2021 신촌지역
대학생 프로그래밍 대회 동아리
연합 여름 대회

SUAPC 2021 summer

Official Problemset

ICPC Sinchon

Sogang ICPC Team Algos Morgorithm EDOC HI-ARC

Sponsors

kakao HYUNDAI AutoEver NAVER D2 FURIOSA STARTLINK SOLVED.AC 한빛미디어 Younglin.com 이지스퍼블리싱

문제 목록

문제지에 있는 문제가 총 12문제가 맞는지 확인하시기 바랍니다.

A 휴먼 파이프라인

B 선인장의 독립집합

C 조각 체스판

D 반짝반짝 2

E 문자열 조작의 달인

F Flat Earth

G 기지국 업그레이드

H 재활용 캠페인

I 브런치북

J 사이클

K 수요응답형 버스

L 다꾸

모든 문제의 메모리 제한은 1GB로 동일합니다.

문제 A. 휴먼 파이프라인

시간 제한	1.5 초
메모리 제한	1024 MB

오늘은 중요한 날이다. SUAPC가 있는 날이기 때문이다.

이렇게 중요한 날이지만 안타깝게도 일을 해야 한다. 오늘 해야 할 일은 상자 K개를 적절한 곳으로 옮겨야 하는 일이다.

상자 K개는 너무 많아서 아무래도 혼자서 전부 나를 수는 없기 때문에, N명의 SUAPC 참가자들이 상자를 나르기 위해 모여 있다. N명 모두가 일을 최대한 빠르게 마치고 SUAPC에 참가하고 싶어한다.

참가자들은 두 팀으로 나눠져서 작업을 진행하기로 했다. 두 팀이 같은 수의 상자를 옮길 필요는 없다. 두 팀 모두 적어도 한 명은 포함되어야 한다. 각 사람의 분당 작업 속도는 v_i며 팀의 작업 속도는

$$(\text{해당 팀에 속한 사람들의 작업 속도 중 가장 느린 작업 속도}) \times (\text{팀에 속한 사람의 수})$$

이다. 상자 K개를 옮기는 팀의 분당 작업 속도가 v일 때, 팀이 작업을 마치는 데에는 $\lceil \frac{K}{v} \rceil$분이 걸린다.

모두가 행복하게 SUAPC에 참가할 수 있게, 모든 상자를 최대한 빠르게 옮길 수 있도록 N명을 적절히 두 팀으로 나누어 두 팀이 동시에 상자를 옮기기 시작했을 때 제일 빨리 끝나는 경우의 시간을 구하자.

입력

다음과 같이 입력이 주어진다.

N K
v_1 v_2 $\cdots$ v_N

- N은 모인 사람의 수다. $(2 \le N \le 200\,000)$
- K는 옮겨야 하는 상자의 개수이다. $(1 \le K \le 10^{18})$
- v_i는 i번째 사람의 분당 작업 속도이며, 1분에 상자 v_i개를 옮길 수 있다는 뜻이다. $(1 \le v_i \le 10^9)$
- 입력으로 주어지는 모든 수는 정수다.

출력

모든 상자를 최대한 빠르게 옮기는 경우의 작업 시간을 분 단위로 출력한다.

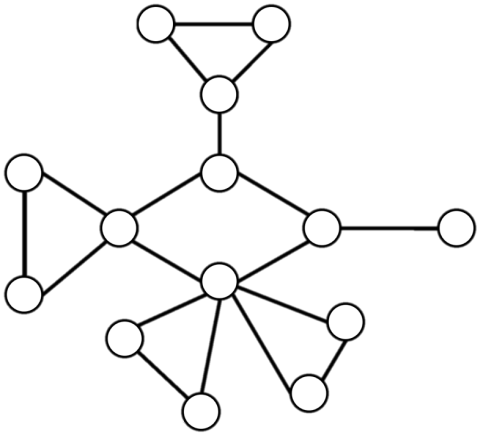
입출력 예시

표준 입력(stdin)	표준 출력(stdout)
5 100 3 1 2 4 5	10
2 15600000 500 1000	10400

이 페이지는 공백입니다

문제 B. 선인장의 독립집합

시간 제한	1 초
메모리 제한	1024 MB



무방향 그래프 $G(V,E)$에서 정점의 부분집합 S에 속한 모든 정점 쌍이 서로 인접하지 않으면 S를 독립 집합(independent set)이라고 한다. 임의의 그래프에 대한 최대 독립 집합 문제는 NP-Complete 문제로 다항시간에 해결할 수 있는지 없는지 아직 밝혀지지 않았다.

선인장 그래프(cactus graph)는 모든 간선이 최대 한 개의 사이클에 속한 연결된 무방향 그래프다. 즉, 임의의 서로 다른 두 사이클이 최대 하나의 공통 정점을 가지는 무방향 연결 그래프를 의미한다.

선인장 그래프에서의 최대 독립 집합을 구해보자.

입력

다음과 같이 입력이 주어진다.

$N\ M$
$u_1\ v_1$
$\vdots$
$u_M\ v_M$

- N은 정점의 개수이고 M은 간선의 개수이다. $(1 \le N \le 100000, N-1 \le M \le 150000)$
- $u_i\ v_i$는 정점 u_i와 v_i를 잇는 간선이 존재한다는 뜻이다. $(1 \le u, v \le N, u \ne v)$
- 입력으로 주어지는 수는 모두 정수다.

출력

첫 번째 줄에 주어진 선인장 그래프에 대한 최대 독립 집합의 크기를 출력한다.

