서강대학교
SOGANG UNIVERSITY

Sogang Programming Contest 2019

Champion Division Problems

22nd November, 2019
Sogang University

STARTLINK

대회 규칙

- 대회 중 제출한 소스코드는 채점 서버에 의해 자동으로 채점되며, 실시간으로 결과를 알 수 있습니다. 문제를 제출하였을 때 '맞았습니다!!'를 받으면 문제를 푼 것으로, 이외의 결과를 받으면 틀린 것으로 생각합니다.
- 문제를 풀 때마다 패널티 점수가 누적됩니다. 패널티 점수는 모든 맞은 문제에 대해, 대회 시작 시간부터 그 문제를 풀기까지 걸린 시간을 t분, 처음으로 문제를 맞기 직전까지 제출한 횟수를 w번이라고 할 때 $(t + 20w)$ 점입니다.
- 순위는 푼 문제가 많은 순서대로, 푼 문제가 같을 경우에는 패널티 점수의 합이 적은 순서대로 결정됩니다.
- 사용 가능 언어는 C, C++, Java, Python 3, Kotlin입니다. 모든 문제는 출제진이 C++과 Java(혹은 Kotlin)으로 정답을 작성했음이 보장됩니다. 각 언어의 컴파일 옵션과 컴파일러 버전은 아래와 같습니다.

C11. gcc (GCC) 8.3.0

컴파일 `gcc Main.c -o Main -O2 -Wall -lm -static -std=c99 -DONLINE_JUDGE -DBOJ`

실행 `./Main`

C++17. gcc (GCC) 8.3.0

컴파일 `g++ Main.cc -o Main -O2 -Wall -lm -static -std=gnu++17 -DONLINE_JUDGE -DBOJ`

실행 `./Main`

Java. OpenJDK Runtime Environment (build 1.8.0_222-8u222-b10-1ubuntu1 16.04.1-b10)

컴파일 `javac -J-Xms1024m -J-Xmx1024m -J-Xss512m -encoding UTF-8 Main.java`

실행 `java -Xms1024m -Xmx1024m -Xss512m -Dfile.encoding=UTF-8 Main`

Java 11. OpenJDK Runtime Environment (build 13+33)

컴파일 `javac -J-Xms1024m -J-Xmx1024m -J-Xss512m -encoding UTF-8 Main.java`

실행 `java -Xms1024m -Xmx1024m -Xss512m -Dfile.encoding=UTF-8 Main`

Python 3. Python 3.7.4

컴파일 `python3 -c "import py_compile; py_compile.compile(r'Main.py')"`

실행 `python3 Main.py`

PyPy3. PyPy 7.0.0 with GCC 6.2.0 20160901 (Python 3.5.3)

컴파일 `python3 -c "import py_compile; py_compile.compile(r'Main.py')"`

실행 `pypy3 Main.py`

Kotlin/JVM. kotlinc-jvm 1.3.50 (JRE 1.8.0_201-b09)

컴파일
```
kotlinc-jvm -J-Xms1024m -J-Xmx1024m -J-Xss512m -include-runtime -d \
Main.jar Main.kt
```

실행 `java -Xms1024m -Xmx1024m -Xss512m -jar Main.jar`

Kotlin/Native. kotlinc-native 1.3.50 (JRE 1.8.0_201-b09)

컴파일 `kotlinc-native -o Main -opt Main.kt`

실행 `./Main.kexe`

- 네트워크 사용은 금지됩니다. 단, 솔루션을 제출하거나 언어 레퍼런스를 확인하는 것은 가능합니다. 접속이 허용된 사이트의 예는 다음과 같습니다.

