

문제 1. 경작

입력 파일: standard input
출력 파일: standard output
시간 제한: 2 seconds
메모리 제한: 256 megabytes

21XX년, IOI 행성의 주민들은 최근 발견된 행성으로 이주하게 되었다.

새로 발견된 행성은 R 행 C 열 격자로 이루어진 밭이 있다. 열은 남-북 방향으로, 행은 동-서 방향으로 놓여있다. 북쪽에서 i 번째, 서쪽에서 j 번째 격자의 번호는 (i, j) 이다. 가장 북서쪽에 있는 격자의 번호는 $(1, 1)$ 이고, 가장 남동쪽에 있는 격자의 번호는 (R, C) 이다. 매년, IOI 행성의 주민들은 밭에 불 바람의 방향을 고른다. 방향은 동, 남, 서 혹은 북이다.

새 행성에서는 농업을 활성화하기 위해서, 밭의 모든 칸에 “JOI 풀”을 심을 것이다. 이주한 첫 연도에는 N 개의 격자에 JOI 풀이 심어져 있다.

JOI 풀은 바람으로 생활권을 늘려간다. 여름마다, JOI 풀의 씨앗이 IOI 행성의 주민들이 고른 바람의 방향으로 날아간다. 씨앗은 원래 JOI 풀이 심어져 있던 곳에서 바람 방향으로 한 칸 움직여 땅에 착지한다. 만약 그 칸에 JOI 풀이 심어져 있지 않다면, 그 칸에는 새로운 JOI 풀이 자란다. 한 격자에 JOI 풀이 심어져 있으면, 그 풀은 영구히 자란다.

바람의 방향을 적당히 설정했을 때, 밭의 모든 칸에 JOI 격자를 심을 수 있으려면 최소 몇 년이 걸리는지 구하고 싶다.

입력 형식

다음 정보가 표준 입력으로 주어진다.

- 첫째 줄에는 공백으로 구분된 두 정수 R, C 가 주어진다. 이는 격자가 R 행 C 열로 이루어져 있다는 의미이다.
- 둘째 줄에는 정수 N 이 주어진다. 이는 이주 첫 연도에 JOI 풀이 심어져 있는 칸의 수이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 공백으로 구분된 두 정수 S_i, E_i 가 주어진다. 이는 이주 첫 연도에 (S_i, E_i) 번 격자에 JOI 풀이 심어져 있다는 뜻이다.

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 출력은 우리가 방향을 적당히 설정했을 때, 모든 격자에 JOI 풀이 심어져 있기 위해 필요한 연수의 최솟값이다.

제한

- $1 \leq N \leq 300$.
- $1 \leq R \leq 1\,000\,000\,000$.
- $1 \leq C \leq 1\,000\,000\,000$.
- $1 \leq S_i \leq R$ ($1 \leq i \leq N$).
- $1 \leq E_i \leq C$ ($1 \leq i \leq N$).
- 이주 첫 연도에 JOI풀이 심어져 있지 않은 격자가 존재한다.

서브태스크 1 (5 점)

- $R \leq 4$

- $C \leq 4$

서브태스크 2 (10 점)

- $R \leq 40$
- $C \leq 40$

서브태스크 3 (15 점)

- $R \leq 40$

서브태스크 4 (30 점)

- $N \leq 25$

서브태스크 5 (20 점)

- $N \leq 100$

서브태스크 6 (20 점)

추가 제한조건이 없다.

예제

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

이 예제에서, 이주 첫 연도에 다음 칸들에 JOI 풀이 심어져 있다.

	0		0
		0	

새 행성의 발. '0'이 쓰여 있는 칸은 이주 첫 연도에 JOI 풀이 심어져 있다.

만약에 처음 3년 동안 바람을 서쪽, 남쪽, 남쪽으로 불어가게 만들면, 모든 격자에는 3년 이후에는 JOI 풀이 심어져 있을 것이다. 다음 숫자는 각 격자에 JOI 풀이 심어진 연도를 의미한다. 이는 최솟값이다.

1	0	1	0
2	1	0	2
3	2	2	3

standard input	standard output
4 4 4 1 1 1 4 4 1 4 4	4

문제 2. 항구 시설

입력 파일: standard input
출력 파일: standard output
시간 제한: 3.5 seconds
메모리 제한: 1024 megabytes

JOI 항구에는 매일 많은 컨테이너가 운반되어 오고, 전국 각지로 트럭을 통해 운반된다.