두 번째 줄에 최대 독립 집합에 속한 정점 번호를 크기 순으로 출력한다. 만약 가능한 최대 독립 집합이 여러 가지인 경우에는 아무거나 출력하면 된다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
10 12 1 2 1 5 1 8 2 3 3 4 4 1 5 6 6 7 7 1 8 9 9 10 10 1	6 2 4 5 7 8 10
9 10 1 2 2 3 3 4 4 5 5 1 1 6 6 7 7 8 8 9 9 1	4 2 4 6 8

표준 입력(stdin)	표준 출력(stdout)
15 17 1 2 2 3 3 1 3 4 4 5 5 6 6 7 6 8 8 9 9 3 2 10 10 11 11 12 12 13 12 14 14 15 15 2	7 3 5 7 8 11 13 15

이 페이지는 공백입니다

문제 C. 조각 체스판

시간 제한	2 초
메모리 제한	1024 MB

높이 N, 너비 M의 정사각형 격자에 검은색과 흰색 중 한 가지 색이 칠해져 있다. 머릿속이 체스로 가득찬 현채는 문득 이 격자를 잘랐을 때 체스판이 되는 경우가 몇 가지인지 궁금해졌다.

체스판은 검은색과 흰색이 번갈아 칠해져 있어야 한다. 구체적으로, 각 칸이 검은색과 흰색 중 하나로 색칠되어 있고, 변을 공유하는 두 개의 사각형은 다른 색으로 칠해져 있어야 한다. 체스판의 모양은 정사각형이어야 하며, 꼭 8×8일 필요는 없다.

현채의 궁금증을 해결해 주자.

입력

다음과 같이 입력이 주어진다.

$$N\ M$$
$$C_{1,1}C_{1,2}\cdots C_{1,M}$$
$$C_{2,1}C_{2,2}\cdots C_{2,M}$$
$$\vdots$$
$$C_{N,1}C_{N,2}\cdots C_{N,M}$$

- $1 \le N, M \le 3000$
- $C_{i,j}$는 (i, j)에 칠해진 색상을 의미하는 문자이며 `B`와 `W` 중 하나이다. $C_{i,j}$가 `B`라면 (i, j)에 검은색이, `W`라면 (i, j)에 흰색이 칠해져 있다.

출력

주어진 격자를 잘랐을 때 체스판이 되는 경우가 몇 가지인지 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
3 3 BWB WBB BWB	11

이 페이지는 공백입니다

문제 D. 반짝반짝 2

시간 제한	1 초
메모리 제한	1024 MB

국렬이는 지난 겨울에 수현이가 크리스마스 트리를 장식하고 남은 전구 스트립을 훔쳐서 자신의 방을 장식하려고 한다.

전구 스트립에는 전구 N개가 일(一)자로 설치되어 있다. 전구들은 전원을 넣었을 때 켜질 확률이 제각각이다.

전자공학의 고수인 국렬이는 어떤 전구가 고장 나는 것이 다른 전구에 영향을 미치지 않도록 회로를 고쳤다. 그래도 그다지 반짝반짝하지 않았다고 생각했는지 전구 스트립의 모든 이웃한 두 전구 사이에 추가 전구를 하나씩 달았다. 이 추가 전구들은 이웃한 두 전구 중 **하나만이** 켜졌을 때 불이 켜진다.

전구 스트립에 불이 들어오는 전구 개수의 기댓값을 구하여라.

입력

다음과 같이 입력이 주어진다.

N

$p_1 \cdots p_N$

- N은 전구의 개수이다. $(1 \le N \le 100\,000)$
- p_i는 i번째 전구에 불이 들어올 확률이며, 정확히 소수점 아래 두 자리까지 주어진다. $(0 \le p_i \le 1)$

출력

불이 들어온 전구의 개수의 기댓값을 출력한다.

출력한 값과 정답과의 절대 오차 또는 상대 오차가 10^{-6} 이하여야 한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
2 0.50 0.50	1.5
3 0.30 0.40 0.50	2.16

노트

첫 번째 입출력 예시에서 발생할 수 있는 모든 경우는 다음과 같다.

- $(1-0.5)\times(1-0.5)=0.25$의 확률로 모든 전구에 불이 들어오지 않는다.
- $0.5\times(1-0.5)=0.25$의 확률로 첫 번째 전구와 추가 전구에 불이 들어온다.
- $(1-0.5)\times0.5=0.25$의 확률로 두 번째 전구와 추가 전구에 불이 들어온다.

- $0.5 \times 0.5 = 0.25$의 확률로 첫 번째 전구와 두 번째 전구에 불이 들어온다.

불이 들어오는 전구의 개수의 기댓값은 $0.25 \times 0 + 0.25 \times 2 + 0.25 \times 2 + 0.25 \times 2 = 1.5$다.

문제 E. 문자열 조작의 달인

시간 제한	2.5 초
메모리 제한	1024 MB

소문자 알파벳으로 이루어진 길이 N의 문자열 S가 있다. 문자열을 자유자재로 다루는 달인 Taro는 여기에 다음과 같은 조작을 M번 가하려고 한다.

- 위치 $1 \leq i \leq N$을 하나 골라서, S_i를 알파벳 순서로 다음에 오는 문자로 바꾼다.
  - 단, 고른 문자가 z라면 조작을 가하더라도 z가 된다.

예를 들어 az라는 문자열이 존재한다고 했을 때, $i = 1$을 고르면 bz로 바뀌지만 $i = 2$를 고르면 문자열이 바뀌지 않는다.

이렇게 조작을 M번 가했을 때 나올 수 있는 문자열의 개수를 구하자.

입력

다음과 같이 입력이 주어진다.

```
N M
S
```

- $1 \leq N \leq 300$, $0 \leq M \leq 10^{18}$
- 입력으로 주어지는 문자열 S는 알파벳 소문자만으로 이루어져 있다.

출력

주어진 문자열에 조작을 M번 가했을 때 나올 수 있는 문자열의 개수를 $10^9 + 7$로 나눈 나머지를 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
2 2 az	3
2 2 ay	3

노트

첫 번째 입출력 예시에서 나올 수 있는 문자열은 az, bz, cz의 3개이다.

두 번째 입출력 예시에서 나올 수 있는 문자열은 cy, bz, az의 3개이다.

