2017
서울대학교 프로그래밍 경시대회

Division 2

주최 및 주관

후원

2017년 9월 10일

참가자를 위한 도움말

주의 사항

- 대회 시간은 13:00부터 17:00까지입니다. 대회가 진행되는 동안 인터넷 검색 및 전자기기 사용 등을 하실 수 없습니다. 단, 아래의 문서에 한해 대회 진행 중에도 참고하실 수 있으며, 책과 노트 등을 가져오신 경우 역시 참고하실 수 있습니다.
  - C reference: http://en.cppreference.com/w/c
  - C++ reference: http://en.cppreference.com/w/cpp
  - Java documentation: http://docs.oracle.com/javase/8/docs/api/
  - Python3 reference: https://docs.python.org/3/reference/
- 대회는 DOMjudge(https://www.domjudge.org/)를 이용하여 진행됩니다. 별도로 제공되는 계정 정보를 이용하여 로그인하신 뒤 코드 제출 및 결과 확인 등을 하실 수 있습니다.
- 모든 입력은 표준 입력으로 주어지며, 모든 출력은 표준 출력으로 합니다.
- 테스트 케이스가 존재하는 문제의 경우, 테스트 케이스에 대한 출력을 모아서 하실 필요 없이, 각 테스트 케이스를 처리할 때마다 출력해도 괜찮습니다.
- *중요!! 리턴 코드와 표준 오류(standard error, stderr) 스트림 출력에 주의하십시오. 프로세스가* 0 이 아닌 리턴 코드를 되돌리는 경우나 표준 오류 스트림에 출력을 하는 경우 "런타임 에러"를 받게 됩니다.
- 문제에 대한 질의 사항은 대회 페이지의 질문 기능을 사용해 주시기 바랍니다. 이 때 대답해 드리기 어려운 질문에 대해서는 "답변을 드릴 수 없습니다"로 대답될 수 있으므로 유의하십시오.

채점 결과에 대하여

CORRECT 제출하신 답안이 모든 테스트 데이터를 정해진 시간 안에 통과하여 정답으로 인정되었음을 의미합니다.

COMPILER-ERROR 제출하신 답안 프로그램을 컴파일하는 도중 오류가 발생하였음을 의미합니다.

RUN-ERROR 제출하신 답안 프로그램을 실행하는 도중 프로세스가 비정상적으로 종료되었음을 의미합니다.

TIMELIMIT 제출하신 답안 프로그램이 정해진 시간 안에 종료되지 않았음을 의미합니다.

WRONG-ANSWER 제출하신 답안 프로그램이 테스트 데이터에 대해 생성한 출력이 출제자의 정답과 일치하지 않음을 의미합니다.

만약 여러 가지의 원인으로 인해 "CORRECT"가 아닌 다른 결과를 얻으셨다면, 그 중 어떤 것도 결과가 될 수 있습니다. 예를 들어 답도 잘못되었고 비정상적인 동작도 수행하는 코드를 제출하신 경우 대부분 "RUN-ERROR"를 받으시게 되지만, 경우에 따라서 "WRONG-ANSWER"를 받을 수도 있습니다.

Problem A. 여우 사인

도주는 동물을 좋아한다. 그 중에서도 여우를 정말 좋아한다! 어느 날, 힐링을 위해 여우 사진을 검색하던 도주는 "여우 사인"이라는 손 모양을 발견했다.

여우 사인은 엄지손가락, 중지, 약지의 끝을 맞닿게 하고 검지와 새끼손가락은 곧게 펴서 여우의 얼굴과 두 귀를 표현한 손 모양이다.

도주는 자신의 여우 사랑을 널리 알리기 위해 때때로 이 손 모양을 하고 귀여운 척을 하기로 했다. 도주의 손 모양이 주어질 때, 그것을 여우 사인이라고 할 수 있는지 판별해 보자. 손 모양을 판별할 때는 손가락이 곧게 펴져 있는지 구부러져 있는지, 손가락의 어느 부분이 닿아 있는지 등의 사소한 것은 신경쓰지 않기로 한다.

Input

첫 번째 줄에 서로 닿아 있는 손가락 쌍의 개수 $N(1 \le N \le 10)$이 주어진다.