JOI 항구는 매우 좁아서, 컨테이너를 넣을 수 있는 공간이 2개 밖에 없다. 각각의 공간에는 컨테이너를 수직으로 쌓아서 몇 개든 넣을 수 있다.

안전상의 이유로, 컨테이너가 항구에서 운반되어 오면 두 공간 중 하나에 컨테이너를 놓아야 한다. 만약 이미 컨테이너가 그 위치에 있으면, 이미 있는 컨테이너의 위에 새 컨테이너를 쌓는다. 트럭으로 운반할 때는, 두 공간에 쌓인 컨테이너 중 가장 위에서부터 차례대로 운반해야 한다.

오늘, JOI 항구에는 N 개의 컨테이너가 배로 운반될 예정이다. 모든 컨테이너는 오늘 내로 트럭으로 운반될 예정이다.

당신은 JOI 항구의 항구시설 관리를 맡고 있어서, 모든 컨테이너가 배로 운반되는 시각과 트럭으로 운반되는 시간을 알고 있다. 컨테이너를 쌓고 가져가는 경우의 수를 1 000 000 007로 나눈 나머지를 구하여라.

입력 형식

다음 정보가 표준 입력으로 주어진다.

- 첫째 줄에는 공백으로 정수 N 이 주어진다. 이는 JOI 항구에 운반될 컨테이너의 수이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 공백으로 구분된 두 정수 A_i, B_i 가 주어진다. 이는 JOI 항구에 i 번째 컨테이너가 시각 A_i 에 와서 시각 B_i 에 트럭으로 운반된다는 의미이다.

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 출력은 컨테이너를 쌓고 가져가는 경우의 수를 1 000 000 007로 나눈 나머지이다.

제한

- $1 \leq N \leq 1\,000\,000$.
- $1 \leq A_i \leq 2N$ ($1 \leq i \leq N$).
- $1 \leq B_i \leq 2N$ ($1 \leq i \leq N$).
- $A_i < B_i$ ($1 \leq i \leq N$).
- $2N$ 개의 정수 $A_1, \dots, A_N, B_1, \dots, B_N$ 은 서로 다르다.

서브태스크 1 (10 점)

- $N \leq 20$

서브태스크 2 (12 점)

- $N \leq 2\,000$

서브태스크 3 (56 점)

- $N \leq 100\,000$

서브태스크 4 (22 점)

추가 제한조건이 없다.

예제

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

컨테이너를 놓는 네 가지 방법이 있다. 각 공간을 A, B라고 하자. 다음 방법으로 컨테이너를 놓을 수 있다.

- 1, 2, 3, 4번 컨테이너를 각각 A, B, A, A에 놓는다.
- 1, 2, 3, 4번 컨테이너를 각각 A, B, A, B에 놓는다.
- 1, 2, 3, 4번 컨테이너를 각각 B, A, B, A에 놓는다.
- 1, 2, 3, 4번 컨테이너를 각각 B, A, B, B에 놓는다.

standard input	standard output
3 1 4 2 5 3 6	0
5 1 4 2 10 6 9 7 8 3 5	8
8 1 15 2 5 3 8 4 6 14 16 7 9 10 13 11 12	16

문제 3. 불꽃놀이 막대

입력 파일: standard input
출력 파일: standard output
시간 제한: 2 seconds
메모리 제한: 256 megabytes

JOI군은 자신을 포함하여 N 명의 친구들과 불꽃놀이 막대를 가지고 놀 것이다. 이번에 사용할 불꽃놀이 막대는 불을 붙이면 정확히 T 초 동안 불이 붙는다.

처음에 JOI군과 친구들은 동서 방향으로 일직선으로 서 있고, 한 사람당 하나의 불꽃놀이 막대를 들고 있다. JOI군과 친구들은 각각 1 이상 N 이하의 번호가 붙어있다. $i < j$ 를 만족하는 i 와 j 에 대해서, i 번째 사람은 j 번째 사람의 서쪽에 서 있거나, 같은 장소에 서 있다. i 번째 사람과 가장 서쪽에 있는 첫 번째 사람의 거리는 X_i 미터이다. JOI군은 K 번째 사람이다.

