snupc 2021

2021 서울대학교 프로그래밍 경진대회

Division 2

주최 및 주관 

후원 NAVER

서울대학교 컴퓨터공학부
Seoul National University
Dept. of Computer Science and Engineering

STARTLINK

참가자를 위한 도움말

주의 사항

- 대회 시간은 13:30부터 17:30까지입니다. **대회가 진행되는 동안 인터넷 검색 등을 하실 수 있습니다.** 단, 인터넷에 있는 코드를 그대로 복사-붙여넣기하시는 경우, 유사도 검사 시 확인을 위해 **코드의 출처(인터넷 URL 등)를 주석으로 적어주시기 바랍니다.** 각 언어의 레퍼런스 사이트는 다음과 같습니다.
  - C/C++ reference: https://en.cppreference.com/w/
  - Python 3 reference: https://docs.python.org/3.9/
  - Java documentation: https://docs.oracle.com/javase/8/docs/api/
  - Kotlin documentation: https://kotlinlang.org/docs/home.html
- 대회는 Baekjoon Online Judge(https://www.acmicpc.net/) 플랫폼을 이용하여 진행됩니다. 별도로 제공되는 계정 정보를 이용하여 로그인하신 뒤 코드 제출 및 결과 확인 등을 하실 수 있습니다.
- 제출하실 답안 코드는 **C11, C++17, C++20, Java 8, Java 8 (OpenJDK), Python 3.9.5, PyPy3, Kotlin (JVM) 으로만** 작성하셔야 합니다. 단, 출제진은 C++20을 제외한 다른 언어로 주어진 시간 제한과 메모리 제한을 지키며 올바른 답을 내는 코드를 작성할 수 있다는 보장을 하지 않습니다.
- 모든 입력은 **표준 입력**으로 주어지며, 모든 출력은 **표준 출력**으로 합니다.
- 테스트 케이스가 존재하는 문제의 경우, 테스트 케이스에 대한 출력을 모아서 하실 필요 없이, 각 테스트 케이스를 처리할 때마다 출력해도 괜찮습니다.
- **리턴 코드와 표준 오류(standard error, stderr) 스트림 출력**에 주의하십시오. 프로세스가 0이 아닌 리턴 코드를 되돌리는 경우나 **표준 오류 스트림에 출력**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.
- 이 문제지는 참고용입니다. 문제지에 적힌 문제와 플랫폼에 올라온 문제가 다를 경우, 플랫폼의 문제가 우선합니다. 특히 이 문제지를 흑백으로 프린트하신 경우 색이 들어간 이미지는 플랫폼에서 보시기 바랍니다.

채점 결과에 대하여

맞았습니다!! 제출하신 답안이 모든 테스트 데이터를 정해진 시간과 메모리 제한 안에 통과하여 정답으로 인정되었음을 의미합니다.

틀렸습니다 제출하신 답안 프로그램이 테스트 데이터에 대해 생성한 출력이 출제자의 정답과 일치하지 않음을 의미합니다.

컴파일 에러 제출하신 답안 프로그램을 컴파일하는 도중 오류가 발생하였음을 의미합니다.

런타임 에러 제출하신 답안 프로그램을 실행하는 도중 프로세스가 비정상적인 행동을 하였음을 의미합니다.

시간 초과 제출하신 답안 프로그램이 정해진 시간 안에 종료되지 않았음을 의미합니다.

메모리 초과 제출하신 답안 프로그램이 정해진 메모리 제한보다 많은 메모리를 사용하였음을 의미합니다.

출력 형식이 잘못되었습니다 답을 올바르게 구했으나, 답안 프로그램의 출력 형식이 문제에 나와 있는 출력 형식과 다름을 의미합니다. 줄 뒤에 의미 없는 공백을 두 칸 이상 출력하거나 의미 없는 빈 줄을 출력할 경우 이 결과를 받을 수 있습니다.

출력 초과 답안 프로그램이 정답에 비해 너무 많은 출력을 했음을 의미합니다. 이 결과는 틀렸습니다와 같은 의미를 갖습니다.