이 페이지는 공백입니다

문제 F. Flat Earth

시간 제한	1 초
메모리 제한	1024 MB

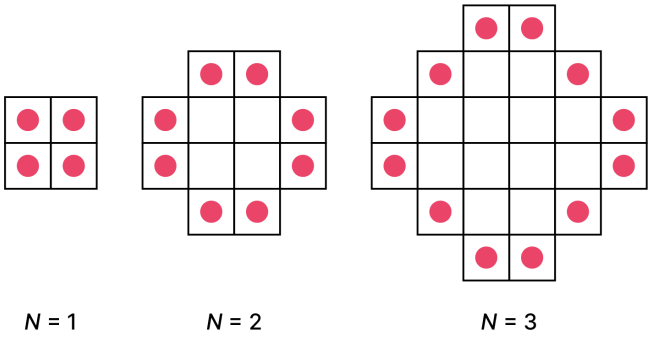
지구는 평평하다. 이를 굳게 믿고 있는 해성이는 지구의 끝으로 가 보려고 한다.

지구는 아래와 같이 칸으로 구분되는 모양을 가지며 크기 N을 가진다.

지구의 크기 N이 1일 때는 정사각형 모양으로 4개의 칸이 존재한다. 지구의 크기가 1 커질 때마다 지구의 끝과 인접한 비어있는 공간에 칸이 하나씩 생겨난다.

지구의 크기 N이 i일 때 지구의 끝은 지구의 크기가 i가 되면서 새로 생긴 칸들을 말한다.

$N = 1$일 때는 모든 칸이 지구의 끝이다.



위 그림에서 동그라미 친 곳이 $N = 1, N = 2, N = 3$일 때의 지구의 끝이다.

해성이는 1초에 1칸씩 움직일 수 있다. 하지만 지구의 크기도 1초에 1 씩 커지기 때문에 이대로는 지구의 끝에 도달할 수 없다는 사실을 깨달은 해성이는 현자인 당신에게 도움을 요청했다.

이를 불쌍히 생각한 당신은 1초에 2칸씩 움직일 수 있는 자동차를 만들어 줬다. 슬프게도 무한동력 배터리가 아직 구현되지 않은 세상이기 때문에 자동차는 K초 동안만 움직일 수 있다.

해성이가 출발할 때의 지구 크기 N과 자동차가 움직일 수 있는 시간 K가 주어질 때, 지구의 끝에 도달할 수 있는 출발칸의 개수를 계산하자.

입력

다음과 같이 입력이 주어진다.

```
T
N1 K1
...
NT KT
```

- 첫 줄에 테스트 케이스의 개수 T가 주어진다. $(1 \le T \le 1000)$
- 2번째 줄 부터 T 줄에 걸쳐 한 줄에 테스트 케이스 하나가 주어진다.

- N은 현재 지구의 크기다. $(1 \le N \le 10^9)$
- 1초에 2칸씩 움직이는 자동차를 K초간 사용할 수 있다. $(1 \le K \le 10^9)$

출력

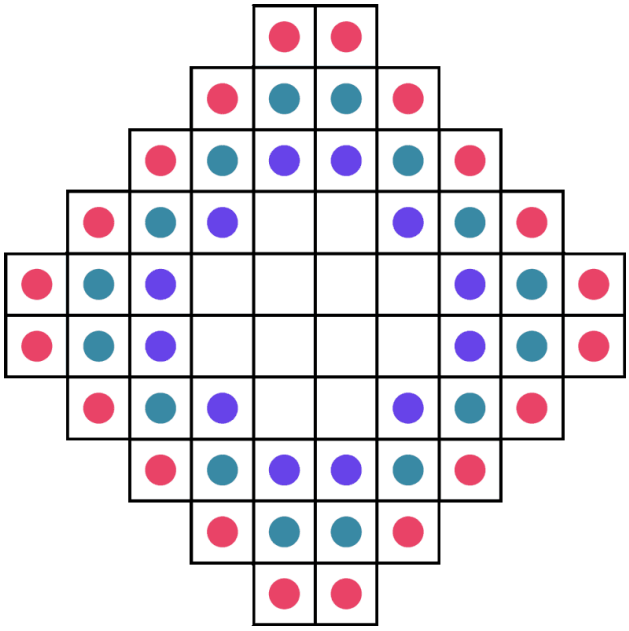
각 테스트 케이스별로 테스트 케이스가 주어진 순서대로 지구의 끝에 도달할 수 있는 칸의 수를 한 줄에 출력하여 총 T줄에 걸쳐 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
2 5 2 3 0	48 12

노트

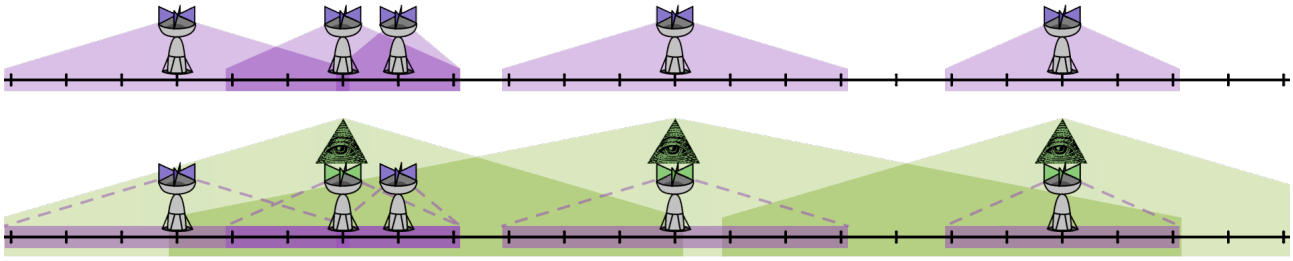
첫 번째 테스트 케이스에서 지구의 끝에 도달 가능한 칸은 아래의 그림에서 색칠된 48칸이다.



두 번째 테스트 케이스에서는 자동차를 움직일 수 없기 때문에 처음에 지구의 끝에 있지 않았다면 지구의 끝에 도달할 수 없다. 문제의 그림에서 볼 수 있듯이 $N = 3$일 때 지구의 끝은 총 12칸이 존재한다.