불꽃놀이를 시작하려 할 때, 라이터의 연료가 충분하지 않다는 사실을 알았다. 오직 하나의 불꽃놀이 막대에만 불을 붙일 수 있다.

그래서 일단 JOI군의 불꽃놀이 막대에 불을 붙이고, 타고 있는 불꽃놀이 막대의 불을 옮겨가면서 불을 붙이기로 했다. 불꽃놀이 막대에서 불을 옮길 때는, 다음 조건을 만족해야 한다.

- 불이 붙지 않은 불꽃놀이 막대를 불을 붙인 지 T 초 이내의 불꽃놀이 막대와 맞닿아야 한다. 불을 붙인 지 정확히 T 초가 지나도 불을 옮길 수 있다.
- 불을 붙이려는 불꽃놀이 막대는, 한 번도 불이 붙은 적이 없어야 한다.
- 불이 붙지 않은 불꽃놀이 막대와 불이 붙은 불꽃놀이 막대를 가진 사람이 같은 장소에 있어야 한다.

우리는 한 불꽃놀이 막대에서 다른 불꽃놀이 막대로 불이 붙기를 기다리는 시간 등을 무시할 것이다.

JOI군과 친구들이 서로 떨어져 서 있기 때문에, 불을 붙이기 위해서는 잘 이동해야 한다. 그들은 임의의 속도로 달릴 수 있지만, 불꽃놀이를 하는 중 달리면 위험하므로, 속도가 초당 s 미터를 넘지 않게 하고 싶다. 여기서, s 는 음이 아닌 정수이다.

모든 불꽃놀이 막대에 불을 붙이기 위해서 속도 제한을 어떻게 정하는 게 좋을까?

입력 형식

다음 정보가 표준 입력으로 주어진다.

- 첫째 줄에는 공백으로 구분된 세 정수 N , K , T 가 주어진다. 이는 N 명의 사람이 있고, JOI군이 K 번째 사람이고, 막대 불꽃놀이에 불을 붙이면 T 초 동안 불이 붙어있다는 것을 말한다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 공백으로 정수 X_i 가 주어진다. 이는 i 번째 사람과 가장 서쪽에 있는 첫 번째 사람의 거리는 X_i 미터라는 것을 말한다.

출력 형식

표준 출력으로 한 개의 줄을 출력하여야 한다. 출력은 모든 불꽃놀이에 불을 붙이기 위한 음이 아닌 정수 속도 제한 s 이다.

제한

- $1 \leq K \leq N \leq 100\,000$.
- $1 \leq T \leq 1\,000\,000\,000$.
- $1 \leq X_i \leq 1\,000\,000\,000$ ($1 \leq i \leq N$).

- $X_1 = 0$
- $X_i \leq X_j$ ($1 \leq i \leq N$).

서브태스크 1 (30 점)

- $N \leq 20$

서브태스크 2 (20 점)

- $N \leq 1\,000$

서브태스크 3 (50 점)

추가 제한조건이 없다.

예제

standard input	standard output
3 2 50 0 200 300	2

이 예제에서, 속도 제한은 초당 2 미터여도 된다.

첫 번째 사람이 동쪽으로, 두 번째와 세 번째 사람이 서쪽으로 움직인다. 속도는 초당 2 미터이다. 50초 이후에, 두 번째 사람은 첫 번째 사람에게 불을 옮길 수 있다.

그리고, 첫 번째 사람이 동쪽으로, 세 번째 사람이 서쪽으로 움직인다. 속도는 초당 2 미터이다. 25초 이후에, 첫 번째 사람은 세 번째 사람에게 불을 옮길 수 있다.

속도 제한이 1 미터였다면, 모든 막대에 불을 붙일 수 없다.

standard input	standard output
3 2 10 0 200 300	8

이 예제에서, 속도 제한은 초당 8 미터여도 된다.

첫 번째와 두 번째 사람이 동쪽으로, 세 번째 사람이 서쪽으로 움직인다. 속도는 초당 8미터이다.

3초 후에, 두 번째 사람이 움직임을 멈춘다. 첫 번째와 세 번째 사람은 계속 움직인다.