만약 여러 가지의 원인으로 인해 "**맞았습니다!!**"가 아닌 다른 결과를 얻으셨다면, 그중 **어떤 것도 결과가 될 수 있습니다**. 예를 들어 답도 잘못되었고 비정상적인 동작도 수행하는 코드를 제출하신 경우 대부분 "**런타임 에러**"를 받으시게 되지만, 경우에 따라서 "**틀렸습니다**"를 받을 수도 있습니다.

문제 목록

문제지에 있는 문제가 총 9문제가 맞는지 확인하시기 바랍니다.

A 5의 수난

B 박스 그림 문자

C 실 전화기

D 누텔라 트리 (Easy)

E 뛰는 기물

F AND와 OR

G 자연수 색칠하기

H 트리 찾기

I 큰 수 뒤집기

모든 문제의 메모리 제한은 1GB로 동일합니다.

문제 A. 5의 수난

시간 제한	1 초
메모리 제한	1024 MB

키파는 문득 3과 4의 견고한 벽에 가로막혀 스포트라이트를 받지 못하는 **5**를 떠올렸다. '세상에 얼마나 많은 것들이 **5**와 관련이 있는데!'

키파는 **5**가 쓰이는 곳을 떠올리기 시작했다. 사람의 손가락도 **5**개, 정다면체의 개수도 **5**개, 알려진 불가촉 홀수는 **5** 뿐이고, 별은 보통 **오**각별, 그리고 무엇보다 "별이 **다섯** 개!"

그러자 문득 키파는 자신의 마음 속에서 **다섯**제곱을 하고 싶은 욕망이 올라오는 것을 느꼈다. 키파를 위해, **다섯** 자리 수를 입력받아, 각 자릿수의 **다섯**제곱의 합을 출력하는 프로그램을 작성해 주자.

입력

첫째 줄에 **다섯** 자리인 양의 정수 n이 주어진다. 주어지는 n은 $10^4 \leq n < 10^5$을 만족한다.

출력

첫째 줄에 각 자릿수의 **다섯**제곱의 합을 출력하라.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
12345	4425
54748	54748
92727	92727
93084	93084

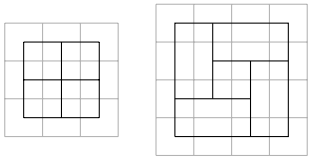
문제 B. 박스 그림 문자

시간 제한	1 초
메모리 제한	1024 MB

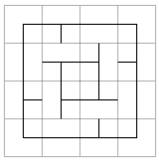
유니코드 문자 중에는 box-drawing character(박스 그림 문자)라는 종류의 문자들이 있다. 이 문자들은 텍스트 UI에서 구역을 나누는 테두리를 그리기 위해 만들어진 것으로, 상하좌우 방향의 선들로 이루어져 있다. 박스 그림 문자의 종류는 다음과 같이 총 11가지이다.

```
┌┬┐ ├┼┤ └┴┘ ─ │
```

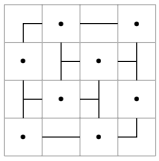
이 문자들을 이용하면 다음과 같이 재밌는 모양들을 만들 수 있다.



위 모양들은 모두 선이 끊어지는 곳이 없다는 특징이 있다. 예를 들어 어떤 문자가 오른쪽 방향의 선을 갖고 있다면, 그 오른쪽에는 반드시 왼쪽 방향의 선을 갖고 있는 문자가 있다. 이러한 특징을 갖는 모양들을 **아름다운 모양**이라고 하자. 다음은 아름다운 모양이 아닌 모양의 예시이다.



실버는 아름다운 모양 하나를 만들어 파일로 저장해 두었다. 하지만 컴퓨터가 바이러스에 감염되어 파일의 일부분이 `.`으로 날아가버렸다!