문제 G. 기지국 업그레이드

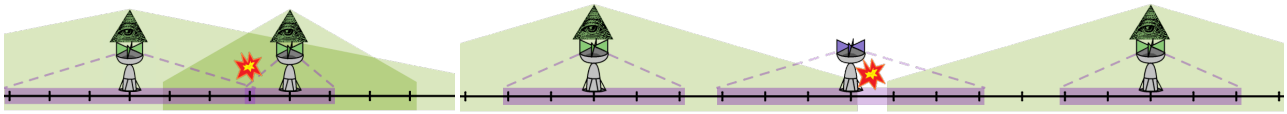
시간 제한	3 초
메모리 제한	1024 MB



천재 과학자 토끼가 사는 도시에는 동서로 길게 뻗어있는 도로가 있고, 이 도로를 따라서 N 개의 기지국이 세워져 있다. 이 기지국은 도로에서의 원활한 통신을 도와준다. 편의상 도로 위에 있는 시청에서 동쪽으로 X 만큼 떨어진 곳을 위치 X라고 하자($X < 0$인 경우에는, 서쪽으로 $-X$ 만큼 떨어져 있다.) i 번 기지국은 위치 X_i 에 있으며, 크기가 H_i이다. 기존 1세대 통신 방법을 사용하면 i 번 기지국으로부터 거리가 H_i 이하인 모든 위치에 1세대 통신 방법으로 통신을 할 수 있다. 즉 p가 폐구간 $[X_i - H_i, X_i + H_i]$ 안에 속해있으면 위치 p에서 i 번 기지국과 1세대 통신 방법으로 통신을 할 수 있다.

토끼는 새로운 2세대 통신 방법을 개발했다. 이 통신 방법을 사용하면 대용량 통신을 할 수 있고, 통신 거리가 3배씩 늘어서 i 번 기지국과 거리가 $3H_i$ 이하인 모든 위치에 2세대 통신 방법으로 통신을 할 수 있다. 즉 p가 폐구간 $[X_i - 3H_i, X_i + 3H_i]$ 안에 속해있으면 위치 p에서 i 번째 기지국과 2세대 통신 방법으로 통신을 할 수 있다. 2세대 통신 방법에는 치명적인 문제가 있는데, 기지국이 너무 촘촘하게 있으면 서로 간섭을 일으킬 수 있다. 어떤 위치가 2개 이상의 기지국과 1세대 통신 방법으로 통신을 할 수 있을 정도로 가까이 있으면, 해당 위치에서 전파 간섭이 일어난다. 다시 말하면, $|X_j - X_k| \le H_j + H_k$인 경우에 j 번 기지국과 k 번 기지국이 담당하는 범위가 겹쳐서 전파 간섭이 일어난다.

이 도시의 시장 로얄은, 기존에 있는 기지국 중 일부를 2세대 통신으로 업그레이드하고, 나머지 기지국을 철거하는 방식으로 2세대 통신을 지원하려고 한다. 이때, 업그레이드한 기지국끼리는 서로 전파 간섭이 일어나서는 안 되고, 하위 호환성을 위해 기존의 1세대 통신 방법을 사용하여 통신을 할 수 있었던 위치에서 2세대 통신 방법을 사용하여 통신할 수 있어야 한다. 다시 말해서, 위치 x에 대해 $x \in [X_i - H_i, X_i + H_i]$인 i가 존재하면, 2세대 통신으로 업그레이드한 j 번 기지국이 존재해서 $x \in [X_j - 3H_j, X_j + 3H_j]$를 만족해야 한다. 로얄을 도와서 어떤 기지국을 2세대 통신으로 업그레이드해야 하는지 계산해주자. **업그레이드할 기지국의 개수를 최소화할 필요가 없음에 유의하자.**



(a) 전파 간섭으로 인해 업그레이드 불가 (b) 하위 호환성을 만족하지 않는 상황

조건을 만족하지 않는 2세대 업그레이드 예시

입력

다음과 같이 입력이 주어진다.

N
X_1 H_1
...
X_N H_N

- N은 기지국의 개수이다. $(1 \leq N \leq 500\,000)$
- X_i는 i 번 기지국의 위치이다. $(-10^{18} \leq X_i \leq 10^{18})$
- H_i는 i 번 기지국의 높이이다. $(1 \leq H_i \leq 10^{18})$
- 입력으로 주어지는 모든 수는 정수다.

출력

첫째 줄에 2세대 통신으로 업그레이드할 기지국의 수 M을 출력한다. 그다음 줄에, 2세대 통신으로 업그레이드할 기지국의 번호 M 개를 공백으로 구분하여 출력한다. 답이 여러 가지면 그 중 아무거나 하나를 출력한다.

만약 문제의 조건을 만족하면서 업그레이드할 기지국을 고르는 것이 불가능하다면, 대신에 첫 번째 줄에 -1을 출력한다.

입출력 예시

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

두 기지국을 모두 업그레이드하면 전파 간섭이 일어나기에, 두 기지국 모두를 업그레이드할 수는 없다.

기존에 1세대 통신 방법으로 위치가 -5 이상 6 이하인 부분에 통신이 가능했고, 2번 기지국을 업그레이드 한 이후에 2세대 통신 방법으로는 위치가 -6 이상 12 이하인 부분에 통신이 가능하기 때문에, 기존에 1세대 통신으로 통신이 가능한 모든 위치에서 2세대 통신을 할 수 있다.