6.5초가 더 지난 이후에, 두 번째 사람과 세 번째 사람이 같은 장소에 모인다. 두 번째 사람과 세 번째 사람이 움직임을 멈춘다. 첫 번째 사람은 계속 움직인다.

0.5초가 더 지난 이후에, 두 번째 사람은 세 번째 사람에게 불을 옮긴다. 첫 번째 사람은 계속 움직인다. 세 번째 사람은 서쪽으로 움직인다. 속도는 초당 8 미터이다.

9초가 더 지난 이후에, 첫 번째 사람과 세 번째 사람이 같은 장소에 모인다. 세 번째 사람은 첫 번째 사람에게 불을 옮긴다.

속도 제한이 7 미터였다면, 모든 막대에 불을 붙일 수 없다.

standard input	standard output
20 6 1 0 2 13 27 35 46 63 74 80 88 100 101 109 110 119 138 139 154 172 192	6

문제 4. 티켓 정리

입력 파일: standard input
출력 파일: standard output
시간 제한: 4 seconds
메모리 제한: 256 megabytes

JOI 공화국에는 1번 부터 N 번 까지 번호가 붙은 N 개의 기차역이 있다. 기차역은 원형 철도에 시계방향으로 위치해있다.

철도를 이용하기 위한 티켓은 N 종류가 있고, 각각 1번부터 N 번까지 번호가 붙어 있다. i 번 ($1 \leq i \leq N-1$) 티켓을 하나 사용하면, i 번 기차역에서 $i+1$ 번 기차역으로, 혹은 $i+1$ 번 기차역에서 i 번 기차역으로 사람 한 명이 이동할 수 있다. N 번 티켓을 하나 사용하면, 1번 기차역에서 N 번 기차역으로, 혹은 N 번 기차역에서 1번 기차역으로 사람 한명이 이동할 수 있다. 이 티켓은, N 종류의 티켓이 정확히 한 장씩 총 N 장이 들어있는 묶음으로 판매되고 있다.

당신은 JOI 공화국의 여행회사에서 일하고 있다. 당신은 고객들에게 티켓을 나눠주어야 한다.

오늘 티켓을 나눠달라는 M 개의 요청이 있었다. i 번째 요청은 C_i 명의 사람이 A_i 번 기차역에서 B_i 번 기차역으로 이동하고 싶다는 요청이었다. C_i 명의 사람들이 모두 같은 경로로 이동 할 필요는 없다.

모든 요청을 처리하기 위해서 사야 할 묶음의 최소 갯수를 알고 싶다.

입력 형식

다음 정보가 표준 입력으로 주어진다.

- 첫째 줄에는 공백으로 구분된 두 정수 N, M 이 주어진다. JOI 공화국에 N 개의 기차역이 있으며, M 개의 요청을 오늘 받았다는 것이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 공백으로 구분된 세 정수 A_i, B_i, C_i 가 주어진다. 이는 i 번째 요청이 C_i 명의 사람이 A_i 번 기차역에서 B_i 번 기차역으로 이동하고 싶다는 요청이라는 것을 의미한다.

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 출력은 모든 요청을 처리하기 위해서 사야 할 묶음의 최소 갯수이다.

제한

- $3 \leq N \leq 200\,000$.
- $1 \leq M \leq 100\,000$.
- $1 \leq A_i \leq N$ ($1 \leq i \leq M$).
- $1 \leq B_i \leq N$ ($1 \leq i \leq M$).
- $1 \leq C_i \leq 1\,000\,000\,000$ ($1 \leq i \leq M$).
- $A_i \neq B_i$ ($1 \leq i \leq M$)

서브태스크 1 (10 점)

- $N \leq 20$
- $M \leq 20$
- $C_i = 1$ ($1 \leq i \leq M$)

서브태스크 2 (35 점)

- $N \leq 300$
- $M \leq 300$
- $C_i = 1$ ($1 \leq i \leq M$)

서브태스크 3 (20 점)

- $N \leq 3\,000$
- $M \leq 3\,000$
- $C_i = 1$ ($1 \leq i \leq M$)

서브태스크 4 (20 점)

- $C_i = 1$ ($1 \leq i \leq M$)

서브태스크 5 (15 점)

추가 제한조건이 없다.

예제

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

모두가 시계방향으로 이동하면, 각 종류의 티켓이 하나씩 필요하다. 즉, 한 묶음만 사도 충분하다.