다행히도 `.`으로 대체된 부분들은 체스판처럼 배열되어 있다. 다시 말하면, $i+j$가 홀수일 때 i행 j열의 문자가 `.`으로 대체되어 있다. 자신의 작품이 망가져 좌절한 실버를 도와 모양을 복원해 보자. 단, 당신은 박스 그림 문자를 제대로 출력할 줄 모르기 때문에 각 문자들을 다음의 3×3 블록으로 치환하여 입출력할 것이다.

...	...	...	.#.	.#.	.#.
.##	###	##.	.##	###	##.
.#.	.#.	.#.	.#.	.#.	.#.
┌	┬	┐	├	┼	┤
.#.	.#.	.#.	...	.#.	...
.##	###	##.	###	.#.	...
...	...	...	...	.#.	...
└	┴	┘	—	|	.

입력

첫째 줄에 실버가 만든 모양의 행 수 N과 열 수 M이 공백을 사이에 두고 주어진다. ($2 \le N \le 300, 2 \le M \le 300$)

이후 $3N$개의 줄에 걸쳐 실버가 만든 모양이 주어진다. 각 줄은 $3M$개의 문자로 이루어져 있으며, 각 문자는 . 또는 #이다.

출력

$3N$개의 줄에 걸쳐 복원된 실버의 작품을 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
4 4 ..... .##...###... .#..... ....#.....#. ....##...##. ....#.....#. .#.....#.... .##...##.... .#.....#.... .....#. ...###...##.	..... .##########. .#..#.....#. .#..#.....#. .#..#######. .#..#..#..#. .#..#..#..#. .#######..#. .#.....#..#. .#.....#..#. .##########. .....

문제 C. 실 전화기

시간 제한	1 초
메모리 제한	1024 MB

서울대학교 신공학관 옥상에는 커다란 정원이 있다. 이 정원에는 다섯 마리의 토끼가 각자 굴을 하나씩 파서 살고 있다. 이 다섯 개의 굴은 정오각형을 이루고 있으며, 시계 방향을 따라 순서대로 1 번부터 5 번까지의 번호가 붙어 있다.

이 토끼들은 소리로 소통을 하는데, 다른 굴까지 전해질 정도로 큰 소리를 내지는 못한다. 토끼들은 넘치는 공대 감성과 문제 해결 의식으로 새로운 통신 수단을 만들기로 했고, 마침내 주변의 잡동사니를 모아서 **실 전화기**를 만들어 냈다. 실 전화기는 실이 팽팽해야 진동을 온전히 전달할 수 있기 때문에, 하나의 실 전화기는 두 개의 굴 사이를 일직선으로 연결한다.

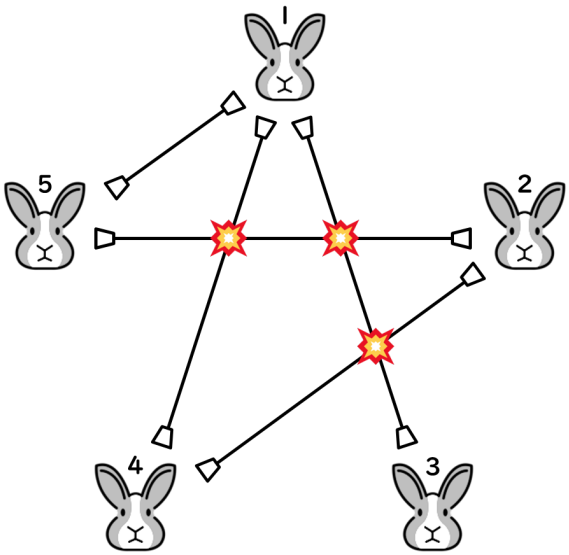


그림 1: 토끼굴들 사이에 5개의 실 전화기가 설치되어 있는 모습

실 전화기의 문제는 두 전화기의 실이 겹치면 진동이 제대로 전달되지 않아 통신이 이뤄지지 않는다는 것이다. 전화기들의 높이가 서로 다르면 문제가 없겠지만, 토끼들은 키가 작아서 전화기의 높이를 자유롭게 조절할 수 없다. 따라서 이를 해결하려면 몇몇 토끼가 다른 적절한 위치로 굴을 옮겨야만 한다.

