

2019

Korea Univ. Collegiate Programming Contest

일반 부문

대회가 시작되기 전까지 절대 표지를 넘기지 마세요.

고려대학교 SW중심대학사업단

NAVER D²



대회 규칙

- 사용 가능한 언어는 C, C++, Java, Python 3, PyPy 3입니다.
 - 모든 문제에 대해 제약 조건을 만족하며 정답을 출력하는 C++17 코드가 있음이 보장됩니다.
- 대회는 대회 전용 DOMjudge 사이트에서 치뤄지며 문제, 채점 실시간 정보 등을 확인할 수 있습니다.
- 순위는 평균 문제가 많은 순서대로, 평균 문제 수가 같을 경우에는 패널티의 합이 낮은 순으로 정렬됩니다.
 - 문제별 패널티는 '(문제를 풀기까지 걸린 시간(분)) + (그 전까지 제출한 횟수) × 10'입니다.
 - 컴파일 에러는 제출 횟수에 포함되지 않습니다.

금지 / 제한 행위

- 대회가 진행되는 동안 화장실 등을 다녀오는 것은 자유이나, 충분히 이동은 제한됩니다.
- 대회 중도 퇴실은 불가합니다.
- 컴퓨터를 두 대 이상 사용하는 것을 금합니다.
- 운영진에게 질문하는 것 외에 다른 사람과 대화하는 것을 금합니다.
- 사전에 코드를 미리 작성해 와서 사용하는 것을 금합니다.
- 허용된 레퍼런스 페이지를 제외한 메신저, 인터넷 검색, 대화, 이동식 저장 매체를 통한 문제 풀이를 금합니다.
- 문제 제출을 비정상적으로 많이 시도하거나, 의도적으로 대회 웹 서버를 공격하는 행위를 금합니다.

대회 규칙을 어기거나, 운영진이 판단하기에 부정한 행위를 저지를 경우 경고 없이 대회 참가 자격이 박탈될 수 있습니다.

레퍼런스 사이트

다음 레퍼런스 사이트는 열람할 수 있습니다.

- C/C++ : <https://en.cppreference.com/w/>, <http://cplusplus.com>
- Java : <https://docs.oracle.com/en/java/javase/11/docs/api/index.html>
- Python : <https://docs.python.org/3/>

채점 서버 환경

채점 서버의 플랫폼은 **DOMjudge**이며, 운영체제는 **Debian 10** (linux kernel 5.3.0-23-generic)입니다.

컴파일 옵션 및 시간 보정

사용 가능한 언어와 컴파일 옵션 및 제한 시간 보정은 다음과 같습니다. "\$@"는 업로드한 코드 및 생성된 프로그램 이름입니다.

- **C11** (gcc 8.3.0)

컴파일 : gcc -x c -Wall -O2 -std=c11 -static -pipe -o "\$DEST" "\$@" -lm

실행 : exec "\$@"

- **C++17** (g++ 8.3.0)

컴파일 : g++ -x c++ -Wall -O2 -std=c++17 -static -pipe -o "\$DEST" "\$@" -lm

실행 : exec "\$@"

- **Java** (Java 11.0.4)

컴파일 : javac -encoding UTF-8 -sourcepath . -d . "\$@" 2> "\$TMPFILE"

실행 : java -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss\${MEMSTACK}k -Xms\${MEMLIMITJAVA}k \ -Xmx\${MEMLIMITJAVA}k '\$MAINCLASS' '\\$@'

– MEMSTACK은 65536이며, MEMLIMIT은 문제의 제한에서 128MB를 뺀 값입니다.

시간 : 3배

- **Python 3** (Python 3.7.3)

컴파일 : python3 -m py_compile "\$@"

실행 : python3 "\$@"

시간 : 4배

- **PyPy 3** (PyPy 7.0.0 with GCC 8.2.0 (Python 3.5.3))

컴파일 : python3 -m py_compile "\$@"

실행 : pypy3 "\$@"

시간 : 4배

DOMjudge 채점

DOMjudge에 코드를 업로드할 때는 다음 조건을 지켜야 합니다.