C/C++. `https://en.cppreference.com/w/`

Java. `https://docs.oracle.com/javase/8/docs/api/`

Python. `https://docs.python.org/3/`

Kotlin. `https://kotlinlang.org/docs/reference/`

- 대회 종료 전에 퇴실할 수 없습니다.
- 책이나 개인이 준비한 인쇄된 참고자료를 열람할 수 있습니다. 참고자료의 양에는 제한이 없습니다.
- 대회 중에 휴대폰 및 전자기기는 사용할 수 없습니다.

문제 목록

인쇄되어 있는 문제가 총 8문제가 맞는지 확인하시기 바랍니다.

A solved.ac

B 마인크래프트

C 카드 놓기

D 로봇 조립

E 분수

F 대농부 김상혁

G 문자열 압축

H 평행우주

모든 문제의 메모리 제한은 1GB로 동일합니다.

A. solved.ac

시간 제한: 1초

solved.ac는 Sogang ICPC Team 학회원들의 알고리즘 공부에 도움을 주고자 만든 서비스이다. 지금은 서강대뿐만 아니라 수많은 사람들이 solved.ac의 도움을 받아 알고리즘 공부를 하고 있다.

ICPC Team은 백준 온라인 저지에서 문제풀이를 연습하는데, 백준 온라인 저지의 문제들에는 난이도 표기가 없어서, 지금까지는 다양한 문제를 풀어 보고 싶더라도 난이도를 가늠하기 어려워 무슨 문제를 풀어야 할지 판단하기 곤란했기 때문에 solved.ac가 만들어졌다. solved.ac가 생긴 이후 전국에서 200명 이상의 기여자 분들께서 소중한 난이도 의견을 공유해 주셨고, 지금은 약 7,000문제에 난이도 표기가 붙게 되었다.

어떤 문제의 난이도는 그 문제를 푼 사람들이 제출한 **난이도 의견**을 바탕으로 결정한다. 난이도 의견은 그 사용자가 생각한 난이도를 의미하는 정수 하나로 주어진다. solved.ac가 사용자들의 의견을 바탕으로 난이도를 결정하는 방식은 다음과 같다.

- 아직 아무 의견이 없다면 문제의 난이도는 0으로 결정한다.
- 의견이 하나 이상 있다면, 문제의 난이도는 모든 사람의 난이도 의견의 30% 절사평균으로 결정한다.

절사평균이란 극단적인 값들이 평균을 왜곡하는 것을 막기 위해 가장 큰 값들과 가장 작은 값들을 제외하고 평균을 내는 것을 말한다. 30% 절사평균의 경우 위에서 15%, 아래에서 15%를 각각 제외하고 평균을 계산한다. 따라서 20명이 투표했다면, 가장 높은 난이도에 투표한 3명과 가장 낮은 난이도에 투표한 3명의 투표는 평균 계산에 반영하지 않는다는 것이다.

제외되는 사람의 수는 위, 아래에서 각각 반올림한다. 25명이 투표한 경우 위, 아래에서 각각 3.75명을 제외해야 하는데, 이 경우 반올림해 4명씩을 제외한다.

마지막으로, 계산된 평균도 정수로 반올림된다. 절사평균이 16.7이었다면 최종 난이도는 17이 된다.

사용자들이 어떤 문제에 제출한 난이도 의견 목록이 주어질 때, solved.ac가 결정한 문제의 난이도를 계산하는 프로그램을 작성하시오.

입력

첫 번째 줄에 난이도 의견의 개수 n이 주어진다. ($0 \le n \le 3 \times 10^5$)

이후 두 번째 줄부터 $1+n$번째 줄까지 사용자들이 제출한 난이도 의견 n개가 한 줄에 하나씩 주어진다. 모든 난이도 의견은 1 이상 30 이하이다.

출력

solved.ac가 계산한 문제의 난이도를 출력한다.

예제 1

입력	출력
5 1 5 5 7 8	6

5명의 15%는 0.75명으로, 이를 반올림하면 1명이다. 따라서 solved.ac는 가장 높은 난이도 의견과 가장 낮은 난이도 의견을 하나씩 제외하고, {5, 5, 7}에 대한 평균으로 문제 난이도를 결정한다.

