을 때는 왼쪽에서부터 4번째 사탕을 먹는다. (맛은 7이다.)
- 두 개의 사탕을 먹을 때는 왼쪽에서부터 2번째, 4번째 사탕을 먹는다. (맛은 5, 7이다.)
- 세 개의 사탕을 먹을 때는 왼쪽에서부터 1번째, 3번째, 5번째 사탕을 먹는다. (맛은 3, 1, 6이다.)

연속된 두 개의 사탕을 모두 고르는 것은 불가능 하다. 예를 들면, 두 개의 사탕을 먹을 때, 왼쪽에서 부터 4번째, 5번째 사탕을 (맛은 7, 6이다) 먹는 것은 불가능하다는 것에 주의하여라.

standard input	standard output
20 623239331 125587558 908010226 866053126 389255266 859393857 596640443 60521559 11284043 930138174 936349374 810093502 521142682 918991183 743833745 739411636 276010057 577098544 551216812 816623724	936349374 1855340557 2763350783 3622744640 4439368364 5243250666 5982662302 6605901633 7183000177 7309502029

문제 11. 도서관

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	2초
메모리 제한:	256MB

수백 년의 시간이 지난 끝에, JOI 도시는 폐허가 되었다. 탐험가 IOI양은 도서관이 있었던 지역을 조사하고 있다. 조사를 통해 다음과 같은 것들을 알 수 있다.

- JOI 도시의 도서관 서재에는 N 권의 책이 있었다. N 권의 책은 책장에 왼쪽에서 오른쪽으로 일렬로 놓여 있었다.
- N 권의 책은 1번부터 N번까지의 번호가 붙어있다. 하지만 책장에 꽂힌 순서와 번호가 붙은 순서는 다를 수도 있다.
- 작업 한 번으로 책장에서 연속으로 놓인 책을 가져갈 수 있었다.

유감스럽게, IOI양은 도서관에서 오래된 책을 찾는 데에 실패했다. 하지만 도서관의 작업을 담당하는 기계를 찾았다. 한 개 이상의 책 번호를 기계한테 물어볼 경우, 기계는 이 책들을 가져가기 위해서 필요한 작업의 최소횟수를 답해주었다.

IOI양은 기계에 질문을 하면서 책이 위치한 순서를 알고 싶어한다. 단, N개의 책이 역순으로 놓여있는 경우에도 답은 변하지 않으므로 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다.

기계가 오래되었기 때문에 최대 20 000번의 질문만 할 수 있다.

구현 명세

당신은 파일 하나를 제출해야 한다.

이 파일의 이름은 `library.cpp`이다. 파일은 다음 함수를 구현해야 한다. 또한, `library.h`를 include해야 한다.

- `void Solve(int N)`

  이 함수는 각 테스트 케이스마다 정확히 한 번 불린다.

  - 인자 `N`은 책의 수 N을 나타낸다.

  당신의 프로그램은 다음 함수를 호출 할 수 있다.

  - `int Query(const std::vector<int>& M)`

    한 개 이상의 책 번호를 기계에 물어볼 경우, 이 함수는 책들을 가져가기 위해서 필요한 작업의 최소횟수를 반환한다.

    * 가져갈 책의 번호들은 인자 `M`으로 표시되며, 이는 크기 N의 'vector'이다. 각 i $(1 \leq i \leq N)$에 대해, `M[`i`-1]`$= 0$인 경우, i 번째 책은 가져가지 않는다. `M[`i`-1]`$= 1$인 경우, i 번째 책은 가져간다. 만약 `M`의 크기가 N과 다를 경우 **오답** **[1]**이 된다. 각 i에 대해, `M[`i`-1]`은 0 혹은 1이어야 한다. `M[`i`-1]`$= 1$인 i가 $(1 \leq i \leq N)$ 적어도 한 개는 존재해야 한다. 이 두 조건을 만족하지 않을 경우 **오답** **[2]**이 된다. `Query`를 20 000번 초과로 호출할 경우, **오답** **[3]**이 된다.

  - `void Answer(const std::vector<int& res)`

    이 함수를 이용하여 책꽂이에 꽂힌 책의 순서를 답할 수 있다. 책이 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다.

    * 인자 `res`는 크기 N의 'vector'이다. 이는 책꽂이에 꽂혀있는 책의 순서를 나타낸다. 각 i $(1 \leq i \leq N)$ 에 대해 책꽂이에 왼쪽에서 i 번째로 꽂혀있는 책의 번호는 `res[`i`-1]`이다. 만약 `res`의 크기가 N과 다를 경우, **오답** **[4]**이 된다. 만약 `res[`i`-1]`는 1 이상 N 이하의 수여야 한다. 만약 이를 만족하지 않을 경우, **오답** **[5]**이 된다. 또한, 수 `res[0]`, `res[1]`, `res[`N`-1]`은 모두 달라야 한다. 만약 이를 만족하지 않을 경우, **오답** **[6]**이 된다.