두 번째 줄부터 N개의 줄에 걸쳐 서로 닿아 있는 두 손가락을 의미하는 1 이상 5 이하의 숫자 두 개가 공백으로 구분되어 주어진다. 1은 엄지손가락, 2는 검지, 3은 중지, 4는 약지, 5는 새끼손가락을 의미한다.

입력 순서가 다른 것을 포함하여 같은 쌍이 여러 번 주어지거나 한 손가락만으로 이루어진 쌍이 주어지는 경우는 없다.

Output

첫 번째 줄에 도주의 손 모양을 여우 사인이라고 할 수 있을 경우 `Wa-pa-pa-pa-pa-pa-pow!`를, 그렇지 않을 경우 `Woof-meow-tweet-squeek`을 출력한다.

Sample input and output

standard input	standard output
3 1 3 4 3 1 4	Wa-pa-pa-pa-pa-pa-pow!
5 1 3 3 4 4 1 1 5 5 4	Woof-meow-tweet-squeek

Notes

첫 번째 예시는 엄지손가락과 중지, 약지와 중지, 엄지손가락과 약지가 서로 닿아 있고 검지와 새끼손가락은 다른 손가락과 닿아 있지 않으므로 여우 사인이라고 할 수 있다.

두 번째 예시는 검지만 펴고 다른 손가락은 접은 손 모양이다.

Problem B. 고장난 시계

논산훈련소에 간 불쌍한 상언이는 첫 날 훈련소에서 다쳐 먼 거리를 이동할 때 버스를 타고 다녔다. 어느 날 버스를 탄 상언이는 훈련소 버스 앞에 붙어있는 시계를 보게 되었는데, 시계가 이상해 보인다는 사실을 관찰했다. 일반적인 시계라면 가리킬 수 없는 시간을 가르키고 있던 것이다. 이 시계를 본 상언이는 어떤 시계의 시침과 분침이 가리키는 방향을 보면 그 시계가 고장났는지 정상인지 판단할 수 있을거라고 생각했지만, 귀찮아서 생각을 그만 두기로 했다. 상언이를 대신해서 시계가 정상인지 이상한지 알려주자.

Input

첫 번째 줄에 시침의 각도 θ_1, 분침의 각도 θ_2 ($0 \leq \theta_1, \theta_2 \leq 359$)가 자연수로 주어진다. 시침, 분침의 각도는 12시 방향을 기준으로 시계방향으로 잰다. 예를 들어 3시는 90, 9시는 270 이다.

Output

첫 번째 줄에 시계의 각도가 정상인 경우 `O`를, 그렇지 않을 경우 `X`를 출력한다.

Sample input and output

standard input	standard output
180 0	O
0 180	X

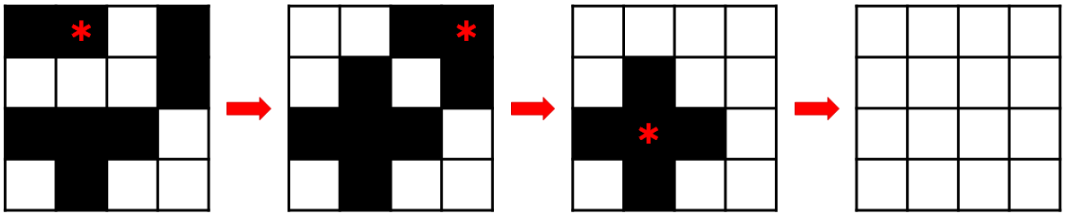
Notes

첫 번째 예시는 시침이 6을, 분침이 0을 가르키고 있는 상태로 6시의 시계 모양이다.

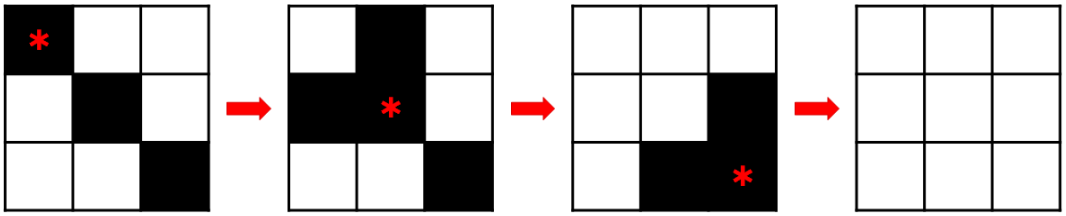
두 번째 예시는 시침이 0을, 분침이 6을 가르키고 있는 상태로 정상적인 시계에선 나올 수 없는 모양이다.