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

다음 방법으로 이동하면 각 종류의 티켓이 세 장씩 필요하다:

- 첫 번째 요청에서, 세 명이 시계방향으로, 한 명이 반시계방향으로 움직인다.
- 두 번째 요청에서, 두 명이 반시계방향으로 움직인다.

그래서, 세 묶음을 사면 충분하다.

두 묶음을 사서 이동하는 것은 불가능하기 때문에, 3을 출력한다.

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

예를 들면 두 묶음을 사서 다음과 같이 나누어 주면 된다.

- 1번 역에서 4번 역으로 이동하고 싶은 사람에게 1, 2, 3번 티켓을 준다.
- 2번 역에서 5번 역으로 이동하고 싶은 사람에게 1, 6, 5번 티켓을 준다.
- 3번 역에서 6번 역으로 이동하고 싶은 사람에게 3, 4, 5번 티켓을 준다.

한 묶음을 사서 이동하는 것은 불가능 하기 때문에, 2를 출력한다.

문제 5. 고장난 기기

시간 제한: 2 seconds
메모리 제한: 256 megabytes

고고학자 Anna와 Bruno는 이란의 유적을 조사하고 있다.

두명이 역할을 분할하여 Anna는 유적을 발견하고 유물을 발견하며, Bruno는 베이스캠프에서 결과를 분석한다.

조사는 총 $Q(= 1\,000)$ 일 동안 계획이 되어있다. 매일, Anna는 Bruno에게 통신 기기를 사용하여 결과를 보낸다. 각 통신기기의 결과는 정수 X 로 표시된다.

안나는 통신 기기를 하루에 한번만 사용할 수 있다. 통신기기는 0 혹은 1로 되어있는 길이 $N(= 150)$ 의 수열을 보낼 수 있다.

하지만 이 기계가 고장나 버려서, 보내는 길이 N 의 수열 중 기능이 제대로 작동하지 않는 위치가 생겨버렸다. 기능하지 않는 위치에는, 어떤 값을 설정해도 0이 보내지게 된다. Anna가 수열을 보낼 때, 고장난 위치가 어딘지 알 수 있다. 하지만, Bruno는 그 위치를 모른다. 고장난 위치가 몇 개인지, 어딘지는 매일 바뀐다.

이 조사가 지연되면 문제가 생길 수 있다. Anna와 Bruno는 이란에서 열리는 국제 프로그래밍 대회의 후보인 당신에게 조사 결과를 보내는 프로그램을 작성해달라고 요청했다.

당신은 Anna와 Bruno 사이에서 통신을 하는 두 개의 프로그램을 작성하여야 한다.

- 수열의 길이 N 과 보내야 할 정수 X 와, 고장난 위치의 갯수 K 와, 고장난 위치 P 가 주어졌을 때, 첫번째 프로그램은 Anna가 보낼 수열 S 를 정한다.
- Bruno가 수열 A 를 받았을 때, 두 번째 프로그램은 정수 X 를 복구한다.

통신 장치가 고장난 위치가 아닌 곳에서는, 수열 S 와 수열 A 는 같은 값을 가진다. 통신 장치가 고장난 곳에서는, 수열 A 는 수열 S 의 값과 관계 없이 0이다.

구현 명세

당신은 같은 프로그래밍 언어로 작성된 파일 두개를 작성해야 한다.

첫 번째 파일의 이름은 `Anna.c` 혹은 `Anna.cpp`이다. 이 파일은 Anna가 보낼 수열을 정하는 역할을 하며, 다음 함수를 구현해야 한다. 이 파일은 `Annalib.h`를 include해야 한다.

- `void Anna(int N, long long X, int K, int P[])`
이 함수는 각 테스트 케이스 마다 정확히 $Q = 1\,000$ 번 불린다.
 - 인자 N 은 보낼 수열의 길이를 나타낸다.
 - 인자 X 는 보낼 숫자를 나타낸다.
 - 인자 K 는 부서진 위치의 갯수를 나타낸다.
 - 인자 $P[]$ 는 부서진 위치를 나타내는 길이 K 의 수열이다.

함수 `Anna`는 다음 함수를 호출해야 한다.