Solve함수가 종료될 때, Answer함수를 호출한 횟수가 한 번이 아니면, **오답** [7]이 된다. 만약 Solve로 주어진 책의 순서가 책장에 꽂힌 순서와 다르면 **오답** [8]이 된다. 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경쓰지 않는다.

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.cpp이다. 당신의 프로그램을 테스트하기 위해서, grader.cpp, library.cpp, library.h를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=c++14 -O2 -o grader grader.cpp library.cpp`

컴파일이 성공적이면, 파일 grader가 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.

- 첫째 줄에는 정수 N이 주어진다. 이는 책꽂이에 책이 N 권 있다는 의미이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 A_i가 주어진다. 이는 왼쪽에서 i 번째 꽂힌 책의 번호가 A_i임을 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. N과 A_i의 의미는, 입력 형식을 참고하여라.

- $1 \leq N \leq 1\,000$.
- $1 \leq A_i \leq N$ ($1 \leq i \leq N$).
- $1 \leq A_i \neq A_j \leq N$ ($1 \leq i < j \leq N$).

서브태스크 1 (19 점)

- $N \leq 200$.

서브태스크 2 (81 점)

추가 제한조건이 없다.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력	예제 함수 호출		
	호출	호출	반환값
5	`Solve(5)`		
4		`Query({1,1,1,0,0})`	
2			2
5		`Answer({4,2,5,3,1})`	
3			(없음)
1			

이 문제에서 책이 놓인 방향이 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다. 그렇기 때문에, 배열을 거꾸로 쓴 `Answer({1,3,5,2,4})`를 호출하여도 정답이다.

문제 12. 멧돼지

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	10초
메모리 제한:	1024MB

멧돼지 JOI군이 사는 IOI 숲에는 N개의 야생동물 급식소가 있고, M개의 도로가 있다. 급식소에는 1번 부터 N번 까지의 번호가 붙어있다. i 번째 ($1 \leq i \leq M$) 도로는 A_i번 급식소와 B_i번 급식소를 양방향으로 이으며, JOI군이 이 도로를 지나는 데 양방향 모두 C_i 시간이 걸린다. 어떤 급식소에서도 다른 모든 급식소까지 한 개 이상의 도로를 통해 서로 오가는 것이 가능하다.

JOI군은 유턴을 잘 못 하기 때문에, 도로를 가던 도중 유턴해서 출발했던 급식소로 돌아올 수 없다. 또한, 급식소에 도착한 이후에는 가장 최근에 사용한 도로를 써서 되돌아가는 것도 할 수 없다.

매일 JOI군은 보급계획에 따라서 음식을 먹는다. 하루의 보급 계획은 음식을 보급받을 L개의 급식소 번호를 차례로 나열한 $X_1, X_2, \cdots, X_L$이다. JOI군은 X_1번 급식소에서 시작해서 해당 순서로 급식소를 방문 하여 X_L번을 마지막으로 보급계획을 끝낸다. 중간에 다른 급식소를 거쳐도 된다. 보급 계획에 같은 급식소가 여러 번 등장할 수는 있지만, 모든 j ($1 \leq j \leq L-1$) 에 대해, $X_j \neq X_{j+1}$을 만족한다. 보급계획 중에는 불가능한 것이 있을 수도 있다.

처음에 JOI군은 최초 보급계획 $X_1, X_2, \cdots, X_L$을 정한다. k 번째 ($1 \leq k \leq T$) 날 아침, JOI군은 P_k번째 값을 Q_k로 바꿀 것이다. (즉, X_{P_k} 가 Q_k가 된다. 그리고 새로운 보급계획을 따라 음식을 먹는다. 모든 j ($1 \leq j \leq L-1$) 에 대해, 값이 바뀐 이후에도 $X_j \neq X_{j+1}$을 만족함이 보장된다.

T 일 동안 각 날의 보급계획에 대해 해당하는 보급계획이 가능한지 불가능 한지 판단하고, 가능하다면 해당 보급계획을 진행할 수 있는 가장 짧은 시간을 출력하여라.

입력 형식

표준 입력에서 다음 입력이 주어진다.