Problem C. 타일 뒤집기

석환이는 신기한 게임판을 가지고 있다. 이 게임판에는 한 면은 검은색, 한 면은 흰색으로 칠해진 타일이 N행 N열으로 배치되어 있다. 각 타일은 제자리에서 뒤집을 수 있는데, 타일 하나를 뒤집으면 그 타일과 상하좌우로 인접한 타일들이 같이 뒤집힌다. 석환이는 타일들이 무작위로 배치된 게임판에서 타일들을 적당히 뒤집어서 모든 타일이 흰색 면이 위를 향하도록 만드는 놀이를 좋아한다.



어느 날, 석환이가 게임판을 가지고 놀다가 자리를 비운 사이 석환이의 동생이 이 게임판을 발견했다. 석환이의 동생은 놀이의 규칙을 모르기 때문에, 그냥 처음 상태에서 검은색 면이 위를 향하고 있는 타일들을 전부 한 번씩 뒤집어 보았다. 그러자 놀랍게도 모든 타일이 흰색 면이 위를 향하게 되었다!



돌아온 석환이는 동생에게 놀이의 규칙을 알려 주려고 했지만, 그 전에 동생이 즐거워하는 모습을 더 보고 싶어져서 같은 특징을 갖는 게임판을 몇 번 더 만들어 주기로 했다. 석환이는 멋진 해커이기 때문에 게임판의 규칙을 따르지 않고 타일을 하나씩만 뒤집을 수도 있다. 하지만 아무 조건 없이 타일을 뒤집는 것은 별로 재미가 없었기 때문에, 석환이는 게임판의 첫 행의 타일들은 뒤집지 않고 원하는 배치를 만들어 보기로 했다.

Input

첫 번째 줄에 게임판의 크기 $N(1 \le N \le 1,000)$이 주어진다.

두 번째 줄에는 게임판의 첫 행의 타일들의 상태를 나타내는 N글자의 문자열이 주어진다. 문자열은 #와 .만으로 이루어져 있으며, #는 검은색 면이 위를 향하고 있는 타일, .는 흰색 면이 위를 향하고 있는 타일을 의미한다.

Output

N개의 줄에 걸쳐 석환이의 동생이 검은색 면이 위를 향하고 있는 타일들을 전부 한 번씩 뒤집어서 모든 타일이 흰색 면이 위를 향하도록 만들 수 있는 게임판의 모양을 출력한다. 입력 조건과 마찬가지로 검은색 면이 위를 향하고 있는 타일은 #, 흰색 면이 위를 향하고 있는 타일은 .로 나타낸다.

답이 유일하게 존재함이 보장된다.

Sample input and output

standard input	standard output
5 #..#.	#..#. .##.# .###. #.##. .#..#
8 #.##.###	#.##.### ..##.#.# ##...### ##.##... ...##.## ###...## #.#.##.. ###.##.#

Problem D. 전생했더니 슬라임 연구자였던 건에 대하여 (Easy)

안녕? 내 이름은 ntopia!

나는 원래 지구에 살고있던 평범한 20대 청년이었어. 어느날 길을 걷다가 괴한의 칼에 찔려 죽어버렸어. 그런데 이게 무슨 일이람! 정신을 차려보니 이세계에 떨어져버렸지 뭐야. 여기에서 나는 슬라임을 전문으로 연구하는 슬라임 연구자가 되어버린 것 같아. 나는 지금 아주 중요한 연구를 진행하고 있어. 이 연구가 성공하면 나는 내가 살던 세계로 돌아갈 수 있게 될거야. 이 연구를 도와주지 않겠니?