표준 입력(stdin)	표준 출력(stdout)
5 0 3 3 2 4 1 9 3 16 2	3 2 4 5

문제 H. 재활용 캠페인

시간 제한	1 초
메모리 제한	1024 MB



고등학생 때였다. 친구의 손에 이끌려 처음으로 가게 된 화장품 가게는 문을 열자마자 은은한 향기가 코끝을 스치는 곳이었다. 친구를 따라가기에 급급했던 한별이의 발걸음은 이제 누가 말하지 않아도 무언가에 빨려 들어간 듯이 앞을 향했다. 파운데이션을 바르고, 블러셔를 두드리고. 거울을 본 한별이는 처음 보는 자신의 모습에 푹 빠져버렸다. 얼굴에 띈 홍조는 블러셔가 무색해질 정도였다. 그런 기분에 감화된 탓인지 화장품 가게를 나서서 집에 돌아갈 때까지도 거울의 그 모습을 잊을 수가 없었다. 마치 자기가 주목받는 느낌이 들어 볼이 한번 더 붉어져 왔다. 한별이는 이 기분을 다른 사람에게도 전해주고 싶어서 나중에 꼭 화장품 가게를 열겠다고 다짐했다.

그동안 이룬 결실도, 못다 한 각오도 있었다. 하지만 화장품 가게를 열겠다는 노력의 결과는 눈앞으로 다가와서, 내일은 한별이의 화장품 가게가 개점한 지 꼬박 1년 되는 날이 된다. 한별이의 화장품 가게는 만들어진 지 얼마 안 되었지만 많은 인기를 얻어 멀리서까지도 화장품을 사러 찾아온다. 이 화장품 가게의 인기 상품은 총 용량이 $X\,\mathrm{m}\ell$인 헤어에센스이고, 특유의 매력적인 딸기향은 모발의 상처만큼이나 마음의 상처를 치유하는 데에도 유용하다.

한별이의 화장품 가게에서는 불필요한 쓰레기를 줄이자는 재활용 캠페인을 하고 있다. 가게의 모든 상품은 재활용이 가능한 용기에 담긴다. 가게로 사용하다 남은 헤어에센스 용기 두 개를 반납하면 새로운 용기에다가 남은 헤어에센스를 모아 주고, 추가로 총 용량의 절반만큼의 헤어에센스를 추가로 채워준다. 단, 총 용량을 넘쳐서 채워주지는 않는다. 다시 말해, 용량이 각각 $A\,\mathrm{m}\ell$와 $B\,\mathrm{m}\ell$ 남은 헤어에센스를 가져가면 $\min\left(A+B+\frac{X}{2},\ X\right)\,\mathrm{m}\ell$의 헤어에센스가 담긴 새로운 용기로 바꿔준다.

한별이의 화장품 가게 단골인 히나는 이제까지 사모은 헤어에센스 용기가 N 개 있다. i 번째 용기에는 헤어에센스가 $C_i\,\mathrm{m}\ell$ 담겨 있다. 히나는 한별이의 화장품 가게에 적당한 순서로 헤어에센스를 교환해서 용량이 꽉 찬, 즉, $X\,\mathrm{m}\ell$가 담겨 있는 헤어에센스 용기를 최대한 많이 만들고 싶다. 히나가 용량이 꽉 찬 헤어에센스를 최대 몇 개 만들 수 있는지 알려주자.

입력

다음과 같이 입력이 주어진다.

$N\ X$
$C_1\ C_2\ \dots\ C_N$

- N은 히나가 가진 헤어에센스 용기의 수이다. $(1 \le N \le 100\,000)$
- X는 헤어에센스 용기의 총 용량이다. $(1 \le X \le 10^{18})$
- C_i는 i 번째 용기에 담겨 있는 헤어에센스의 용량이 $C_i\,\mathrm{m}\ell$라는 의미이다. $(0 \le C_i \le X)$
- 입력으로 주어지는 모든 수는 정수다.

출력

한별이의 화장품 가게에 가서 적당한 순서로 헤어에센스를 교환해서 용량이 꽉 찬 헤어에센스 용기를 최대 몇 개 만들 수 있는지 출력하여라.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
7 13 0 1 2 3 5 8 13	3

첫 번째 용기와 두 번째 용기를 가져가서 용량이 $\left(0+1+\frac{13}{2}\right)\mathrm{m}\ell = 7.5\,\mathrm{m}\ell$ 남은 용기를, 세 번째 용기와 네 번째 용기를 가져가서 용량이 $\left(2+3+\frac{13}{2}\right)\mathrm{m}\ell = 11.5\,\mathrm{m}\ell$ 남은 용기를, 다섯 번째 용기와 여섯 번째 용기를 가져가서 용량이 $13\,\mathrm{m}\ell$ 남은 헤어 에센스 용기를 받을 수 있다. 마지막으로 용량이 $7.5\,\mathrm{m}\ell$ 남은 용기와 $11.5\,\mathrm{m}\ell$ 남은 용기를 가져가서 용량이 $13\,\mathrm{m}\ell$ 남은 헤어 에센스 용기를 하나 더 받을 수 있다.

문제 I. 브런치북

시간 제한	1 초
메모리 제한	1024 MB

You can make anything by writing.

- C.S.Lewis

좋은 글은 수많은 사람에게 영향을 미치고, 시간이 지나 다시 읽어도 그 가치가 오롯이 살아있다. 카카오에서 운영하는 "브런치"는 좋은 글을 쓰고 싶은 모든 이들을 위해 준비한 서비스이다.

브런치는 언제 어디서나 글감을 기록하고, 어디에서나 멋지게 글을 읽을 수 있도록 모든 디바이스에서 글을 작성, 수정하고 읽을 수 있다. 이렇게 작성한 글 들을 주제별로, 혹은 같은 관심사를 다른 작가와 함께 묶어서 연재할 수 있는 "매거진"을 만들 수 있다. 매거진에 펼쳐둔 아이디어를 섬세하게 다듬어서 "브런치북"을 만들어 자신만의 고유한 원작을 만들 수 있고, 브런치팀은 이 원작이 책으로, 강연으로, 또 새로운 2차 저작물로 재탄생하는 과정을 지원해 준다.

능력 있는 프로그래머이면서 동시에 심금을 울리는 글솜씨로 모두를 사로잡는 시인 종서도 최신 트렌드에 맞춰 브런치에 자신의 시를 올리기로 했다. 글을 읽어본 희원은 종서가 브런치북으로 시집을 출판했으면 좋겠다고 생각해서, T 개의 시를 선택했다. 종서가 브런치북을 작업하다가 희원이 예상외의 시를 선택했다는 생각이 들어서 얘기해 본 결과, 희원이 시를 다운로드 받은 후 사용하는 제목 정렬 방식이 종서의 환경과 다르다는 것을 알았다.