- 첫째 줄에는 네 정수 N, M, T, L이 공백으로 구분되어 주어진다. 이는 IOI 숲에 N개의 급식소와 M개의 도로가 있으며, JOI군은 T 일 동안의 보급계획을 생각 중이고, 각 보급계획이 L개의 급식소 번호를 차례로 나열했다는 의미이다.
- 다음 M 개의 줄의 i 번째 ($1 \leq i \leq M$) 줄에는 공백으로 구분된 세 정수 A_i, B_i, C_i가 주어진다. 이는 왼쪽에서 i 번째 도로가 A_i번 급식소와 B_i번 급식소를 양방향으로 연결하며, 이 도로를 지나는 데 양방향 모두 C_i 시간이 걸린다는 것이다.
- 다음 L 개의 줄의 j 번째 ($1 \leq j \leq L$) 줄에는 정수 X_j가 주어진다. 이는 최초 보급계획이 $X_1, X_2, \cdots, X_L$임을 의미한다.
- 다음 T 개의 줄의 k 번째 ($1 \leq k \leq T$) 줄에는 공백으로 구분된 두 정수 P_k, Q_k가 주어진다. 이는 k 번째 날 아침에 JOI군이 보급계획의 P_k 번째 값을 Q_k로 바꾼다는 의미이다.

출력 형식

표준 출력으로 T 개의 줄을 출력하여라. k 번째 ($1 \leq k \leq T$) 줄은 k번째 보급계획이 불가능 하면 `-1`, 가능하면 보급계획을 실행하는 데 드는 최소시간이다.

제한

- $2 \leq N \leq 2\,000$.
- $N - 1 \leq M \leq 2\,000$.
- $1 \leq T \leq 100\,000$.

- $2 \leq L \leq 100\ 000$.
- $1 \leq A_i < B_i \leq N$ $(1 < i \leq M)$.
- $(A_i, B_i) \neq (A_j, B_j)$ $(1 \leq i < j \leq M)$.
- 어떤 급식소에서도 다른 모든 급식소까지 한 개 이상의 도로를 통해 서로 오가는 것이 가능하다.
- $1 \leq C_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq M)$.
- $1 \leq X_j \leq N$ $(1 \leq j \leq L)$.
- $1 \leq P_k \leq L$ $(1 \leq k \leq T)$.
- $1 \leq Q_k \leq N$ $(1 \leq k \leq T)$.
- 모든 j $(1 \leq j \leq L-1)$ 에 대해, $X_j \neq X_{j+1}$을 만족한다. 또한, 보급 계획이 변경된 이후에도 모든 j $(1 \leq j \leq L-1)$ 에 대해, $X_j \neq X_{j+1}$을 만족한다.

서브태스크 1 (12 점)

- $N \leq 10$.
- $M \leq 10$.
- $T = 1$.
- $L \leq 10$.
- $C_i \leq 10$ $(1 \leq i \leq M)$.

서브태스크 2 (35 점)

- $N \leq 500$.
- $M \leq 500$.
- $T = 1$.

서브태스크 3 (15 점)

- $T = 1$.

서브태스크 4 (38 점)

추가 제한조건이 없다.

예제

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

이 입력 예제에서는 최초 보급계획은 1, 2, 3이다. JOI군은 첫 번째 날 아침에 세 번째 값을 1로 바꾼다. 즉, 첫 번째 날의 보급계획은 1, 2, 1이다.

처음에 JOI군은 1번 급식소에서 음식을 보급받을 것이다. 그리고, 1번째 도로를 사용하여, 2번 급식소로 갈 것이다. 그 후 2번 급식소에서 음식을 보급받을 것이다. 그리고, 2번째 도로를 사용하여, 3번 급식소로 갈 것이다. 마지막으로, 3번째 도로를 사용하여, 1번 급식소로 갈 것이다. 그 후 1번 급식소에서 음식을 보급받을 것이다. 이 방법으로 음식을 보급받는 데에는 3시간이 걸린다. 이것이 최소시간이므로 3을 출력한다.

JOI군은 유턴을 할 수 없으므로 $1 \to 2 \to 1$과 같은 순서로 급식소를 방문하는 것은 불가능하다.

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

이 입력 예제에서는 첫 번째 날의 보급계획은 4, 1, 4이다. 처음에 JOI군은 4번 급식소에서 음식을 보급받을 것이다. 그리고, 4번째 도로를 사용하여, 1번 급식소로 갈 것이다. 그 후 1번 급식소에서 음식을 보급받을 것이다. 그 후, 1, 2, 3, 4번째 도로 순서로 사용하여 급식소를 $1 \to 2 \to 3 \to 1 \to 4$ 순서로 움직여, 4번 급식소에서 음식을 보급받을 것이다. 이것이 최소 시간이다.

네 번째 날의 보급계획은 2, 4, 2이다. 이 보급계획은 실행할 수 없기 때문에, -1을 출력한다.

standard input	standard output
5 6 1 5 1 2 8 1 3 8 1 4 8 2 5 2 3 4 6 4 5 6 2 5 1 5 3 5 2	38