예제 2

입력	출력
10 1 13 12 15 3 16 13 12 14 15	13

B. 마인크래프트

시간 제한: 1초

팀 레드시프트는 대회 준비를 하다가 지루해져서 샌드박스 게임인 '마인크래프트'를 켰다. 마인크래프트는 $1 \times 1 \times 1$(세로, 가로, 높이) 크기의 블록들로 이루어진 3차원 세계에서 자유롭게 땅을 파거나 집을 지을 수 있는 게임이다.

목재를 충분히 모은 lvalue는 집을 짓기로 하였다. 하지만 고르지 않은 땅에는 집을 지을 수 없기 때문에 땅의 높이를 모두 동일하게 만드는 '땅 고르기' 작업을 해야 한다.

lvalue는 세로 N, 가로 M 크기의 집터를 골랐다. 집터 맨 왼쪽 위의 좌표는 $(0, 0)$이다. 우리의 목적은 이 집터 내의 땅의 높이를 일정하게 바꾸는 것이다. 우리는 다음과 같은 두 종류의 작업을 할 수 있다.

1. 좌표 (i, j)의 가장 위에 있는 블록을 제거하여 인벤토리에 넣는다.
2. 인벤토리에서 블록 하나를 꺼내어 좌표 (i, j)의 가장 위에 있는 블록 위에 놓는다.

1번 작업은 2초가 걸리며, 2번 작업은 1초가 걸린다. 밤에는 무서운 몬스터들이 나오기 때문에 최대한 빨리 땅 고르기 작업을 마쳐야 한다. '땅 고르기' 작업에 걸리는 최소 시간과 그 경우 땅의 높이를 출력하시오.

단, 집터 아래에 동굴 등 빈 공간은 존재하지 않으며, 집터 바깥에서 블록을 가져올 수 없다. 또한, 작업을 시작할 때 인벤토리에는 B개의 블록이 들어 있다. 땅의 높이는 256블록을 초과할 수 없으며, 음수가 될 수 없다.

입력

첫째 줄에 N, M, B가 주어진다. $(0 \le M, N \le 500, 0 \le B \le 6.4 \times 10^7)$

둘째 줄부터 N개의 줄에 각각 M개의 정수로 땅의 높이가 주어진다. $(i+2)$번째 줄의 $(j+1)$번째 수는 좌표 (i, j)에서의 땅의 높이를 나타낸다.

출력

첫째 줄에 땅을 고르는 데 걸리는 시간과 땅의 높이를 출력하시오. 답이 여러 개 있다면 그중에서 땅의 높이가 가장 높은 것을 출력하시오.

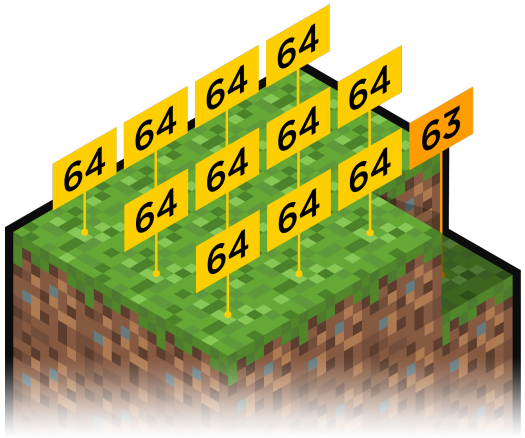
예제 1

입력	출력
3 4 99 0 0 0 0 0 0 0 0 0 0 0 1	2 0

맨 오른쪽 아래의 블록을 제거하면 모두 높이가 0으로 고른 상태가 된다. 따라서 블럭을 한 번 제거하는 시간 2초가 소요된다.

예제 2

입력	출력
3 4 1 64 64 64 64 64 64 64 64 64 64 64 63	1 64



인벤토리에 블록이 하나 있기 때문에, 맨 오른쪽 아래에 블록을 하나 채우면 된다.