- `void Set(int pos, int bit)`
이 함수는, 통신 기기로 보낼 수열 S 를 설정한다.
 - * 인자 pos 는 수열의 값을 설정할 위치이다. pos 는 0 이상 $N - 1$ 이하이다. 위치가 0부터 시작함에 유의하여라. 만약에 이 범위를 벗어나서 함수를 호출 한 경우, **오답 [1]**이 된다. 같은 pos 를 인자로 하여 함수를 두 번 이상 호출 한 경우, **오답 [2]**이 된다.
 - * 인자 bit 는 pos 번째에 설정할 값이다. bit 의 값은 0 혹은 1이어야 한다. 다른 인자로 함수를 호출 한 경우 **오답 [3]**이 된다.

함수 `Set`은 함수 `Anna` 안에서 정확히 N 번 호출되어야 한다. `Anna` 함수가 종료되었을 때, `Set`이 호출된 횟수가 N 과 다르면, **오답 [4]**이 된다.

만약 `Anna`가 함수를 올바르게 호출 될 경우 프로그램이 종료된다.

첫 번째 파일의 이름은 `Bruno.c` 혹은 `Bruno.cpp`이다. 이 파일은 탐사 결과를 복구하는 역할을 하며, 다음 함수를 구현해야 한다. 이 파일은 `Brunolib.h`를 `include`해야 한다.

- `long long Bruno(int N, int A[])`

이 함수는 각 테스트 케이스마다 정확히 $Q = 1\,000$ 번 불린다.

- 인자 N 은 Bruno가 받은 수열의 길이를 나타낸다.
- 인자 $A[]$ 는 Bruno가 받은 길이 N 의 수열이다.
- 함수 Bruno는 X 를 찾아서 반환해야 한다.

채점은 다음과 같은 방식으로 진행된다. 만약 프로그램이 오답으로 판단된 경우, 즉시 채점은 종료된다.

1. `cnt=0`으로 설정한다.
2. 함수 `Anna`를 1회 호출한다.
3. 함수 `Anna`에 호출된 수열을 S 라고 하자. S 중 P 에 포함된 위치를 0으로 바꾸는 작업을 A 에 한 후, 함수 `Bruno`를 1회 호출한다.
4. `cnt=cnt+1`로 설정한다. `cnt<Q`이면 2.로 돌아간다. `cnt=Q`이면, 5.로 간다.
5. 채점을 한다.

참고 사항

- 실행 시간과 메모리 사용량은 채점 방식의 1, 2, 3, 4에서 계산된다.
- 당신의 프로그램은 채점 방식 2.의 `Anna` 혹은 채점 방식 3.의 `Bruno`에서 오답으로 판단되면 안된다. 당신의 프로그램은 런타임 에러 없이 실행되어야 한다.
- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다. 제출한 프로그램은 그레이더와 함께 컴파일 되어 하나의 실행파일이 된다. 모든 전역변수나 내부 함수는 다른 파일과의 충돌을 피하기 위해 `static`으로 선언되어야 한다. `Anna`와 `Bruno`는 2개의 별개의 프로세스로 실행되기 때문에 채점 될 때 전역변수를 공유하지 않는다.
- 각 프로세스에서, `Anna`와 `Bruno`는 각각 $Q = 1\,000$ 번 호출된다. 사용할 변수의 초기화는 적절히 진행되어야 한다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안된다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트 하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 `grader.c` 혹은 `grader.cpp`이다. 당신의 프로그램이 `Anna.c`와 `Bruno.c` 혹은, `Anna.cpp`와 `Bruno.cpp` 인 경우 다음 커맨드로 컴파일할 수 있다.

- `C g++ -std=c11 -O2 -o grader grader.c Anna.c Bruno.c -lm`
- `C++ g++ -std=c++14 -O2 -o grader grader.cpp Anna.cpp Bruno.cpp`

컴파일이 성공적이면, 파일 `grader`가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여야. 샘플 그레이더는 하나의 프로세스에서 실행되며, 입력을 표준 입력으로 부터 받고, 출력을 표준 출력에 출력한다.

입력 형식

샘플 그레이더는 다음 형식으로 표준 입력으로 부터 데이터를 입력받는다.

