문제 1. 사탕

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	5 seconds
메모리 제한:	512 megabytes

책상 위에 N개의 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 몇개를 먹기로 결심했다. JOI양은 자신이 먹는 사탕의 맛의 합을 최대화 하고 싶다. 하지만, JOI양은 사탕을 평범하게 선택하는 것이 재미 없다고 생각한 JOI양은 연속된 두개의 사탕을 고르면 안된다는 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 N개의 사탕이 있다는 의미이다.
- 다음 N개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 A_i가 주어진다. 이는 왼쪽에서 i 번째 사탕의 맛이 A_i 임을 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 j개의 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 왼쪽에서 부터 각각 3, 5, 1, 7, 6이다.

JOI양은 다음과 같이 사탕을 먹는다.

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

연속된 두개의 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 2. 도서관

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

수백년의 시간이 지난 끝에, JOI도시는 폐허가 되었다. 탐험가 IOI양은, 도서관이 있었던 지역을 조사하고 있다. 조사를 하면, 다음과 같은 것들을 알 수 있다.

- JOI시의 도서관 서재에는 N 권의 책이 있었다. N 권의 책은 책장에 왼쪽에서 오른쪽으로 일렬로 놓여 있었다.
- N 권의 책은 1번 부터 N번 까지의 번호가 붙어있다. 하지만 책장에 꽂힌 순서와 번호가 붙은 순서는 다를 수도 있다.
- 작업 한번으로, 책장에서 연속으로 놓여진 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 기계한테 질문을 하면서 책이 위치한 순서를 알고 싶어한다. 단, N개의 책이 역순으로 놓여있는 경우에도 답은 변하지 않으므로, 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경쓰지 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 케이스 마다 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 기계한테 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 신경쓰지 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 안된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

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.cpp이다. 당신의 프로그램을 테스트 하기 위해서, grader.cpp, library.cpp, library.h를 같은 디렉토리 안에 놓고, 컴파일 하기 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 실행 되며, 입력을 표준 입력으로 부터 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 N개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 A_i가 주어진다. 이는 왼쪽에서 i 번째 꽂힌 책의 번호가 A_i 임을 의미한다.

출력 형식

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

- 정답으로 판단 된 경우, Query함수의 호출 횟수를 "`Accepted: 100`"과 같은 형식으로 출력한다.
- 오답으로 판단 된 경우, 오답의 종류를 "`Wrong Answer [1]`"과 같은 형식으로 출력한다.

프로그램이 다양한 오답의 종류에 속해 있을 경우, 샘플 그레이더는 그 중 하나만 출력 할 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 신경쓰지 않는다. 그렇기 때문에, 배열을 거꾸로 쓴 `Answer({1,3,5,2,4})`를 호출하여도 정답이다.

문제 3. 멧돼지

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	10 seconds
메모리 제한:	1024 megabytes

멧돼지 JOI군이 살고 있는 IOI숲에는 N개의 야생동물 급식소가 있고, M개의 도로가 있다. 급식소에는 1번부터 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 못한기 때문에, 도로를 가던 도중 유턴해서 출발했던 급식소로 돌아오지 못한다. 또한, 급식소에 도착한 이후에는, 가장 최근에 사용한 도로를 써서 되돌아가는 것도 하지 못한다.

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 여러번 등장할 수는 있지만, 모든 j ($1 \leq j \leq L-1$) 에 대해, $X_j \neq X_{j+1}$을 만족한다. 보급계획 중에는 불가능 한게 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 IOI숲에 N개의 급식소와 M개의 도로가 있으며, JOI군은 T 일 동안의 보급계획을 생각 중이고, 각 보급계획이 L개의 급식소 번호를 차례로 나열했다는 의미이다.
- 다음 M개의 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 L개의 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 T개의 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_k번째 값을 Q_k로 바꾼다는 의미이다.

출력 형식

표준 출력으로 T개의 줄을 출력하여라. k 번째 ($1 \leq k \leq T$) 줄은 k번째 보급계획이 불가능 하면 `-1`, 가능하면 보급계획을 실행하는데 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 급식소 까지 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 보급계획을 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