예제 3

입력	출력
3 4 0 64 64 64 64 64 64 64 64 64 64 64 63	22 63

인벤토리가 비어 있기 때문에, 맨 오른쪽 아래를 제외한 모든 좌표에서 블록을 하나씩 제거해야 한다.

C. 카드 놓기

시간 제한: 2초

수현이는 카드 기술을 연습하고 있다. 수현이의 손에 들린 카드를 하나씩 내려놓아 바닥에 쌓으려고 한다. 수현이가 쓸 수 있는 기술은 다음 3가지다.

1. 제일 위의 카드 1장을 바닥에 내려놓는다.
2. 위에서 두 번째 카드를 바닥에 내려놓는다. 카드가 2장 이상일 때만 쓸 수 있다.
3. 제일 밑에 있는 카드를 바닥에 내려놓는다. 카드가 2장 이상일 때만 쓸 수 있다.

수현이는 처음에 카드 N장을 들고 있다. 카드에는 1부터 N까지의 정수가 중복되지 않게 적혀 있다. 기술을 N번 사용하여 카드를 다 내려놓았을 때, 놓여 있는 카드들을 확인했더니 위에서부터 순서대로 $1, 2, \cdots, N$이 적혀 있었다!

놀란 수현이는 처음에 카드가 어떻게 배치되어 있었는지 궁금해졌다. 처음 카드의 상태를 출력하여라.

입력

첫 번째 줄에는 N $(1 \le N \le 10^6)$이 주어진다.

두 번째 줄에는 길이가 N인 수열 A가 주어진다. A_i가 x이면, i번째로 카드를 내려놓을 때 x번 기술을 썼다는 뜻이다. A_i는 1, 2, 3 중 하나이며, A_n은 항상 1이다.

출력

초기 카드의 상태를 위에서부터 순서대로 출력하여라.

예제 1

입력	출력
5 1 1 1 1 1	5 4 3 2 1

예제 2

입력	출력
5 2 3 3 2 1	1 5 2 3 4

D. 로봇 조립

시간 제한: 4초

성규는 로봇을 조립해야 한다. 상자 안에는 여러 로봇의 부품들이 섞여 있다. 그런데 어떤 부품이 어느 로봇의 부품인지 표시가 되어있지 않다. 호재는 전자과라서 두 부품을 보면 같은 로봇의 부품인지 알 수 있다. 그래서 성규는 호재의 지시에 따라 부품들을 정리하기로 하였다.

부품들은 1부터 10^6까지의 정수로 표현된다. 그리고 부품 i가 속한 로봇은 robot(i)라고도 표현한다. 예를 들어, 부품 11과 부품 22가 로봇 A의 부품이라고 알고 있는 경우, robot(11)은 로봇 A를 의미하고, robot(22)도 로봇 A를 의미한다.

서로 다른 로봇은 공통 부품을 가지지 않는다. 즉 어떤 부품이 로봇 A의 부품이라면, 로봇 B의 부품은 될 수 없다.

호재는 2가지 지시를 한다.

- 서로 다른 부품 2개를 말해주며, 두 부품은 같은 로봇의 부품이라는 정보를 알려준다.
- 부품 i에 대해서, 지금까지 알아낸 robot(i)의 부품이 몇 개냐고 물어본다.

초기에는 부품에 대한 정보가 존재하지 않는다.

입력

첫 번째 줄에 호재의 지시 횟수 N이 들어온다. $(1 \le N \le 10^6)$

다음 줄부터 N개의 지시가 들어온다.