- 첫째 줄에 정수 Q 가 주어진다.
- 그리고, Q 개의 쿼리의 정보가 주어진다.
- 각 쿼리의 정보는 다음과 같은 두 줄로 되어있다.
 - 첫 번째 줄은 공백으로 구분된 세 정수 N, X, K 가 주어진다. 이는 보낼 수열의 길이가 N 이고, Anna가 보낼 정수가 X 고, 부서진 위치가 K 개라는 뜻이다.
 - 두 번째 줄은 공백으로 구분된 K 개의 정수 P_0, P_1, \dots, P_{K-1} 이 주어진다. 이는, 각 i ($0 \leq i \leq K-1$)에 대해, 수열의 P_i 번째 위치가 고장났다는 것이다.

출력 형식

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

- 오답으로 판단 된 경우, 오답의 종류를 “Wrong Answer [1]”과 같은 형식으로 출력하고, 프로그램이 종료된다.
- 만약 모든 Anna의 호출이 오답으로 판단되지 않을 경우, “Accepted”와 L^* 을 표준 출력으로 출력한다. L^* 의 값은 배점 항목을 참고하여야.

프로그램이 다양한 오답의 종류에 속해 있을 경우, 샘플 그레이더는 그 중 하나만 출력 할 것이다.

제한

- $Q = 1000$.
- $N = 150$.
- $0 \leq X \leq 1\,000\,000\,000\,000\,000\,000$.
- $1 \leq K \leq 40$.
- $0 \leq P_i \leq N-1$ ($0 \leq i \leq K-1$).
- $P_i < P_{i+1}$ ($0 \leq i \leq K-2$).

배점

- L^* 를 이 문제의 모든 테스트 케이스의 최솟값이라고 하자.
 - $K \leq L$ 인 모든 쿼리에 대해서, Bruno가 정답을 말한 최대 정수 $L \leq 40$.
- 이 문제의 점수는 다음과 같이 계산된다.
 - $L^* = 0$ 인 경우, 점수는 0점이다.
 - $1 \leq L^* \leq 14$ 인 경우, 점수는 8점이다.

- $15 \leq L^* \leq 37$ 인 경우, 점수는 $(L^* - 15) \times 2 + 41$ 점 이다.
- $38 \leq L^* \leq 40$ 인 경우, 점수는 $(L^* - 38) \times 5 + 90$ 점 이다.

예제

예제 입력과 이에 해당하는 함수 호출을 보여준다. 이 예제는 $Q = 2, N = 3$ 이기 때문에 문제의 제한을 만족하지 않음에 유의하여야.

예제 입력	예제 함수 호출			
	호출	반환값	호출	반환값
2 3 14 1 2 3 9 2 0 1	Anna(...)			
			Set(0,0)	
				(없음)
			Set(1,0)	
				(없음)
			Set(2,1)	
				(없음)
		(없음)		
	Bruno(...)			
		14		
	Anna(...)			
			Set(0,0)	
				(없음)
			Set(1,1)	
				(없음)
			Set(2,1)	
				(없음)
		(없음)		
	Bruno(...)			
		9		

여기서 Anna(...), Bruno(...), Anna(...), Bruno(...) 호출의 인자들은 다음과 같다.

인자	Anna(...)	Bruno(...)	Anna(...)	Bruno(...)
N	3	3	3	3
K	14		9	
X	1		2	
P	{2}		{0, 1}	
A		{0, 0, 0}		{0, 0, 1}

문제 6. 철도 여행

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

JOI 전철은 철도 하나를 운영하고 있는 회사이다. JOI 전철의 철도에는 1번부터 N 번까지의 번호가 붙어있는 N 개의 철도역이 일직선상에 놓여있다. 각 i ($1 \leq i \leq N-1$)에 대해서, i 번 역과 $i+1$ 번 역은 선로로 연결되어있다.

JOI전철은 양방향으로 달리는 K 종류의 열차가 있다. 열차의 종류는 1이상 K 이하의 정수로 표현된다. 각 역에는 **중요도**라고 불리는 1이상 K 이하의 정수가 하나씩 붙어 있다. i 번 ($1 \leq i \leq N$) 역의 중요도는 L_i 이다. 양 끝에 있는, 즉 1번과 N 번 역은 중요도가 K 이다.

j 번 ($1 \leq j \leq K$) 종류의 열차는 중요도가 j 이상인 철도역에서만 멈추고, 다른 철도역에서는 멈추지 않는다. 양 끝에 있는 1번과 N 번 철도역은 중요도가 K 이기 때문에 모든 열차는 이 철도역에서는 멈춘다.