이 곳의 슬라임은 모두 슬라임 에너지라는 것을 갖고 있고 그 양은 2 이상의 자연수로 표현돼. 나는 슬라임을 분할했을 때 슬라임 에너지가 어떻게 변화하는지에 대해 연구하고 있어.

슬라임 분할 과정은 1마리를 분할해서 2마리를 만들어내는 식으로 이루어져. K만큼의 슬라임 에너지를 갖고 있는 슬라임이 있었다고 해보자. 이 슬라임을 적절히 분할하면 A만큼의 에너지를 갖고 있는 슬라임과 B만큼의 에너지를 갖고 있는 슬라임을 만들 수 있고 (A, B는 2 이상의 자연수), 항상 $K = A * B$ 를 만족해. 이렇게 분할하다보면 언젠가는 분할이 되지 않는 슬라임도 생기겠지?

그리고 슬라임 분할 기술이 아직 완벽하지 않아서 슬라임을 분할할 때마다 흠집이 하나씩 생기게 돼. 구체적으로, 흠집이 T개인 슬라임을 분할하면 흠집이 $T + 1$개인 슬라임 2마리가 생기는 것이지.

나에겐 지금 슬라임 에너지가 K이고 흠집이 하나도 없는 슬라임이 있어. 이 슬라임을 분할하고 또 분할해서 분할이 가능한 슬라임이 존재하지 않을 때 까지 마구마구 분할해야해. 그렇게 다 분할하고나면 마지막에 남은 슬라임들에 흠집이 적당히 생겼겠지? (물론 생기지 않았을 수도 있어) 그 슬라임들 중에서 흠집이 제일 많이 생긴 녀석의 흠집의 개수가 최소가 되도록 분할을 적절히 수행하는 것이 내 연구의 목표야.

내 연구를 도와줘! 부탁이야!!

Input

첫 번째 줄에 처음 주어진 슬라임의 에너지 K ($2 \leq K \leq 100,000$) 가 주어진다.

Output

슬라임을 끝까지 분할했을 때, 가장 많이 생긴 흠집의 개수의 최소값을 출력한다.

Sample input and output

standard input	standard output
10	1
3	0
24	2

Notes

처음에 에너지가 24이고 흠집이 없는 슬라임이 있다. 이 슬라임을 에너지가 4인 슬라임과 6인 슬라임으로 분할할 수 있고 각각은 흠집이 1개가 된다.

에너지가 4이고 흠집이 1개인 슬라임은 에너지가 2이고 흠집이 2개인 슬라임 2마리로 분할할 수 있다.

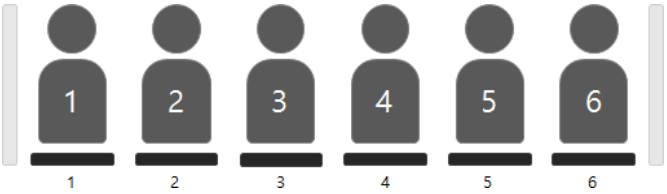
에너지가 6이고 흠집이 1개인 슬라임은 에너지가 2이고 흠집이 2개, 에너지가 3이고 흠집이 2개인 슬라임으로 분할할 수 있다.

이렇게 분할하고 나면 더이상 분할이 가능한 슬라임이 존재하지 않게 된다.

이 때 가장 많이 생긴 흠집의 개수는 2개이다. 이보다 더 적게 되도록 분할하는 방법은 존재하지 않는다.

Problem E. 셔틀버스

서울대학교 내부를 운행하는 셔틀버스에는 한쪽 벽면에 N개의 좌석이 일렬로 놓여 있다. 각 좌석은 가장 왼쪽 좌석부터 시작하여 1번부터 N번까지의 번호가 붙어 있다. 이 버스는 학교 입구에서 N명의 학생들을 태운 뒤 출발하고, 학생들은 각자 하나의 좌석을 골라 앉는다. 편의상 출발할 때 i번 좌석에 앉은 학생을 i번 학생으로 부르기로 한다.