예를 들어서, a 번째 책의 b 번째 시에 `Book`a`Poem`b와 같은 제목이 붙어있다고 하자. 2번째 책의 10번째 시에는 `Book2Poem10`, 10번째 책의 1번째 시에는 `Book10Poem1`이란 제목이 붙을 것이다. 종서가 사용한 일반적으로 프로그래밍에서 사용하는 문자열 비교는 앞에서부터 시작해서 비교해가며, 서로 다른 문자가 나왔을 때 해당 문자의 비교로 문자열 비교를 한다. 여기서는 처음으로 다른 문자가 등장하는 5번째 문자에서 `2`가 `1`보다 크기 때문에, `Book2Poem10`이 `Book10Poem1`보다 더 크다. 하지만 각 시가 2번째와 10번째 책의 시라는 사실을 알고 있으면, 10번째 책의 시가 더 크도록 비교해야 자연스러울 것이다.

그래서 희원은 다른 곳에서 많이 사용하는 자연스러운 문자열 비교를 사용해서 책의 제목을 정렬했다. 해당 문자열 비교에서는 먼저 대문자가 있는 경우에 모두 소문자로 고친다. 그 후, 연속된 숫자를 묶어서 수로 생각해서 비교한다. 앞에서부터 문자를 확인하면서 양쪽이 모두 숫자면 해당 숫자가 속해있는 수를, 그렇지 않으면 두 문자를 사용해서 비교한다. 비교해야할 수가 `0`으로 시작하는 수의 경우에, 두 수의 앞에 있는 `0`을 제외하고 같은 수라면 `0`의 개수가 많은 문자열이 더 작다. 다음은 해당 문자열 비교를 사용하여 제목을 비교한 예이다: `005c` < `05bc` < `5abc` < `5bcd` < `006a` < `6abc` < `10ab` < `abcd`.

종서는 X 번째 날에 16^X 개의 시를 썼고, 각 시의 제목은 각 문자가 `0123456789abcdef` 중 하나로 이루어진 X 자리 문자열 전부를 중복 없이 사용했다. 희원은 T 개의 시 중 i 번째 시는 N_i 번째 날에 작업한 K_i 번째 시가 좋을 것 같다고 말했다. 여기서 희원이 말한 의미는 N_i 번째 날의 모든 시 16^{N_i} 개의 제목을 자연스러운 문자열 비교로 정렬했을 때 K_i 번째 시라는 의미였기 때문에, 종서가 일반적으로 프로그래밍에서 사용하는 문자열 비교와 다른 책을 가리키는 경우가 있었다. 평소 같았으면 직접 프로그램을 작성해서 시의 제목을 알아보았을 종서지만, 브런치북 작업을 빨리 진행하고 싶었기 때문에 당신에게 희원이 선택한 각 시의 제목을 알려줬으면 좋겠다는 부탁을 했다.

입력

다음과 같이 입력이 주어진다.

T
N_1 K_1
...
N_T K_T

- T은 희원이 고른 시의 수이다. ($1 \le T \le 1000$)
- N_i와 K_i는 희원이 고른 시의 정보를 나타낸다. i 번째로 선택한 시가 N_i 번째 날의 모든 시를 16^{N_i} 개를 자연스러운 문자열 비교로 정렬했을 때 K_i 번째 시라는 의미이다. ($1 \le N_i \le 15, 1 \le K_i \le 16^{N_i}$)
- 입력으로 주어지는 모든 수는 정수다.

출력

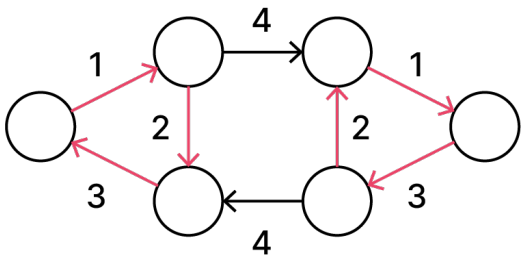
총 N 개의 줄을 출력한다. i 번째 줄에는 희원이 선택한 i 번째 시의 제목을 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
9 1 1 10 9809476352 11 178182069613 11 13935560805565 15 768614336404564651 15 910344248188228555 15 1003434262550842233 15 1003468668299564026 15 1152921504606846976	0 0be5e9ee5e 0c7e7add1c7 caca0c0ffee aaaaaaaaaaaaaaa ca5cadeda7aba5e decea5ed0ffbea7 ded1ca7ed9ad9e7 fffffffffffffff

문제 J. 사이클

시간 제한	2 초
메모리 제한	1024 MB



정점이 N개, 가중치가 있는 간선이 M개인 방향 그래프가 주어진다.

경로는 두 정점을 잇는 간선의 수열을 의미하고, 사이클은 시작점과 끝점이 같은 경로를 의미한다. 사이클 중에서 단순 사이클은 시작점과 끝점을 제외하고 중복되는 정점이 없는 사이클을 의미한다.

주어진 그래프에서 가중치의 합이 최소가 되도록 모든 정점을 포함하고 서로 겹치지 않는 단순 사이클의 집합을 구하는 프로그램을 작성하시오.

입력

다음과 같이 입력이 주어진다.

$$N\ M$$
$$u_1\ v_1\ w_1$$
$$\vdots$$
$$u_M\ v_M\ w_M$$

- N은 정점의 개수이고 M은 간선의 개수이다. $(2 \le N \le 200, 0 \le M \le N(N-1))$
- $u_i\ v_i\ w_i$는 정점 u_i에서 v_i로 가는 가중치 w_i의 간선이 존재한다는 뜻이다. $(1 \le u, v \le N, u \ne v, -10^9 \le w \le 10^9)$

주어진 그래프 내에서 루프는 존재하지 않으며, 서로 다른 두 정점 사이에 최대 한 개의 간선이 존재한다.