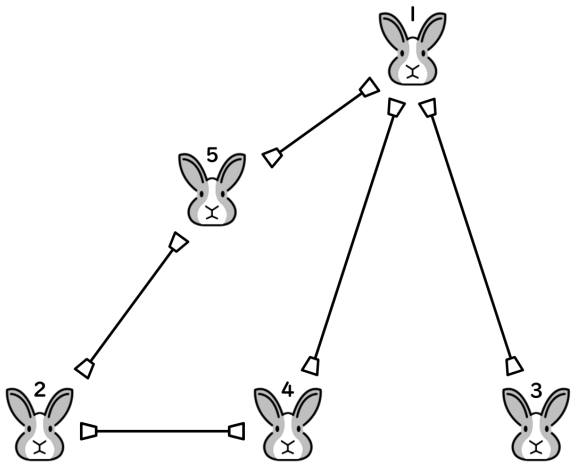


그림 2: 2번 토끼굴이 왼쪽 아래로 옮겨가서 문제가 해결된 모습

정원은 토끼들에게는 끝없는 평원처럼 느껴질 정도로 커다랗다. 따라서 각 토끼굴은 하나의 아주 작은 점으로 여길 수 있으며, 토끼굴을 옮길 때 새로운 굴의 위치에는 아무런 제약도 없다. 단, 굴을 다른 토끼굴과 완전히 같은 위치로 옮기거나, 굴을 옮긴 뒤에 전화기의 실이 다른 토끼굴 위를 지나가게 되어서는 안 된다.

현재 실 전화기의 배치가 주어질 때, 최소 몇 마리의 토끼가 굴을 옮겨야 실 전화기들이 서로 겹치지 않게 되는지 알아보자.

입력

첫째 줄에 토끼굴들 사이에 설치되어 있는 실 전화기의 개수 $N(1 \le N \le 10)$이 주어진다. 이후 N개의 줄에 걸쳐 각 전화기가 잇는 두 토끼굴의 번호가 공백으로 구분되어 주어진다.

각 전화기는 반드시 서로 다른 두 토끼굴을 연결하며, 어떤 두 토끼굴 사이에 여러 개의 실 전화기가 설치되어 있는 경우는 없다.

출력

첫째 줄에 모든 전화기의 실이 서로 겹치지 않도록 하기 위해 최소 몇 마리의 토끼가 굴을 옮겨야 하는지 출력한다. 만약 토끼들이 굴을 어떻게 옮기더라도 실이 겹치는 지점이 생긴다면 `-1`을 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
5 2 5 4 2 1 4 3 1 1 5	1
2 5 4 1 3	0
10 1 2 1 3 1 4 1 5 2 3 2 4 2 5 3 4 3 5 4 5	-1

노트

5개의 정점으로 이루어진 완전 그래프는 평면 위에 간선이 겹치지 않도록 그릴 수 없음이 잘 알려져 있다.

문제 D. 누텔라 트리 (Easy)

시간 제한	2 초
메모리 제한	1024 MB

민제는 정점이 N개인 트리를 가지고 있다. 이 트리의 각 정점은 빨간색 또는 검은색으로 칠해져 있다.

민제는 빨간색과 검은색 정점들로 가득한 이 트리를 보고 누텔라(Nutella)를 떠올렸다. 누텔라는 민제가 가장 좋아하는 초콜릿 잼으로, 로고는 다음과 같이 생겼다. 맨 앞 글자는 검은색, 나머지 글자는 빨간색임에 주목하자.

그림 1: 누텔라 로고

민제는 트리에서 누텔라 로고를 몇 개나 찾을 수 있을지 궁금해졌다.

다음 조건들을 만족하는 서로 다른 정점들의 열 $[v_1, v_2, \cdots, v_k]$를 **누텔라 경로**라 정의하자.

- k는 2 이상이다.
- 각 $1 \le i \le k-1$에 대해, v_i와 v_{i+1}은 트리에서 간선으로 직접 연결되어 있다.
- v_1은 검은색이다.
- 각 $2 \le i \le k$에 대해, v_i는 빨간색이다.