학생들은 칸막이에 기대서 조는 것을 좋아하기 때문에 되도록 양쪽 끝 자리에 앉고 싶어한다. 그래서 바로 옆에 앉아 있던 학생이 내리거나 다른 자리로 옮겨 가서 좌석이 비었을 때 그 좌석으로 옮겨 앉는 게 양쪽 끝 자리에 더 가까워질 경우, 즉 1번 좌석과 N번 좌석 중 더 가까운 좌석까지의 거리가 줄어들 경우 그 좌석으로 옮겨 앉는다. 한 학생이 버스에서 내리는 즉시 모든 학생이 이 규칙에 따라 이동한다.

셔틀버스 기사 찬수는 버스에서 내리는 학생들을 보면서 지금 어떤 좌석에 어떤 학생이 앉아 있는지 궁금해졌다. 찬수를 위해 아래의 두 가지 연산을 입력되는 순서대로 수행하는 프로그램을 작성해 주자.

1. `1 x` : x번 학생이 버스에서 내린다. 이 학생은 버스에 타고 있던 학생임이 보장된다.
2. `2 x` : x번 좌석에 앉아 있는 학생의 번호를 출력한다. 좌석이 비어 있을 경우는 `0`을 출력한다.

찬수가 운전하는 버스는 차고지로 들어가는 버스이기 때문에 새로운 학생을 태우지는 않는다.

Input

첫 번째 줄에 셔틀버스에 있는 좌석의 수 $N(1 \le N \le 100,000)$, 처리해야 하는 연산의 수 $M(1 \le M \le 200,000)$이 주어진다.

두 번째 줄부터 M개의 줄에 걸쳐 각 쿼리의 종류(`1` 또는 `2`)와 $x(1 \le x \le N)$ 값이 공백을 사이에 두고 주어진다.

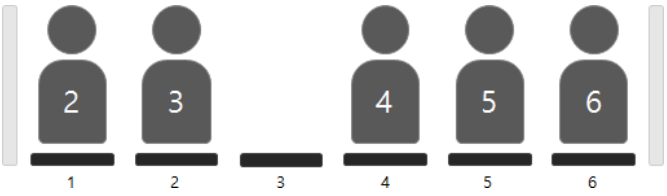
Output

각 2번 쿼리의 결과를 입력되는 순서대로 한 줄에 하나씩 출력한다.

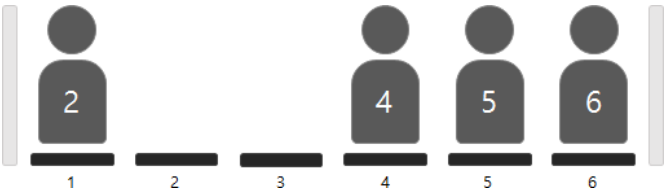
Sample input and output

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

Notes



1번 학생이 내리면 2번 학생과 3번 학생이 순서대로 이동한다. 4번 학생은 3번 좌석으로 옮겨 앉더라도 양 끝 좌석까지의 거리가 변하지 않기 때문에 이동하지 않는다.



3번 학생이 내리면 4번 학생은 2번 좌석으로 이동하는 것이 이득이지만, 바로 옆 자리가 아니므로 이동하지 않는다.

5번 학생이 내리면 4번 학생이 이동한다.

Problem F. 관악산 등산

서울대학교에는 "누가 조국의 미래를 묻거든 고개를 들어 관악을 보게 하라"라는 유명한 문구가 있다. 어느 날 Unused는 Corea에게 조국의 미래를 물었고, Corea는 직접 관악산에 올라가 조국의 미래를 보고 답해 주기로 했다.

관악산의 등산로는 1부터 N까지의 서로 다른 번호가 붙어 있는 N개의 쉼터와 두 쉼터 사이를 오고갈 수 있는 M개의 길들로 이루어져 있다. Corea는 지면에서부터 산을 올라가는 것은 너무 귀찮다고 생각했기 때문에, 케이블카를 타고 임의의 쉼터에서 내린 다음 등산을 시작하기로 했다. 심지어 등산로 지도를 보는 것도 귀찮았던 Corea는 지금 있는 쉼터와 길 하나로 연결되어 있으면서 지금보다 더 높은 곳에 있는 쉼터를 하나 골라서 올라가는 것을 반복하면 산의 정상에 도착할 수 있을 거라고 생각했다. 물론 실제로는 그렇지 않을 수도 있다.