부품 2개가 같은 로봇의 부품인지 알려줄 때에는 $I\ a\ b$ 의 형태로 들어온다. 부품 a와 부품 b는 같은 로봇의 부품이라는 의미이다. $(1 \le a, b \le 10^6, a \ne b, a, b$는 정수$)$

어떤 로봇의 부품이 몇 개인지 물어볼 때에는 $Q\ c$ 의 형태로 들어온다. 지금까지 알아낸 robot(c)의 부품이 몇 개냐는 의미이다. $(1 \le c \le 10^6, c$는 정수$)$

입력으로 $Q\ c$의 형태가 적어도 한 번 들어온다.

출력

Q로 시작하는 입력에 대해서 한 줄에 하나씩, 지금까지 알아낸 해당 로봇의 부품 개수를 출력한다.

예제

입력	출력
4 I 1 2 I 3 2 Q 1 Q 4	3 1

E. 분수

시간 제한: 1초

임의의 정수 i가 주어졌을 때, 어떤 수의 소수점 위 또는 아래 i째 자리 숫자를 계산하는 알고리즘이 존재한다면, 그러한 수를 "계산 가능한 수"(computable number)라고 한다.

반면에, 그러한 알고리즘이 존재하지 않는다면, 그러한 수를 "계산 불가능한 수"(uncomputable number)라고 한다. 예를 들어, 랜덤으로 생성된 프로그램이 무한 루프를 돌지 않을 확률을 나타내는 카이틴 상수(Chaitin constant)는 계산 불가능한 수이다. 즉, 어떠한 알고리즘으로도 이 수를 임의의 정확도로 계산할 수 없다.

준석이는 분모와 분자가 자연수인 분수는 모두 계산 가능한 수인지 궁금해졌다. 준석이를 도와 a/b의 소수점 아래 i째 자리 숫자를 계산하는 알고리즘을 작성하여 모든 유리수가 계산 가능한 수임을 증명하자.

입력

첫째 줄에 테스트 케이스의 수 T가 주어진다. $(1 \le T \le 100)$

각 테스트 케이스의 첫째 줄에 정수 a와 b가 공백으로 구분되어 주어진다. $(1 \le a, b \le 10^{18})$

각 테스트 케이스의 둘째 줄에 정수 i와 n이 공백으로 구분되어 주어진다. $(1 \le i \le 10^{18}, 1 \le n \le 100)$

출력

각 테스트 케이스마다, a/b를 소수로 표현했을 때, 소수점 아래 i째 자리부터 $(i+n-1)$째 자리까지 n개의 숫자를 출력하라.

예제 1

입력	출력
1 1 7 1 6	142857

$1/7 = 0.\mathbf{142857}142857\cdots$에서 소수점 아래 1번째 자리부터 6개의 숫자는 142857이다.

예제 2

입력	출력
2 1 625 2 7 3 1250 1 9	0160000 002400000

$1/625 = 0.0\mathbf{0160000}0\cdots$ 에서 소수점 아래 2번째 자리부터 7개의 숫자는 0160000이다.

$3/1250 = 0.\mathbf{002400000}\cdots$ 에서 소수점 아래 1번째 자리부터 9개의 숫자는 002400000이다.

예제 3

입력	출력
1 295105340762488656 73766470618827515 165686182540289332 100	7647234964950876597705392951164898875222160879054875613327515544263303837362248466480172179521593664