많은 승객들이 JOI 전철을 이용하고 있다. 여행 중에 승객들은 목적지와 반대방향으로 움직이거나, 목적지를 통과 할 수도 있지만, 결국에는 목적지에 멈추어야 한다. 승객들은 역에 멈추는것을 좋아하지 않는다. 그렇기 때문에 도중에 정차하는 철도역의 수를 최소화 하고 싶다. 어떤 승객이 열차를 갈아타기 위해 역에 멈춘 경우에도 한 번으로 계산하고, 출발역과 도착역은 세지 않는다.

당신의 업무는 승객의 출발역과 도착역이 주어졌을 때, 이 두 역의 사이를 이동하는 도중에 정차하는 철도역의 수를 구하는 프로그램을 작성해야 한다.

입력 형식

다음 정보가 표준 입력으로 주어진다.

- 첫째 줄에는 공백으로 구분된 세 정수 N, K, Q 가 주어진다. 이는 JOI 전철에는 N 개의 역이 있고, K 종류의 열차가 있으며, 질문의 갯수가 Q 개라는 것이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 L_i 가 주어진다. 이는 i 번 역의 중요도가 L_i 라는 것이다.
- 다음 Q 개의 줄의 k 번째 ($1 \leq k \leq Q$) 줄에는 공백으로 구분된 두 정수 A_k, B_k 가 주어진다. 이는 k 번째의 승객의 출발역이 A_k 번 역이고, 도착역이 B_k 번 역이라는 것을 의미한다.

출력 형식

표준 출력으로 Q 개의 줄을 출력하여야 한다. k 번째 ($1 \leq k \leq Q$) 줄에는 A_k 번 역에서 B_k 번 역으로 이동 할 때, 도중에 정차하는 철도역의 최소값을 출력하여야 한다.

제한

- $2 \leq N \leq 100\,000$.
- $1 \leq K \leq N$.
- $1 \leq Q \leq 100\,000$.
- $1 \leq L_i \leq K$ ($1 \leq i \leq N$).
- $1 \leq A_k \leq N$ ($1 \leq k \leq Q$).
- $1 \leq B_k \leq N$ ($1 \leq k \leq Q$).
- $A_k \neq B_k$ ($1 \leq k \leq Q$).

서브태스크 1 (5 점)

- $N \leq 100$
- $K \leq 100$
- $Q \leq 50$

서브태스크 2 (15 점)

- $Q \leq 50$

서브태스크 3 (25 점)

- $K \leq 20$

서브태스크 4 (55 점)

추가 제한조건이 없다.

예제

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

이 예제에서, 질문은 세 가지가 있다.

- 첫 번째 질문은, 2번 역에서 4번 역까지 이동하는 것이다. 이 때, 2번 역에서 4번 역까지 1번 종류의 열차를 이용하면 도중에 정차하는 역은 3번 역 하나 뿐이 된다.
- 두 번째 질문은, 4번 역에서 9번 역까지 이동하는 것이다. 이 때, 우선 4번 역에서 5번 역까지 1번 종류의 열차를 이용하고, 다음에 5번 역에서 1번 역까지 2번 종류의 열차를 이용하고, 마지막으로 1번 역에서 9번 역까지 3번 종류의 열차를 이용하면 도중에 정차하는 역은 5번 역, 1번 역, 8번 역의 셋이 된다.
- 세 번째 질문은, 6번 역에서 7번 역까지 이동하는 것이다. 이 때, 6번 역에서 7번 역까지 2번 종류의 열차를 이용하면 도중에 정차하는 역 없이 이동하는 것이 가능하다.

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

도중에 목적지가 있는 역을 지나쳐도 되는 점에 주의하여라.

standard input	standard output
15 5 15	2
5	1
4	1
1	3
2	2
3	0
1	3
1	4
2	0
4	1
5	3
4	4
1	1
5	2
3	2
5	
8 1	
11 1	
5 3	
6 11	
9 12	
15 14	
15 2	
3	
12	
2 1	
4 8	
15 5	
12 6	
1 13	
13 8	
14 9	