- 파일 이름은 알파벳 및 숫자로 시작해야 하며, 알파벳 대소문자 / 숫자 / +.-만 사용 가능합니다.
- 확장자는 C는 .c, C++는 .cpp / cc / cxx / c++, Java는 .java, Python은 .py / .py3여야 합니다.
- 제출한 코드는 표준 입출력만으로 통신하여야 합니다 (파일 입출력은 금지됩니다).
- 제출한 소스코드의 크기는 256 MiB 이하여야 합니다.
- 한 문제에 제출은 최대 100번 할 수 있습니다.

DOMjudge 채점 결과

Submit을 한 다음에 Scoreboard 탭에서 제출 결과를 확인할 수 있습니다.

- PENDING : 제출되었으며, 채점 대기중이거나 채점중입니다.
- CORRECT : 제출한 코드가 모든 테스트 케이스에 대해 시간 제한 / 메모리 제한 내에서 올바른 답을 내었고, 정상적으로 종료되었습니다. 이 경우 제출자는 해당 문제를 **풀었습니다**.
- COMPILER-ERROR : 컴파일 과정 중에 에러가 발생하여 채점이 진행되지 않았습니다.
- TIMELIMIT : 프로그램 수행 시간이 제한 시간을 초과하였습니다.
- RUN-ERROR : 프로그램 수행 중 에러가 발생하였습니다. (예시 : 0으로 나누기, 잘못된 주소 참조)
- WRONG-ANSWER : 프로그램이 오답을 출력하였습니다.
- OUTPUT-LIMIT : 프로그램이 지나치게 많은 출력을 하였습니다.

CORRECT가 아닌 결과가 나온 테스트 케이스가 최초로 등장하면, 해당 채점 결과가 제출 결과로 보여집니다.

대회 중 ‘request clarification’ 탭을 통해 주최진에게 질문을 물을 수 있습니다.

문제 목록

대회는 3시간 동안 진행되며, 총 7문제로 구성되어 있습니다.

총 문제지가 표지를 제외하고 12쪽인지 확인하시길 바랍니다.

문제의 목록은 다음과 같으며, 출제진이 판단하기에 난이도 순으로 정렬되어 있습니다.

- A 함정에 빠진 이동관 (trap)
- B 대자보 (poster)
- C 신앙 (faith)
- D 얼마나 예뻐? (pretty)
- E Sum without carries (nocarry)
- F 서버 증축 (server)
- G 기묘한 여행계획 2 (trip)

문제 A. 함정에 빠진 이동관 (trap)

시간 제한: 2초

메모리 제한: 512 MB

이동관은 $N \times M$ 격자 모양의 함정에 빠졌다. 모든 격자에는 숫자가 적혀있는 트램펄린이 존재하고, 트램펄린을 통해서만 다른 격자로 이동할 수 있다.

트램펄린의 이동 규칙은 다음과 같다.

- 함정 밖으로 이동할 수는 없다. 동관이의 위치는 (a, b) 는 항상 $1 \leq a \leq N, 1 \leq b \leq M$ 를 만족해야 하며, 이 조건을 만족하는 위치로만 이동할 수 있다.
- 현재 위치의 트램펄린에 x 가 적혀 있다면, 상하좌우로 x 칸 떨어진 곳으로만 이동할 수 있다. 즉, 현재 위치가 (a, b) 라면, $(a+x, b), (a-x, b), (a, b+x), (a, b-x)$ 이며 위의 규칙을 만족할 때 이동할 수 있다.
- 만약 어느 칸으로도 이동할 수 없으면 영원히 그 위치에 있게 된다.

한 번 이동하는데 1만큼의 시간이 걸린다고 할 때, 탈출구까지 도달하는데 걸리는 최단 시간을 구하여라. 이동관의 출발 위치는 항상 $(1, 1)$ 이다.

입력 형식

첫 번째 줄에 함정의 행의 개수 N 과 열의 개수 M 이 공백으로 구분되어 주어진다. ($1 \leq N, M \leq 1,000$)

이후 N 개의 줄에 걸쳐 M 개의 자연수 a_1, a_2, \dots, a_m 이 공백으로 구분되어 주어진다. ($1 \leq a_j \leq \min(N, M)$) $i+1$ 번째 줄의 a_j 는 (i, j) 에 있는 트램펄린에 적혀 있는 수이다.