F. 대농부 김상혁

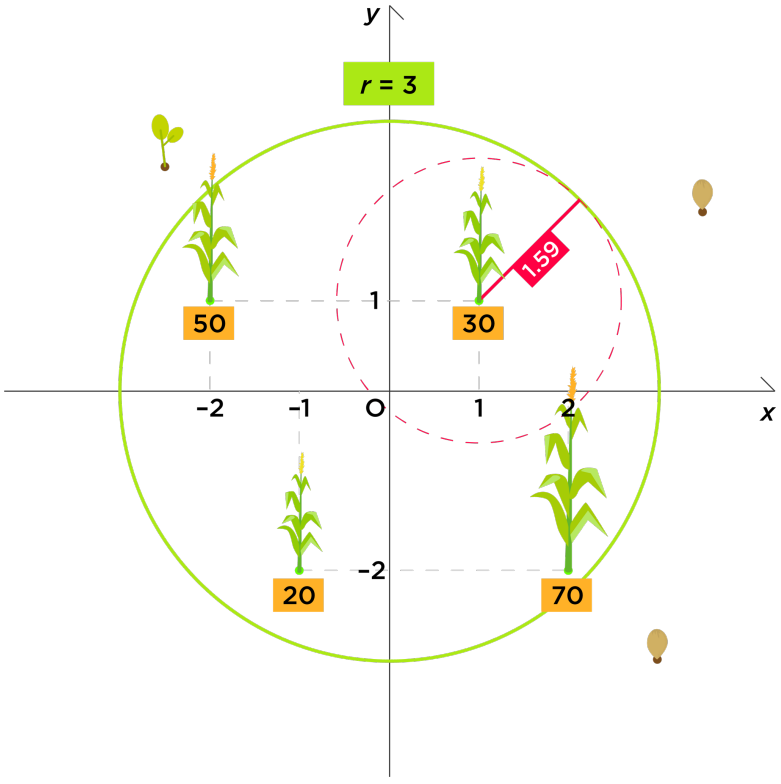
시간 제한: 2초

상혁이는 부농이 되기 위해 낯선 나라의 한 황무지에 정착했다. 상혁이는 황무지에 농작물 N개를 심었다. 이 농작물은 물만 주면 매일매일 수확할 수 있는 신기한 작물이다.

농작물 N개를 심은 뒤에야 상혁이는 황무지를 농토로 개간하는 걸 계획하기 시작했다. 황무지를 2차원 평면으로 생각하자. 상혁이는 $(0, 0)$을 중심으로 반경 r 내의 모든 황무지를 개간하기로 했다. 이 개간된 농토를 관리하는데 매일 $A \times r^2$ 달러가 소비된다.

각 농작물 i는 (x_i, y_i)에 심겨 있고 w_i만큼의 가치를 가지고 있다. 황무지에 심어진 농작물에서는 아무것도 수확할 수 없다. 농토의 경계에 놓인 농작물도 수확할 수 있다. 농토에 심어진 농작물의 수확량은 아래와 같이 계산된다.

농작물로부터 농토 경계까지의 최단 거리를 c_i라고 하면, 농작물마다 매일 $w_i \times c_i$ 달러



위의 예시는 농작물들과 농토를 2차원 좌표평면 상에 나타낸 그림이다. 초록색 원$(r = 3)$은 농토의 경계를 의미한다. 농작물 밑에 나온 수들은 농작물의 가치이다. 이 때 농토 바깥에 있는 농작물들은 황무지에 있으므로 수확되지 않는다.

가치 30인 농작물에서 농토 경계까지의 최단 거리는 위의 빨간 선분으로 표시되어 있고 아래에 그 거리의 값이 소수점 아래 2번째 자리까지 나타나 있다. 이런 구성에서 $A = 10$ 이라 할 때, 관리비용은 $10 \times 3 \times 3 = 90$, 농작물로

부터 얻는 이득은 총 약 113.058936이다. 따라서 이 때 상혁이가 얻는 이득은 약 23.058936달러이다. (최대 이득은 아니다)

A와 농작물들의 좌표, 가치들이 주어졌을 때, 농토 관리 비용과 농작물 총 수확량을 고려할 때 상혁이가 최대로 이득을 보기 위한 r을 결정해 주자.

입력

첫 번째 줄에 정수 N과 정수 A가 주어진다. ($1 \le N \le 10^5$, $1 \le A \le 10^3$)

두 번째 줄부터 $N+1$번째 줄까지, 각 줄마다 정수 x_i, y_i, w_i가 주어진다. ($-10^3 \le x_i, y_i \le 10^3$, $1 \le w_i \le 200$)

출력

상혁이가 황무지를 개간하여 얻을 수 있는 최대의 이득(달러)를 출력한다. 만약 아예 개간하지 않는 것이 최대 이득을 준다면 0을 출력한다.