출력

모든 정점을 포함하고 서로 겹치지 않는 사이클의 집합이 존재하는 경우 첫 번째 줄에 `1`을 출력한다. 그렇지 않으면 `0`을 출력한다.

모든 정점을 포함하고 서로 겹치지 않는 단순 사이클의 집합이 존재하는 경우, 두 번째 줄에 이러한 집합들의 가중치의 합의 최솟값을 출력하고, 이후 N개의 줄에 걸쳐서 해당 집합의 단순 사이클에 속한 간선들을 출력한다. 간선들의 출력 순서는 상관없으며, 답이 여러 개인 경우 그 중 아무 것이나 출력하면 된다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
4 8 1 2 1 2 3 1 3 4 1 4 1 1 2 1 2 3 2 2 4 3 2 1 4 2	1 4 1 2 2 3 3 4 4 1
6 8 1 2 1 2 3 2 3 1 3 2 4 4 4 5 1 5 6 3 6 4 2 6 3 4	1 12 1 2 2 3 3 1 4 5 5 6 6 4

문제 K. 수요응답형 버스

시간 제한	1 초
메모리 제한	1024 MB

현대오토에버는 In-Car와 Out-Car 영역 전반의 소프트웨어와 인프라 관련 업무를 수행하는 회사이다. 현재 현대오토에버에서 수요응답형 버스(MOD)를 개발하고 있다. 수요응답형 버스는 승객이 호출하면 실시간으로 가장 빠른 경로가 생성되고 배차가 이뤄지는 수요응답형 버스로, 노선 체계가 갖춰지기 시작하는 도시개발 중간단계에서 주민들의 이용 편의를 향상시킬 수 있는 신개념 모빌리티 솔루션이다. 고정된 노선 없이 실시간 호출에 의해 배차되고 운행되므로 시민의 차량 대기 시간과 이동 시간이 단축돼 대중교통 편의성이 크게 향상된다.

당신은 출퇴근 시간처럼 호출이 몰렸을 때, 동시에 매칭해주는 시스템을 개발하고 있다. 배차 요청은 N 개가 들어왔고, 각 배차 요청은 탑승 인원과 최대 대기 가능 시간으로 이루어져 있다. 버스는 M 대가 있고, 각 버스에는 정원과 도착 예정 시간이 있다. 또한 버스는 도착하면 1분 이상 기다리지 않고 떠난다.

어떤 배차 요청에 버스를 배정할 수 있으려면, 버스의 정원이 탑승 인원보다 크거나 같아야 하며, 도착 예정 시간이 최대 대기 가능 시간보다 작거나 같아야 한다. 예를 들어 탑승 인원이 4명이고 최대 대기 가능 시간이 7분이면, 정원이 4명 이상이고, 도착 예정 시간이 7분 이내인 버스를 배차할 수 있다.

배차 요청과 버스를 대응시킬 때, 서로 1:1 대응시켜야 한다는 제약사항이 있다. 즉, 하나의 배차 요청을 두 대 이상의 버스에 배정할 수 없으며, 반대로 하나의 버스가 두 개 이상의 배차 요청에 배정되는 것은 정책상 허락하지 않기로 하였다. 또한, 같은 시간에 여러 개의 요청을 동시에 처리할 수 있다.

당신은 최대한 많은 배차 요청에 버스를 대응시켜주는 프로그램을 기능을 구현해야 한다.

입력

다음과 같이 입력이 주어진다.

$N\ M$
$a_1\ b_1$
$\cdots$
$a_N\ b_N$
$c_1\ d_1$
$\cdots$
$c_M\ d_M$

- N은 배차 요청의 수이고, M은 버스의 수이다. $(1 \le N, M \le 200\,000)$
- a_i와 b_i는 배차 요청의 정보를 나타낸다. i 번째 배차 요청의 탑승 인원이 a_i 명이고, 최대 대기 가능 시간이 b_i 분임을 나타낸다. $(1 \le a_i, b_i \le 10^9)$
- c_j와 d_j는 배차 요청의 정보를 나타낸다. j 번째 버스의 정원이 c_j 명이고, 도착 예정 시간이 d_j 분임을 나타낸다. $(1 \le c_j, d_j \le 10^9)$
- 입력으로 주어지는 모든 수는 정수다.

출력

첫 번째 줄에 버스를 배정할 수 있는 배차 요청 수의 최댓값을 출력하여라.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
2 2 4 3 7 4 5 2 8 3	2

첫 번째 배차 요청에 첫 번째 버스를 배정하고, 두 번째 배차 요청에 두 번째 버스를 배정하면, 모든 배차 요청을 처리할 수 있다.

표준 입력(stdin)	표준 출력(stdout)
3 2 4 3 1 3 7 4 5 2 8 3	2

하나의 버스에는 오직 하나의 배차 요청을 한 그룹만 태울 수 있다. 즉, 배차 요청 1과 2를 버스 1로 처리하거나 배차 요청 2와 3을 버스 2로 처리하는 행동은 불가능하다.

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

배차 요청 1을 한 그룹을 버스 1과 버스 2에 나눠서 태우는 행동은 불가능하다.