마지막 줄에는 두 자연수 x, y 가 공백으로 구분되어 주어진다. ($1 \leq x \leq N, 1 \leq y \leq M$) 이는 탈출구가 (x, y) 에 존재함을 의미한다. 출발 위치가 탈출구일 수도 있다.

출력 형식

첫 번째 줄에 동관이가 탈출할 수 있다면 탈출하는데 걸리는 시간을, 없으면 -1을 출력한다.

예제

표준 입력	표준 출력
3 3 2 1 2 1 1 1 2 1 2 2 2	-1
2 3 1 1 1 2 1 1 2 3	2

설명

첫 번째 예제에서, 동관이는 $(1, 1), (1, 3), (3, 1), (3, 3)$ 이외의 격자에는 갈 수 없기 때문에 평생 트램펄린을 탈 것이고, 따라서 탈출구까지 갈 수 없다.

두 번째 예제는 $(1, 1) \rightarrow (2, 1) \rightarrow (2, 3)$ 로 간다면 2번만에 이동할 수 있다. 이보다 빠른 방법은 없다.

문제 B. 대자보 (poster)

시간 제한: 2초

메모리 제한: 512 MB

고려대학교에는 학생들의 소통을 위한 게시판이 설치되어 있다. 고려대학교 학생들은 게시판에 대자보를 붙여 자신의 신념을 밝히기도 하고, 비리나 부당한 상황을 고발하기도 한다.

그러던 어느날 게시판에 숫자로만 이루어진 긴 수열의 대자보가 붙었다. 사람들은 며칠간 수열에 담긴 의미를 찾아보려고 노력했지만, 그 누구도 해결하지 못했고 결국 해당 대자보는 철거되었다. 하지만 같은 형식의 수열로 이루어진 대자보가 몇차례 등장하였고, 사람들은 이를 누군가의 장난이라고 여기게 되었다.

하지만 ALPS의 회장 이세정은 그 정답을 알고 있었다. 대자보의 수열은 알파벳 소문자로 이루어진 단어를 아래 방법을 따라 변환시킨 것이다.

- 단어의 각 알파벳 소문자를 알파벳 순으로 1에서 26으로 변환한다. a는 1로, b는 2로, c는 3으로, ..., y는 25로, z는 26으로 변환된다.
- 이웃한 두 숫자를 곱한 결과를 한 글자씩 나열한다. 예를 들어, apple의 각 알파벳은 1 16 16 12 5으로 변환된다. 이웃한 두 숫자를 곱하면 16 256 192 60이 되며, 한 자씩 나열하면 1 6 2 5 6 1 9 2 6 0이 된다.

세정이는 해당 대자보가 올라올 때마다 대자보를 해석하면서, 대자보를 붙인 사람에 대한 증거를 모으고 있다. 하지만 한 대자보의 수열이 수많은 알파벳 문자열로 변환이 될 수 있다. 일단 세정이는 가능한 원본 문자열의 개수를 구해보고자 한다. 값이 매우 커질 수 있으므로, 1,000,000,007로 나눈 나머지를 구하자.

입력 형식

첫 번째 줄에 수열의 길이인 정수 N 이 주어진다. ($3 \leq N \leq 300,000$)

두 번째 줄에 N 개의 한 자리 정수 a_1, a_2, \dots, a_N 이 공백으로 구분되어 주어진다. ($0 \leq a_i \leq 9$)

a_i 는 수열의 i 번째 숫자를 의미한다. 첫 번째 숫자가 0인 경우는 주어지지 않는다.

출력 형식

주어진 수열에 대해 세정이가 확인해야 하는 단어의 수를 1,000,000,007로 나눈 나머지를 출력한다.

예제

표준 입력	표준 출력
3	7
4 4 4	
6	12
3 6 3 6 3 6	
10	0
2 8 2 1 9 0 9 2 2 4	

문제 C. 신앙 (faith)

시간 제한: 1 초

메모리 제한: 512 MB

새내기 시절 안수빈은 중앙광장에서 코딩하던 도중 노트북 배터리를 모두 소모한 적이 있다. 이에 수빈이는 화를 참지 못하고 발을 세게 굴렸고, 그 때의 여파로 3층 건물 하나스퀘어는 지하에 박히게 되었다. 그 때의 여파로 노벨 광장 한복판에 박힌 석조물은 아직까지도 뽑히지 않아 지금은 랜드마크로 자리잡게 되었다.