주어진 트리에서 누텔라 경로가 총 몇 개 있는지 구하시오.

입력

첫째 줄에 트리의 정점의 개수 N이 주어진다. ($2 \le N \le 100000$)

이후 $(N-1)$개 줄에 걸쳐 각 간선이 잇는 두 정점의 번호 u_i, v_i가 공백을 사이에 두고 주어진다. ($1 \le u_i \le N, 1 \le v_i \le N, u_i \ne v_i$)

그 다음 줄에는 알파벳 `B`, `R`로만 이루어진 길이 N의 문자열 C가 주어진다. C의 i번째 문자는 i번 정점의 색을 나타내며, `B`는 검은색, `R`는 빨간색을 의미한다.

출력

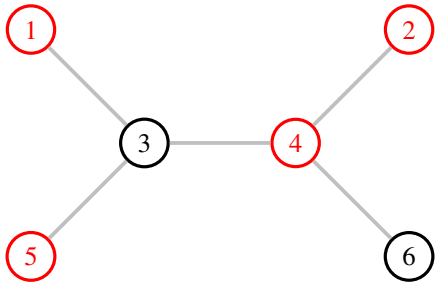
첫째 줄에 누텔라 경로의 개수를 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
6 1 3 2 4 5 3 4 6 3 4 RRBRRB	6

노트

예제로 주어진 트리를 그림으로 나타내면 다음과 같다.



문제 E. 뛰는 기물

시간 제한	0.5 초
메모리 제한	1024 MB

장기에는 마(馬)와 상(象)이라는 두 가지의 **뛰는 기물**이 있습니다. 다른 기물들과 다른 이 기물들의 공통점은 가로와 세로로 동시에 움직인다는 것입니다. 따라서 이동 경로를 직선으로 그리게 되면 항상 장기판의 가로 선과 세로 선을 빗겨가게 됩니다. 마(馬)는 가로 혹은 세로 방향으로 한 칸, 다른 방향으로 두 칸 이동하고, 상(象)은 가로 혹은 세로 방향으로 두 칸, 다른 방향으로 세 칸 이동합니다.

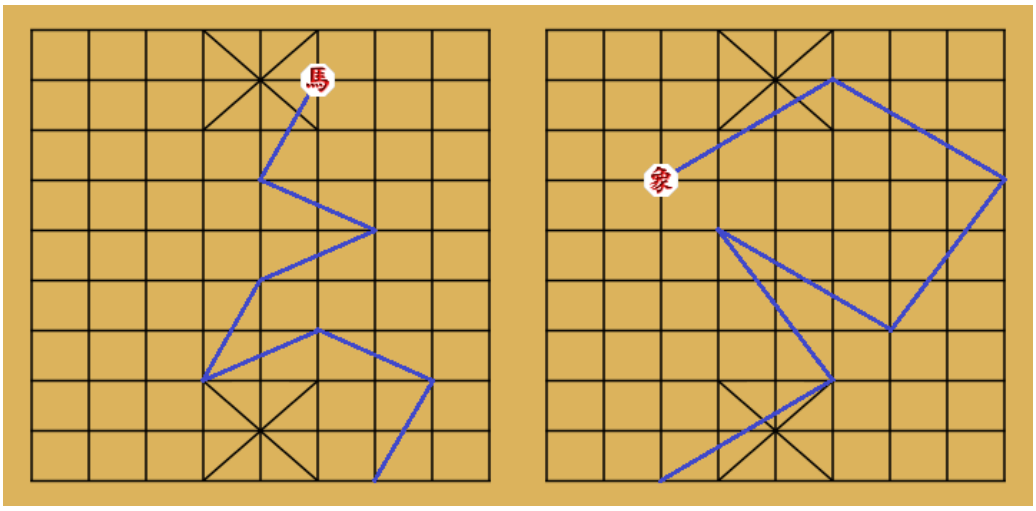


그림 1: 마(馬)와 상(象)의 이동 경로 예시, 왼쪽이 마(馬), 오른쪽이 상(象)입니다.