상대/절대 오차가 10^{-6} 이내인 경우에만 정답이다.

예제 1

입력	출력
4 10 1 1 30 -2 -1 50 -1 -2 20 2 -2 70	325.558936

본문의 예시에 해당하는 입력이다.

예제 2

입력	출력
5 10 1 3 50 -5 -2 80 3 1 40 2 -3 60 8 8 100	659.3779

예제 3

입력	출력
2 20 1 1 10 2 2 20	0

G. 문자열 압축

시간 제한: 2초

알파벳 소문자로만 이루어진 문자열 S가 주어진다. 사전에는 K개의 단어가 사전 순으로 등재되어 있다. 또한, 사전의 각 단어에는 '단어 번호'가 있다. 사전에서 i번째로 등장하는 단어의 번호는 i이다.

우리가 사용할 압축 방법은, 문자열을 사전에 등재된 단어들로 쪼개어, 각 단어를 해당하는 단어 번호로 바꾸는 것이다. 같은 단어를 여러 번 사용해도 된다.

주어진 문자열을 압축했을 때, 결과적으로 나오는 수열의 길이가 가장 짧게 되도록 압축하라.

입력

첫째 줄에 K가 주어진다. ($1 \le K \le 10^4$)

둘째 줄부터 K개의 줄에 걸쳐 1번부터 사전 순으로 사전의 단어가 주어진다. 각 단어는 알파벳 소문자로만 이루어져 있고, 길이는 1자 이상 10^3자 이하이다. 중복된 단어는 주어지지 않는다.

마지막 줄에는 우리가 압축할 문자열 S가 주어진다. S의 길이는 1자 이상 10^5자 이하이다.

출력

첫째 줄에 압축 결과 수열의 길이를 출력한다.

둘째 줄에 수열을 공백으로 구분하여 출력한다. 답이 여러 개라면, 그중에서 사전 순으로 가장 먼저 오는 것을 출력한다.

만약, 압축이 불가능하다면 첫 줄에 `impossible`을 출력한다.

예제 1

입력	출력
8 a abc b c d def e f abcdef	2 2 6

문자열 “abcdef”는 “a” + “b” + “c” + “d” + “e” + “f” 여섯 조각으로 쪼개서 압축할 수도 있고, “abc” + “d” + “e” + “f” 네 조각, 또는 “a” + “b” + “c” + “def” 네 조각으로도 압축할 수 있지만, 사전의 2번째 단어 “abc”와 6번째 단어 “def” 두 조각으로만 쪼개서 압축하는 것이 더 짧다.

예제 2

입력	출력
6 able conceiv in inc ivable once inconceivable	3 3 2 1

“inconceivable”은 “in” + “conceiv” + “able” (3 2 1)과 ”inc” + “once” + “ivable” (4 6 5) 두 방법으로 압축할 수 있지만, 수열 (3 2 1)이 (4 6 5)보다 사전 순으로 먼저 오기 때문에 (3 2 1)을 출력한다.

예제 3

입력	출력
1 peanutbutter bojackhorseman	impossible

“bojackhorseman”은 사전에 있는 단어를 사용하여 압축될 수 없으므로, ”impossible”을 출력한다.

H. 평행우주

시간 제한: 1초

서강 프로그래밍 대회가 열리고 있는 지금도 수많은 별들이 다산관의 하늘을 수놓고 있다.

태한이는 컴퓨터공학과지만, 취미로 별을 연구하고 있다. 이미 평행우주가 존재한다는 것을 증명하는 데 성공한 태한이는 별 연구에 너무 심취한 나머지 모든 평행우주의 모든 별들을 별자리로 만들었다.