Corea가 임의의 쉼터에서 출발하여 자신의 생각에 따라 산을 오를 때, 최대 몇 개의 쉼터를 방문할 수 있는지 구하여라.

Input

첫 번째 줄에 등산로에 있는 쉼터의 수 $N(1 \leq N \leq 3000)$과 두 쉼터 사이를 연결하는 길의 수 $M(1 \leq M \leq 50,000)$이 주어진다.

두 번째 줄에는 각 쉼터의 높이를 나타내는 N개의 정수가 번호 순서대로 주어진다. 각 쉼터의 높이는 1 이상 $1,000,000$ 이하이며 서로 다르다.

세 번째 줄부터 M개의 줄에 걸쳐 각각의 길이 연결하는 두 쉼터의 번호가 공백으로 구분되어 주어진다. 쉼터의 번호는 1 이상 N 이하의 정수이다. 양 끝점이 같은 쉼터인 길은 없으며, 임의의 두 쉼터를 연결하는 길이 여러 개 존재할 수 있다.

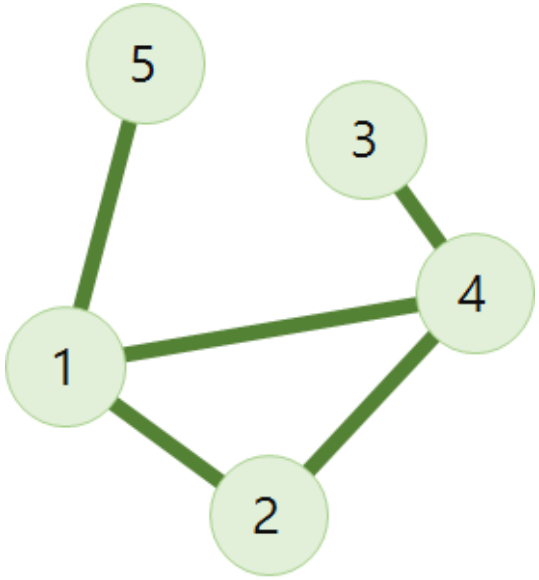
Output

Corea가 각 쉼터에서 출발할 때 최대로 방문할 수 있는 쉼터의 개수를 번호 순서대로 한 줄에 하나씩 N줄에 걸쳐 출력한다.

Sample input and output

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

Notes



2번 쉼터에서 출발하면 1번, 4번, 3번 쉼터를 차례대로 방문할 때 가장 많은 쉼터를 방문할 수 있다.

5번 쉼터는 3번 쉼터보다 높은 곳에 있지만 길 하나로 연결되어 있지 않으므로 3번 쉼터에서 5번 쉼터로 이동할 수 없다.

Problem G. 앵무새

자가용 비행기를 타고 세계 일주를 하던 pps789와 cseteram은 어느 날 엔진 고장으로 인해 이름 모를 섬에 불시착하게 된다. 그들은 이 섬을 탐험하는 도중 아주 신기한 사실을 알게 되었는데, 바로 이 섬에 사는 앵무새들은 놀라울 정도로 인간의 말을 흉내내는 데 뛰어나다는 것이다. 그들은 서로 떨어져 섬을 탐험하기로 하였으며, 필요하다면 앵무새를 이용해 서로에게 연락을 하기로 약속하였다.

1개월 후, pps789는 섬의 비밀을 밝힐 결정적인 증거를 찾게 된다. 그는 이 세기의 대발견을 cseteram에게 공유하고자 하였으나, 그의 발견은 방대하여 앵무새 한 마리가 기억하기에는 너무 많은 양이었다. 그렇기에 pps789는 앵무새 한 마리 대신 앵무새 N마리를 이용하여 자신의 발견을 기록하였으며, 이 앵무새들을 cseteram을 향해 날렸다.

한편 섬의 반대편에서 탐험을 계속하던 cseteram은 앵무새 N마리가 자신에게 날아와 각자 할 말을 하는 것을 보고 당황하였다. pps789가 긴 글을 전달하고 싶었던 것은 알아차렸지만, 각각의 앵무새들이 말하는 것을 차례대로 기록하다 보니 원문이 무엇인지 알 수 없을 정도로 단어의 순서가 엉켜버린 것이다. 대신 그는 관찰을 통해 몇 가지 규칙을 발견할 수 있었다.