하지만 그런 비화를 모르는 수많은 사람들은 안암에 토착 신이 존재한다고 믿게 되었다. 토착 신을 믿는 신도들은 토착 신에게 기도를 드리며 신앙을 증명하며, 한편으로 하나스퀘어 사태 복구를 위한 성금을 주기적으로 낸다. 각자의 신앙심 f_i 나 현금 금액 m_i 는 양의 정수로 표현된다.

이들을 이끄는 사제는 신도들의 마음을 조종할 수 있다. 사제는 빨간 약과 파란 약을 이용해서 신앙심이나 내고자 하는 성금을 조작할 수 있다.

- 빨간 약: 먹은 사람의 신앙심의 값을 2배로 증가시킨다. ($f_i := 2f_i$)
- 파란 약: 먹은 사람의 현금 금액을 신앙심의 값으로 바꾼다. ($m_i := f_i$)

사제가 빨간 약과 파란 약을 신도들에게 적절히 먹였을 때 얻을 수 있는 현금 총합을 최댓값을 구해라. 약을 모두 사용할 필요는 없다.

입력 형식

첫째 줄에 신도의 수 N , 빨간 약의 수 R , 파란 약의 수 B 가 공백으로 구분되어 주어진다. ($1 \leq N \leq 1,000$, $0 \leq R \leq 20$, $0 \leq B \leq 1,000$)

이후 N 개의 줄에 걸쳐 두 정수 f_i 와 m_i 가 공백으로 구분되어 주어진다. ($1 \leq f_i, m_i \leq 100,000$) 이는 i 번째 신도의 신앙심과 현금 금액을 의미한다.

출력 형식

첫 번째 줄에 사제가 빨간 약과 파란 약을 신도들에게 적절히 먹였을 때 얻을 수 있는 현금 총합의 최댓값을 출력한다.

예제

표준 입력	표준 출력
3 1 2 3 5 3 1 2 2	13
5 2 2 8 7 5 2 17 100 10 12 5 5	157

문제 D. 얼마나 예뻐? (pretty)

시간 제한: 1초

메모리 제한: 512 MB

12월 25일은 크리스마스이다. 크리스마스를 맞은 피카츄는 지우를 위해 트리를 만들기로 했다. 금전적인 문제로 살아있는 나무를 구할 수 없게 된 피카츄는 나무 대신 그래프의 일종인 트리를 꾸미기로 했다.

트리란 모든 정점이 연결되어 있으며, 한 정점에서 다른 정점으로 가는 경로가 유일한 그래프이다. 트리에서 간선으로 연결된 두 정점 u 와 v 에 대해 u 가 루트 정점에 더 가까운 정점이라고 할 때, v 의 부모는 u 이며 u 는 v 를 자식으로 가진다. 트리에 루트 정점은 단 하나 존재하며, 정의에 의해 루트 정점은 부모 정점이 없다. 정점 x 의 서브트리란 x 와 x 의 자식들의 서브트리로 구성된 트리를 의미한다.

정점 x 에서의 전위 순회는 트리를 특정한 규칙에 의해 방문하는 순서로, 다음과 같이 정의된다.

- 정점 x 를 방문한다.
- 자식 정점의 번호의 오름차순으로 (작은 번호부터 큰 번호로) 전위 순회를 한다.

전위 순회의 정의에 의해 전위 순회는 정점별로 유일하다. 추가적으로, 트리의 전위 순회는 루트 정점에서의 전위 순회로 정의한다.

한 정점에서의 전위 순회는 그 서브트리의 모든 정점을 정확히 한 번만 방문하기 때문에 의미가 있다.

수학적 정의가 귀찮았지만 트리를 그리는 건 좋았던 피카츄는 인터넷 검색을 통해 1번 정점이 루트인 트리를 그렸다. 트리를 더 장식하기 위해 각 정점에 팔호(여는 팔호 '(' 또는 닫는 팔호 ')')를 한 개씩 그렸는데, 트리를 본 지우는 정점이 다음의 성질을 만족하면 아름답다고 정의했다.