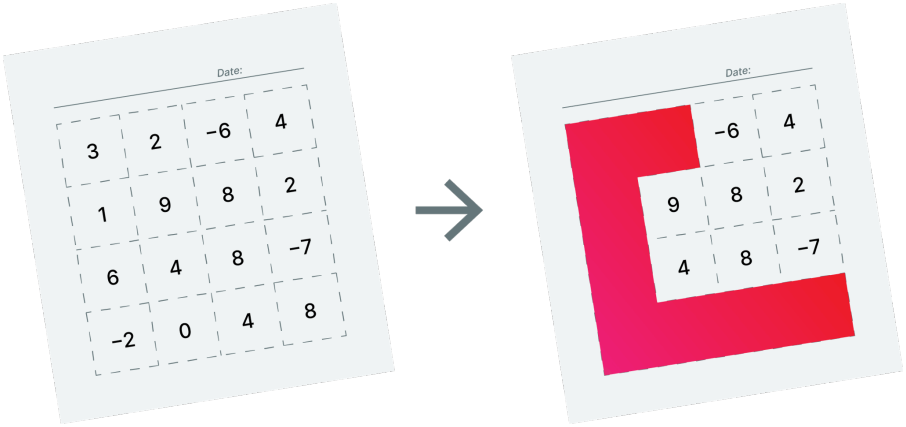
문제 L. 다꾸

시간 제한	3 초
메모리 제한	1024 MB

유행은 돌고 돈다고 하지 않았던가. 스마트폰의 등장으로 많은 사람들이 종이는 이제 멸종의 길을 걷게 될 것이라고 걱정했지만, 2021년의 20대들 사이에서는 '다이어리 꾸미기', 일명 '다꾸'가 유행이다.

우리의 청한이도 '다꾸'에 빠진 20대 중 한 명이다. 청한이는 다이어리를 꾸미기 위해 오늘도 지갑을 털어서 문구점에서 네모 테두리 모양 스티커를 많이 사 왔다.

청한이는 스티커를 잘라 최적의 위치에 붙이고 #다꾸 #다이어리꾸미기 #다꾸스타그램 #스꾸 #스티커 #스티커꾸미기 해시를 달아 SNS에 자랑할 생각에 신이 나 있다. 하지만 스티커를 어떻게 잘라 붙여야 잘 붙였다고 소문이 날까 고민이었던 나머지, 전공을 살려 다꾸러들의 미적 감각을 빅데이터로 분석하여 가장 높은 좋아요 수를 이끌어낼 만한 스티커 위치를 계산해 주는 인공지능을 만들기로 했다.



다이어리 속지는 높이 H, 너비 W의 격자 모양이다. 맨 왼쪽 위 칸을 $(1,1)$, 맨 오른쪽 아래 칸을 (H,W)로 나타낼 수 있다. 청한이는 빅데이터를 활용해 어떤 칸에 스티커가 붙으면 얼마나 '예쁜지'를 각 칸마다 계산해 두었다. 이 값은 -10^9 이상 10^9 이하의 정수이다.

네모 테두리 모양 스티커의 테두리 두께는 한 칸이다. 여러 가지 크기의 스티커가 준비되어 있지만 청한이는 어떤 직사각형 영역을 정해 그 직사각형 영역에 꼭 맞는 크기의 스티커를 고를 것이다.

스티커의 '예쁨'은 스티커가 다이어리에 붙었을 때 가려진 칸들의 '예쁨'의 합이다. 고른 스티커를 영역의 테두리에 맞춰 붙일 때 제일 예쁘게 붙을 수 있도록 적절히 잘라 한 조각만 붙일 것이다. 물론 스티커를 자르지 않고 테두리 모양 전체를 붙일 수도 있다. 스티커가 새로 붙어도 각 칸의 '예쁨'은 바뀌지 않는다.

또한 청한이는 올망졸망한 손글씨로 오늘 있었던 일을 쓰려고 한다. 이 경우 청한이가 정한 칸의 '예쁨'이 바뀐다.

청한이는 이미 스티커가 붙은 위치 위에 스티커를 겹쳐 붙일 수도 있고, 손글씨가 적혀 있는 자리에 스티커를 덮어 붙일 수도 있다. 청한이를 도와 최적의 스티커 위치를 정해 주자.

입력

다음과 같이 입력이 주어진다.

$H\ W\ T$
$A_{1,1}\ A_{1,2}\ \cdots\ A_{1,W}$
$A_{2,1}\ A_{2,2}\ \cdots\ A_{2,W}$
$\vdots\ \ \vdots\ \ \ddots\ \ \vdots$
$A_{H,1}\ A_{H,2}\ \cdots\ A_{H,W}$
op_1
$\vdots$
op_T

- H와 W는 다이어리 내지의 크기를 나타내는 정수이다. $(3 \le H, 3 \le W, 9 \le HW \le 10^6)$
- T는 청한이가 다이어리 꾸미기를 하는 횟수이다. $(1 \le T \le 100\,000)$
- A_{ij}는 (i, j)의 최초 '예쁨'이다. $(-10^9 \le A_{ij} \le 10^9)$
- op_i는 청한이가 다이어리를 어떻게 꾸밀지를 설명한다.
  - 스티커를 붙이려는 경우, 다음과 같이 주어진다. 이는 칸 (x_1, y_1)과 (x_2, y_2)를 대각 꼭지점으로 하는 네모 테두리 모양 지점에 스티커를 붙일 예정임을 의미한다.

    $1\ x_1\ y_1\ x_2\ y_2$

    * $1 \le x_1 < x_2 \le H$, $1 \le y_1 < y_2 \le W$
    * $x_1 + 2 \le x_2$, $y_1 + 2 \le y_2$
  - 손글씨를 쓰려는 경우, 다음과 같이 주어진다. 이는 칸 (x, y)의 '예쁨' 값을 p로 변경한다는 의미이다.

    $2\ x\ y\ p$

    * $1 \le x \le H$, $1 \le y \le W$, $-10^9 \le p \le 10^9$

출력

스티커를 붙일 때마다, 청한이가 스티커를 최대한으로 예쁘게 붙였을 때의 예쁨을 한 줄에 하나씩 출력한다.

입출력 예시

표준 입력(stdin)	표준 출력(stdout)
4 4 3 3 2 -6 4 1 9 8 2 6 4 8 -7 -2 0 4 8 1 1 1 4 4 2 2 1 -9 1 1 1 4 4	22 16
4 8 2 0 0 2 0 0 -1 -1 -1 0 1 1 1 0 -1 9 -1 0 1 1 1 0 -1 -1 -1 0 2 -9 2 0 1 1 1 1 1 1 4 5 1 1 6 3 8	6 -1

이 페이지는 공백입니다