별자리는 별 n개와 별을 잇는 선 $n-1$개로 정의되며, 한 별자리에 속한 모든 별들은 연결되어 있다. 태한이는 별자리를 정의할 때 한 평행우주에 같은 위상의 별자리가 있을 수 없도록 정의했다. 한 별자리의 별들의 연결 관계를 바꾸지 않고 위치만을 움직여서 다른 별자리를 만들 수 있다면 두 별자리의 위상이 같다고 말한다.

한편 한나는 우리가 살고 있는 평행우주에 존재하는 모든 별자리를 찍기 위해, 태한이에게서 별자리들의 정보가 담긴 연구 노트를 받았다. 태한이의 연구 노트에는 모든 평행우주의 모든 별자리에 대한 정보가 적혀 있었지만, 안타깝게도 각각의 별자리가 어떤 평행우주에 있는지는 적혀 있지 않았다.

태한이의 연구 노트를 토대로 한나가 찍을 사진의 최대 장 수를 계산해 주자. 한나는 별자리 하나당 한 장의 사진을 찍는다.

입력

첫 번째 줄에는 모든 평행우주에 존재하는 별자리의 총 수 n이 주어진다. ($1 \le n \le 10^6$) 별의 수의 총합은 10^6을 넘지 않는다.

두 번째 줄부터는 각 별자리의 정보가 주어진다.

각 별자리마다, 첫 번째 줄에는 별자리를 구성하는 별의 수 s가 주어진다. $(1 \le s \le 30)$

이후 $s-1$개의 줄에 별들 사이의 연결 관계를 의미하는 두 정수 u, v가 주어진다. $(0 \le u, v < s, u \ne v)$

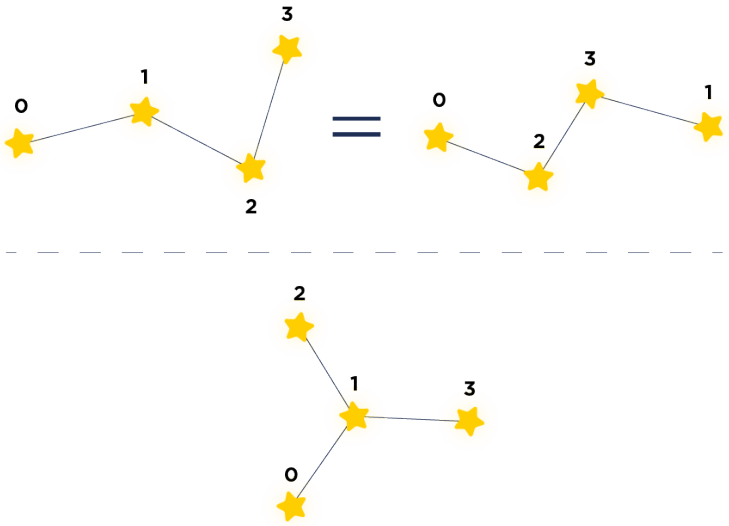
편의를 위해 각 별자리 안에서 별들은 $0, 1, 2, \cdots, s-1$번으로 번호가 붙어 있다고 생각하며, u v는 u번 별과 v번 별이 연결되어 있다는 의미이다. 별자리의 위상을 고려할 때는 별의 번호는 고려하지 않는다.

출력

한나가 찍을 사진의 최대 장 수를 출력한다.

예제

입력	출력
3 4 0 1 1 2 2 3 4 0 2 2 3 3 1 4 0 1 1 2 1 3	2



첫 번째 별자리와 두 번째 별자리는 같은 위상을 가지므로, 우리 평행 우주에는 첫 번째 별자리와 두 번째 별자리가 동시에 존재할 수 없다.

세 번째 별자리는 첫 번째와 두 번째 별자리와 다른 위상을 가진다. 따라서 우리 평행 우주에 존재할 수 있는 별자리는 최대 2개이다.