뛰는 기물의 존재 여부는 장기의 후반 흐름을 좌우할 만큼 중요합니다. 오로지 장기가 재밌어서 이런 중책을 천 번도 넘게 맡긴 키파는, 마(馬)와 상(象)에게 뛰는 기물로서의 자부심을 가질 수 있게 특별한 이름을 붙여 주기로 했습니다. 그 이름은, 기물이 각 방향으로 움직이는 칸 수를 앞에 붙여서 (n,m)-기물과 같이 정했습니다. 예를 들어 마(馬)는 이제 $(1,2)$-기물 혹은 $(2,1)$-기물, 상(象)은 $(2,3)$-기물 혹은 $(3,2)$-기물입니다.

이런 이름을 붙이자 키파는 다른 (N,M)-기물의 특성 역시 생각해 보기로 했습니다. 이를 위해 장기판은 조금 답답했기 때문에, 무한히 넓은 좌표평면을 생각하기로 했습니다. 곧 키파는 마(馬)와 상(象)과는 다르게 N, M에 따라 특정 지점에서 다른 지점으로 항상 갈 수 있는 것은 아니라는 것을 깨달았습니다. 예를 들어 $(2,4)$-기물은 좌표평면의 $(0,0)$ 위치에서 $(3,3)$ 위치로 갈 수 없습니다.

키파는 (N,M)-기물을 기특히 여겨 선물을 주려고 합니다. 그러나 기물이 너무 멀리 뛰어가 버려서 키파는 기물의 위치를 정확히 알 수 없었고, 대신 격자점 위에 선물을 두고 돌아가기로 했습니다. 기물이 선물이 놓인 격자점을 방문할 수 없게 되는 것이 걱정된 키파는 선물을 많이 가져와서 여러 곳의 격자점에 하나씩 놓기로 했습니다. 기물이 어떤 위치에 있더라도 선물이 놓인 격자점에 방문할 수 있도록 하기 위해, 키파가 준비해야 하는 선물의 최소 개수를 구해 봅시다.

입력

첫째 줄에 음이 아닌 정수 N과 M이 공백을 사이에 두고 주어집니다. N과 M은 모두 10^9보다 작거나 같으며, N과 M

이 모두 0인 입력은 주어지지 않습니다.

출력

첫째 줄에 키파가 준비해야 하는 선물의 최소 개수를 출력합니다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
2 4	4
1 2	1
2 3	1

노트

첫 번째 예제의 경우 가로 축과 세로 축에서 모두 홀수 점들과 짝수 점들이 나누어져 있기 때문에 답은 최소 $2 \times 2 = 4$ 입니다. 선물을 4개만 준비해도 된다는 사실을 증명할 수 있습니다.

두 번째 예제의 경우, 예를 들어 $(0,0) \to (1,2) \to (3,1) \to (1,0)$과 같은 방식으로 한 칸씩 이동할 수 있기 때문에 어떤 위치에든 선물을 하나만 놓아도 기물이 선물을 가져갈 수 있습니다.

세 번째 예제의 경우, 두 번째 예제와 마찬가지로 기물이 한 칸씩 이동할 수 있는 방법이 있습니다. 이 경우는 기물이 한 칸을 이동하는 데 최소 다섯 번의 이동을 필요로 합니다.

문제 F. AND와 OR

시간 제한	1 초
메모리 제한	1024 MB

당신에게 N개의 수가 주어집니다. 당신은 다음과 같은 작업을 하고 싶은 만큼 할 수 있습니다.

- 두 수를 고르고, 이 두 수를 두 수와 bitwise AND 및 bitwise OR가 모두 같은 두 음이 아닌 정수로 바꿉니다.

당신은 작업을 수행한 후 이 수들의 곱을 최소화하고 싶습니다. N개의 수의 최소화된 곱을 구하세요. 단, 이 곱이 매우 커질 수 있으니 10^9+7로 나눈 나머지를 대신 출력합니다.