- 정점 x 에서의 전위 순회를 하며, 방문한 순서대로 각 정점에 부여된 팔호를 추가해가며 문자열을 얻을 수 있다. 이렇게 만들 수 있는 팔호 문자열이 올바른 팔호 문자열이라면 x 는 아름답다.

올바른 팔호 문자열은 다음과 같이 정의된다.

1. 빈 문자열은 올바른 팔호 문자열이다.
2. A가 올바른 팔호 문자열이면 팔호를 씌운 (A)도 올바른 팔호 문자열이다.
3. A와 B가 올바른 팔호 문자열이면 둘을 이어붙인 AB도 올바른 팔호 문자열이다.

지우는 트리의 아름다움을 트리의 정점 중 아름다운 정점의 수라고 정의했다. 피카츄를 도와 피카츄가 꾸민 트리가 얼마나 아름다운지 구하자.

입력 형식

첫 번째 줄에 노드의 개수 N ($2 \leq N \leq 200,000$)이 주어진다.

두 번째 줄에는 0과 1로만 이루어진 N 개의 정수 a_1, a_2, \dots, a_N 가 공백으로 구분되어 주어진다. a_i 가 0이면 정점 i 에 여는 팔호 '('가 적혀 있으며, 1이면 ')'이 적혀 있다.

이후 $N - 1$ 개 줄에 걸쳐, 두 정수 a 와 b 가 공백으로 구분되어 주어진다. ($1 \leq a, b \leq N$) 이는 정점 a 와 정점 b 가 간선으로 연결되어 있음을 의미한다.

입력으로 들어오는 그래프는 트리이다.

출력 형식

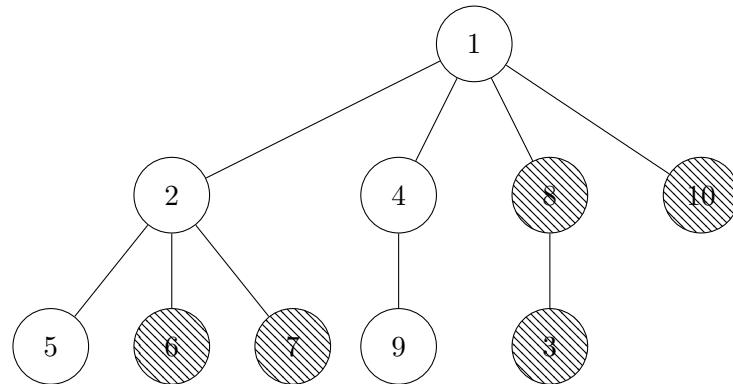
첫 번째 줄에 피카츄가 만든 트리의 아름다움을 출력한다.

예제

표준 입력	표준 출력
10 0 0 1 0 0 1 1 1 0 1 1 2 1 8 1 4 1 10 2 5 6 2 2 7 8 3 4 9	2

설명

첫 번째 예제에서 닫는 괄호가 있는 칸에 빗금을 쳐서 그리면 다음과 같다.



정점 1에서의 전위 순회는 $1 \rightarrow 2 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 4 \rightarrow 9 \rightarrow 8 \rightarrow 3 \rightarrow 10$ 이고, 올바른 괄호 문자열 $((())(()))$ 을 생성하기에 아름다운 정점이다.

정점 2에서의 전위 순회는 $2 \rightarrow 5 \rightarrow 6 \rightarrow 7$ 으로, 올바른 괄호 문자열 $((()$ 을 생성하기에 아름다운 정점이다.

그 외에 아름다운 정점은 존재하지 않는다.

문제 E. Sum without carries (nocarry)

시간 제한: 3초

메모리 제한: 1024 MB

정수 x, y 에 대해 $x @ y$ 라는 연산을 정의하자. $x @ y$ 는 x 와 y 를 d 진법 상에서 더하되, carry(자리 올림)가 없는 연산이다. 즉, 각 자리수를 따로 더하고 각 자리마다 d 로 나눈 나머지를 취하는 연산이다.