1. 한 앵무새는 한 문장을 기억하고 있다. 문장은 여러 단어로 이루어져있는데, 앵무새는 이 단어들을 순서대로 말한다.
2. 한 앵무새가 단어를 말하고 그 다음 단어를 말하기 전에는 약간의 간격이 있는데, 이 때 다른 앵무새가 말을 가로채고 자신의 문장을 말할 수 있다.
3. 한 앵무새가 단어를 말하는 도중에는, 다른 앵무새가 말을 가로채지 않는다.

pps789가 각각의 앵무새들에게 전달한 문장 S_i와, cseteram이 받아 적은 문장 L이 주어진다. 이 때 문장 L이 위 규칙들을 이용하여 나올 수 있는 문장인지 판별하시오.

Input

첫 번째 줄에 앵무새의 수 N ($1 \leq N \leq 1000$) 이 주어진다.

두 번째 줄부터 N개의 줄에 걸쳐 각 앵무새가 말한 문장 S_i ($1 \leq i \leq N$) 가 주어지는데, 각 문장을 이루는 단어는 스페이스 한 칸을 구분으로 하여 주어진다. 문장 S_i를 이루는 단어의 수는 100개를 넘지 않으며, 각 단어는 최대 32개의 영문 소문자로 구성되어있다.

$N+2$ 번째 줄에는 cseteram이 받아 적은 문장 L이 주어진다.

Output

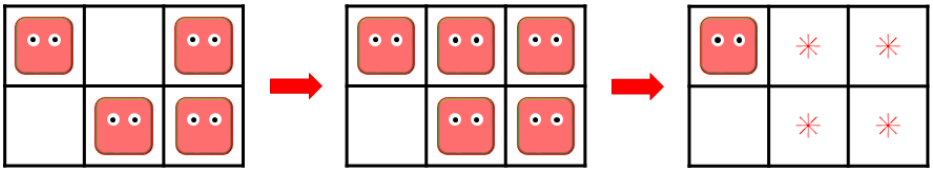
문장 L이 가능한 문장이라면 Possible을, 불가능한 문장이라면 Impossible을 출력한다.

Sample input and output

standard input	standard output
3 i want to see you next week good luck i want next good luck week to see you	Possible
2 i found an interesting cave i found an cave interesting	Impossible
2 please be careful pen pineapple apple pen	Impossible

Problem H. 넴모넴모

네모는 뿌×××× 게임에 깊은 감명을 받아 "넴모넴모" 게임을 만들었다. 이 게임은 직사각형 모양의 격자판과 "넴모"라는 수수께끼의 생물을 이용해 하는 아주 간단한 게임이다. 게임의 규칙은 격자판의 비어 있는 칸을 임의로 골라 "넴모"를 하나 올려놓고, "넴모"가 올라간 칸 네 개가 2×2 사각형을 이룰 경우 그 위에 있는 "넴모"들을 모두 없애는 것을 질릴 때까지 반복하는 것이다.



하지만 안타깝게도 게임은 정말 재미가 없었고, 네모는 아주 빨리 질려 버리고 말았다. 실망한 네모는 격자판 위에 더 이상 없앨 "넴모"가 없는 상태에서 게임을 그만두기로 했다. 게임의 결과로 나올 수 있는 상태의 가짓수를 구하여라.

Input

첫 번째 줄에 격자판의 행의 개수 N, 열의 개수 M($N \geq 1$, $M \geq 1$, $1 \leq N \times M \leq 25$)이 공백으로 구분되어 주어진다.

Output

첫 번째 줄에 "넴모"들이 올라간 칸이 2×2 사각형을 이루지 않도록 격자판에 "넴모"들을 배치하는 가짓수를 출력한다.

Sample input and output

standard input	standard output
2 2	15
2 3	57
3 5	22077

Notes

2×2 격자판에 "넴모"들을 안정되게 배치하는 방법은 모든 경우($2^4 = 16$) 중 네 칸 모두에 "넴모"가 올라가 있는 경우를 제외한 15가지가 있다.