Bitwise AND와 bitwise OR에 대한 설명은 노트 란을 참고하세요.

입력

첫째 줄에 2 이상 300000 이하인 정수 N이 주어집니다.

둘째 줄에 N개의 양의 정수가 공백을 사이에 두고 주어집니다. 각 정수는 2^{30}보다 작습니다.

출력

첫째 줄에 최소화된 곱을 10^9+7로 나눈 나머지를 출력합니다.

곱을 10^9+7로 나눈 나머지를 최소화하는 것이 아님에 주의하세요.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
3 3 6 10	60

노트

Bitwise operation은 각 연산을 비트 단위로 시행하는 것입니다.

예를 들어, 28과 87을 bitwise AND 연산한다고 합시다. 먼저 28과 87을 비트가 잘 드러나도록 2진수로 바꾸겠습니다.

$$\begin{aligned} &\texttt{0001 1100} &&= 28 \\ &\texttt{0101 0111} &&= 87 \end{aligned}$$

각 위치의 비트를 순서대로 AND해서 아래쪽에 적습니다. AND 연산의 경우 두 비트가 모두 1이어야만 결과가 1이고, 그렇지 않은 경우 0입니다.

$$\begin{aligned} &\texttt{0001 1100} &&= 28 \\ \texttt{AND}\quad &\texttt{0101 0111} &&= 87 \\ \hline &\texttt{0001 0100} &&= 20 \end{aligned}$$

마찬가지로 bitwise OR 연산을 시행할 수도 있습니다. `OR` 연산의 경우 두 비트가 모두 `0`이어야만 결과가 `0`이고, 그렇지 않은 경우 `1`입니다.

$$\begin{array}{rll} & \texttt{0001 1100} & = 28 \\ \texttt{OR} & \texttt{0101 0111} & = 87 \\ \hline & \texttt{0101 1111} & = 95 \end{array}$$

문제 G. 자연수 색칠하기

시간 제한	1 초
메모리 제한	1024 MB

1 부터 N 까지의 자연수를 색칠한다. 단, 서로소인 서로 다른 두 자연수는 다른 색으로 칠해야 한다. 최소한의 색을 써서 모든 자연수를 칠하는 방법을 찾는 프로그램을 작성하자.

입력

첫째 줄에 자연수 N 이 주어진다. ($1 \leq N \leq 500\,000$)

출력

첫째 줄에 사용한 색의 수 K 를 출력한다.

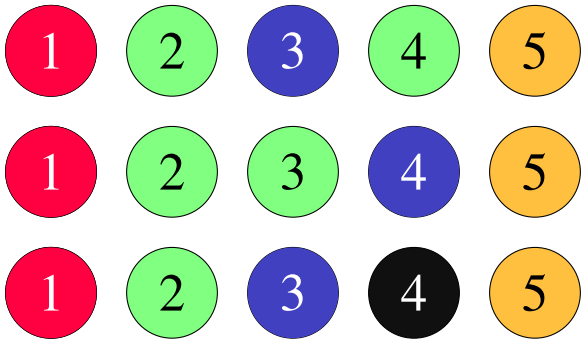
둘째 줄에 N 개의 수를 공백을 사이에 두고 출력한다. i 번째 수는 자연수 i 의 색이다. 색은 1 이상 K 이하의 정수로 나타낸다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
5	4 1 2 3 2 4

노트

다음은 $N = 5$ 일 때의 올바른 색칠과 올바르지 않은 색칠의 예시다.



첫 번째 색칠은 **올바른 색칠**이다. 2와 4는 같은 색으로 색칠했지만, 서로소가 아니므로 문제의 조건에 위배되지 않는다. 또한, 5까지의 자연수를 색칠하기 위해 최소 4개의 색이 필요함을 증명할 수 있다.

두 번째 색칠은 서로소인 2와 3이 같은 색으로 칠해졌기 때문에 **올바르지 않은 색칠**이다.

세 번째 색칠은 최소한의 색깔로 칠하지 않았기 때문에 **올바르지 않은 색칠**이다.