예를 들어, $d = 4$ 이면, $7_{10} @ 10_{10} = 13_4 @ 22_4 = 31_4 = 13_{10}$ 이 된다.

당신에게 q 개의 쿼리가 주어진다. 각 쿼리는 2개의 정수 l, R 을 담고 있다. 당신은 각 쿼리마다 다음을 만족하는 r 중에서 가장 큰 r 값을 구해야 한다.

- $l < r \leq R$
- $a_l @ a_{l+1} @ \dots @ a_r$ 이 가능한 한 제일 높아야 한다.

입력 형식

첫 번째 줄에 세 정수 n, d, q 가 공백으로 구분되어 주어진다. ($2 \leq n \leq 10^5$, $2 \leq d \leq 10$, $1 \leq q \leq 10^5$) n 은 수열의 길이, d 는 진법, q 는 쿼리의 개수이다.

두 번째 줄에 수열의 값을 의미하는 n 개의 정수 a_1, a_2, \dots, a_n 이 공백으로 구분되어 주어진다. ($0 \leq a_i \leq 10^{18}$)

세 번째 줄부터 q 개의 줄에 걸쳐 쿼리를 의미하는 2개의 정수 l, R 이 공백으로 구분되어 주어진다. ($1 \leq l < R \leq n$)

출력 형식

q 개의 줄에 걸쳐 각 쿼리에 대한 정답 r 을 출력한다.

예제

표준 입력	표준 출력
7 10 6	5
16 27 90 0 73 55 4	4
1 6	5
2 4	6
4 6	4
5 6	2
3 5	
1 3	

문제 F. 서버 증축 (server)

시간 제한: 1초

메모리 제한: 512 MB

“지난 밤, 선량한 서버가 죽었습니다. 관리자들은 모두 고개를 들어주세요.”

서버 관리자에게 서버가 다운되는 것만큼 무섭고 고된 일은 없을 것이다. 상현이도 그 마수에서 벗어날 수 없었다. 상현이는 자신이 도맡아 관리하는 서버들이 뺀는 걸 원하지 않기에 서버 용량을 늘리는 방향을 선택했다.

상현이는 회사로부터 $2^0, 2^1, \dots, 2^{k-1}$ MB의 용량을 지니고 있는 디스크를 각 용량별로 a 개씩 지원받았다. 상현이가 받은 총 ak 개의 디스크들은 서로 구분된다. 딥러닝을 열심히 돌려본 결과 서버에 정확히 n MB의 용량을 추가로 늘리면 최적의 성능을 발휘할 것으로 분석되었다. 문제는, n MB의 용량을 만드는 방법이 너무 다양했다는 것이다! 상현이는 서버 증축을 할 수 있는 방법의 수를 구해보고자 한다. 그 수가 너무 커질 걸 우려해, 1048573으로 나눈 나머지를 계산하려고 한다. $1048573 = 2^{20} - 3$ 은 소수이다.

입력 형식

첫 번째 줄에는 세 정수 k, a, n 이 공백으로 구분되어 주어진다. ($1 \leq k \leq 40, 1 \leq a \leq 10^6, 1 \leq n \leq 10^{15}$)

k 는 제공되는 디스크의 용량의 개수, a 는 각 용량별 서로 다른 디스크의 개수, n 은 증축해야 하는 용량을 의미한다.

출력 형식

첫 번째 줄에 정확히 n MB를 증축할 수 있도록 디스크를 선택하는 경우의 수를 1048573으로 나눈 나머지를 출력한다.

예제

표준 입력	표준 출력
2 3 5	12
5 10 1000	0
11 23 58	182185

설명

첫 번째 예제에서 총합 5 MB를 고르는 방법엔 (2 MB 1개, 1 MB 3개)의 3가지, (2 MB 2개, 1 MB 1개)의 9 가지로 총 12가지가 있다.

참고 사항

이 문제를 풀 때 뤼카의 정리(Lucas' Theorem)을 사용하면 편리할 것이다.