문제 H. 트리 찾기

시간 제한	5 초
메모리 제한	1024 MB

이 문제는 인터랙티브 문제입니다.

N개의 정점으로 이루어진 트리 T가 있다. 처음에 T의 간선들은 주어지지 않는다.

여러분은 T의 간선들을 모두 알아내야 한다. 이를 위해 채점 시스템에 다음의 질의를 할 수 있다.

- 자연수 K와 서로 다른 K개의 정점 $u_1, \cdots, u_K$를 선택한다. $(1 \le K \le N)$

이 질의에 대해 채점 시스템은 다음 조건을 만족하는 정점 v의 개수를 알려준다.

- u_i와 u_j를 잇는 최단 경로 위에 v가 있도록 하는 i, $j(1 \le i \le j \le K)$가 존재한다.

질의를 11111회 이하로 사용해 모든 간선들을 알아내야 한다.

입력

입력의 첫 줄에 T의 정점의 수 N이 주어진다. $(2 \le N \le 1000)$

이후 인터랙션이 시작된다.

질의를 하는 방법

T의 간선들을 알아내기 위해 채점 시스템에 다음과 같은 질의를 할 수 있다.

- 첫 줄에 K를 출력한다.
- 다음 줄에 $u_1, \cdots, u_K$를 공백으로 구분하여 출력한다.

출력 후에는 표준 출력 버퍼를 flush해 주어야 한다. 언어별로 표준 출력 버퍼를 flush하는 방법은 다음과 같다.

- C: `fflush(stdout)`
- C++: `std::cout << std::flush`
- Java: `System.out.flush()`
- Python: `sys.stdout.flush()`
- Kotlin: `System.out.flush()`

각각의 질의 이후에는 채점 시스템이 질의에 대한 답을 줄 것이다. 표준 입력으로 정수 하나를 입력받아 질의의 답을 얻을 수 있다.

단, 질의를 11112회 이상 사용했거나 잘못된 형식의 질의를 했다면 **틀렸습니다**를 받는다. 프로그램을 즉시 종료하지 않을 경우 다른 결과를 받을 수 있음에 유의하라.

알아낸 간선들을 출력하는 방법

모든 간선들을 알아냈다면 이를 출력하고 프로그램을 종료해야 한다. 간선들을 출력하는 방법은 다음과 같다.

- 첫째 줄에 `!`를 출력한다.
- 다음 $(N-1)$개의 줄에 걸쳐 각 간선이 잇는 두 정점의 번호를 공백으로 구분하여 출력한다.

T의 모든 간선들을 정확히 구했다면 **맞았습니다**를 받는다. 그렇지 않다면 **틀렸습니다**를 받는다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
5 3 5 2 4 3	 2 4 5 3 5 3 4 2 1 2 3 5 4 1 3 2 4 5 ! 3 1 2 4 2 1 5 2

문제 I. 큰 수 뒤집기

시간 제한	3 초
메모리 제한	1024 MB

문자열 S와 정수 X가 있다. 처음에 S는 빈 문자열이며 $X = 0$이다. 이때, 다음 쿼리를 수행해야 한다.

- 1부터 9까지의 숫자 중 하나가 주어지면, S의 맨 뒤에 그 숫자를 덧붙인다. 그리고 X에 S를 10진법 정수로 읽었을 때의 값을 더한다.
- 하이픈(-)이 주어지면, S를 뒤집는다.

모든 쿼리를 처리한 뒤 X의 값을 구하는 프로그램을 작성하자.

입력

첫째 줄에 쿼리 정보를 담고 있는 길이 1 이상 2000000 이하의 문자열 T가 주어진다. 각 문자는 1부터 9까지의 숫자 중 하나 또는 -이며, 첫 번째 문자는 -이 아니다.

출력

쿼리를 순서대로 모두 처리한 뒤 X의 값을 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
1234-567-89-123	1093443373651
12345-12345-12345-12345-12345	603572702347049670763373 5

노트

정답이 매우 클 수 있음에 유의하자.