뤼카의 정리란, 양의 정수 m, n 이 소수 p 에 대해 p 진법으로 각각 $m_k m_{k-1} \cdots m_0 p, n_k n_{k-1} \cdots n_0 p$ 으로 표현될 때,

$$\binom{m}{n} \equiv \binom{m_k}{n_k} \times \binom{m_{k-1}}{n_{k-1}} \times \cdots \times \binom{m_1}{n_1} \times \binom{m_0}{n_0} \pmod{p}$$

이 성립한다는 것이다. $\binom{n}{r} = \frac{n!}{r!(n-r)!}$ 이며, $n < r$ 일 때 $\binom{n}{r} = 0$ 으로 정의한다.

예를 들어 $\binom{7}{4}$ 를 5로 나눈 나머지를 뤼카의 정리를 이용해 구해보자. $7 = 12_5$ 이고 $4 = 04_5$ 이기 때문에 $\binom{7}{4} \equiv \binom{1}{0} \cdot \binom{2}{4} \equiv 1 \cdot 0 \equiv 0 \pmod{5}$ 이다. 실제로 $\binom{7}{4} = 35$ 로, 5로 나눈 나머지는 0이다.

다른 예시로 $\binom{111}{11}$ 를 7로 나눈 나머지는, $111 = 216_7$ 이고 $11 = 014_7$ 이므로 $\binom{111}{11} \equiv \binom{2}{0} \cdot \binom{1}{1} \cdot \binom{6}{4} \equiv 1 \cdot 1 \cdot 1 \equiv 1 \pmod{7}$ 이다. 실제로 $\binom{111}{11} = 473239787751081 = 7 \times 67605683964440 + 1$ 이다.

문제 G. 기묘한 여행계획 2 (trip)

시간 제한: 1초

메모리 제한: 512 MB

기묘한 세계의 이상한 도시에 사는 앤리스는 다른 도시로 여행을 떠날 계획을 세웠다. 기묘한 세계에는 총 N 개의 도시가 있으며, M 개의 양방향 도로로 연결되어 있다. 각 도시는 순서대로 1번부터 N 번까지 번호가 붙어 있으며 앤리스가 사는 도시의 번호는 1번이다. 각 도로는 이용하는데 필요한 비용 v 가 있고, 도로를 이용하기 위해서는 돈을 내야 한다. 기묘한 세계에서 도로 이용 비용을 부과하는 방법은 다음과 같다.

목적지에 도달할 때 까지 이용한 도로의 비용을 순서대로 v_1, v_2, \dots, v_k 라고 했을 때 $v_1 \oplus v_2 \oplus \dots \oplus v_k$ 가 부과된다. (\oplus 는 bitwise xor이다.)

앤리스는 가난하기 때문에 여행을 떠나기 전에 가장 효율적인 여행 방법을 구하려고 한다. 계산에 서툰 앤리스를 대신하여 앤리스가 사는 도시를 제외한 모든 도시까지 가는 최소 비용을 구하는 프로그램을 작성하시오.

앤리스가 사는 도시의 번호는 1번이다. 원하는 목적지에 도달할 때까지 이용하는 도로의 개수에 제한은 없으며, 1번 도시에서 출발해 도로들을 이용하여 다른 모든 도시에 도달할 수 있음이 보장된다.

입력 형식

첫 번째 줄에 도시의 수 N 과 도로의 수 M 이 공백으로 구분되어 주어진다.

$(2 \leq N \leq 2 \times 10^5, N - 1 \leq M \leq 2 \times 10^5)$

두 번째 줄부터 M 개의 줄에 걸쳐 도로의 정보를 의미하는 세 정수 a, b, c 가 공백으로 구분되어 주어진다. $(1 \leq a, b \leq N, a \neq b, 0 \leq c \leq 10^9)$ 이는 도시 a 와 도시 b 사이에 c 의 비용을 가지는 도로가 존재한다는 뜻이다.

두 도시를 직접적으로 연결하는 도로는 최대 1개 있다.

출력 형식

첫 번째 줄에 $N - 1$ 개의 수 C_2, C_3, \dots, C_N 를 공백으로 구분하여 출력한다. C_i 는 1번 도시에서 i 번 도시로 가는 최소 비용이다.

예제

표준 입력	표준 출력
3 3 1 2 1 2 3 5 1 3 6	1 4
4 5 1 2 5 2 3 7 1 3 1 4 1 0 3 4 4	0 1 0