제3회 흐즈로컵 풀이

Official Solutions

by

hjroh0315 a.k.a chromate00

문제		의도한 난이도	출제자
A	이상한 섞기 연산	Easy	hjroh0315
B	슈퍼 소수	Easy	hjroh0315
C	Y	Easy	hjroh0315
D	자료 구조의 왕	Medium	hjroh0315
E	세계 일주	Medium	hjroh0315
F	연결된 지배 집합	Medium	hjroh0315
G	어려운 정수 맞히기 게임	Hard	hjroh0315
H	수열과 어렵지 않은 쿼리	Hard	hjroh0315
I	선인장 접기	Hard	hjroh0315
J	혼합 정수 이차 계획법	Challenging	hjroh0315
K	선인장 접기 Plus	Challenging	hjroh0315

A. 이상한 섞기 연산

`ad_hoc, implementation`

출제진 의도 – **Easy**

- 제출 286번, 정답 117명 (정답률 43.7%)
- 처음 푼 참가자: **sharaelong**, 1분
- 출제자: `hjroh0315`

A. 이상한 섞기 연산

$n = 1$ 인 경우부터 순차적으로 차근차근 생각해 봅시다.

- ✓ $n = 1$: $n = l$ 이므로 교환은 발생하지 않습니다.
- ✓ $n = 2$: $n = l$ 이므로 교환은 발생하지 않습니다.
- ✓ $n = 3$: $l = 1$ 이므로 B_1 과 B_3 을 교환합니다. 이제 1 은 3 번 인덱스에 있습니다.
- ✓ $n > 3$: l 은 2 의 거듭제곱이므로 3 이 될 수 없습니다. 1 의 위치는 더 이상 바뀌지 않습니다.

따라서 $n \leq 2$ 인 경우 정답은 1 이고, 나머지 경우 정답은 3 입니다.

B. 슈퍼 소수

`sieve`

출제진 의도 – **Easy**

- 제출 214번, 정답 151명 (정답률 74.3%)
- 처음 푼 참가자: **mj1000j**, 3분
- 출제자: `hjroh0315`

B. 슈퍼 소수

- 에라토스테네스의 체를 구현하여 문제에 설명된 그대로의 답을 계산하면 됩니다.
- 제한 하에서 최댓값이 $318\,137$임을 예제로 추정할 수 있습니다.
- 따라서 시간 복잡도에 대해 매우 엄밀한 분석은 요구되지 않습니다.

C. Y

degree_sequence, combinatorics
출제진 의도 – **Easy**

- 제출 233번, 정답 127명 (정답률 56.7%)
- 처음 푼 참가자: **sharaelong**, 5분
- 출제자: hjroh0315

- 그래프에서 Y의 루트를 고정하고 나면 나머지 세 리프를 결정하는 가짓수로 Y의 개수가 결정된다는 사실을 어렵지 않게 관찰할 수 있습니다.
- 루트의 차수가 k일 때, 세 리프를 결정하는 가짓수는 다음과 같습니다.

$$\binom{k}{3} = \frac{k(k-1)(k-2)}{6}$$

- 따라서 차수열을 구하고 나면 문제의 정답은 다음과 같습니다.

$$\sum_{i=1}^{n} \binom{deg(i)}{3}$$

- 시간 복잡도는 $\mathcal{O}(n+m)$ 이 됩니다.

D. 자료 구조의 왕

`implementation`
출제진 의도 – **Medium**

- 제출 97번, 정답 75명 (정답률 77.3%)
- 처음 푼 참가자: **aeren**, 10분
- 출제자: `hjroh0315`

- 잔디 깎기를 한 번 실행할 때마다 최악의 경우 $\mathcal{O}(n)$ 또는 $\mathcal{O}(m)$의 시간이 걸립니다.
- 이를 감안하면 최악의 경우 총 $\mathcal{O}((n+m)Q)$의 시간이 걸릴 것 같습니다.
- 하지만 실제로는 문제에 적힌 것을 그대로 구현하면 AC를 받습니다.
- 왜 그럴까요?

- 각 칸에서 잔디를 깎을 때마다 $\mathcal{O}(1)$ 의 시간이 걸립니다.
- 어떤 칸에서 잔디를 깎는 사건은 최대 1 번 발생합니다.
- 초기에 잔디가 nm 칸 있었으므로, 이 사건이 기여한 시간 복잡도는 $\mathcal{O}(nm)$ 입니다.

- 각 칸 또는 잔디밭 끝에서 잔디를 깎지 않고 잔디 깎기를 끝낼 때마다 $\mathcal{O}(1)$ 의 시간이 걸립니다.
- 잔디 깎기가 끝나는 사건은 잔디 깎기를 시작할 때마다 정확히 1 번 발생합니다.
- 잔디 깎기는 Q 번 넘게 시행될 수 없으므로, 이 사건이 기여한 시간 복잡도는 $\mathcal{O}(Q)$ 입니다.

- 따라서 총 시간 복잡도는 $\mathcal{O}(nm + Q)$ 입니다.
- 이러한 시간 복잡도 분석 방법을 "분할상환분석"이라고 부르며, 동적 배열 등 다양한 알고리즘이나 자료 구조의 시간 복잡도 증명에도 이와 같은 분석 방법이 사용됩니다.

E. 세계 일주

`tsp, backtracking, geometry`

출제진 의도 – **Medium**

- 제출 103번, 정답 40명 (정답률 39.8%)
- 처음 푼 참가자: **sharaelong**, 15분
- 출제자: `hjroh0315`

- 문제의 첫 번째 조건만 보면 유클리드 거리를 사용하는 외판원 문제 (TSP)와 동일합니다.
- 문제의 두 번째 조건이 최적해에 어떤 영향을 줄까요?

- 사실 두 번째 조건이 있으나 없으나 최적해는 같습니다. 최적해가 항상 두 번째 조건을 만족하기 때문입니다.
- 이것은 귀류법으로 증명 가능합니다.

- 최적해가 어떤 해밀턴 회로 $A \to \cdots \to B \to C \to \cdots \to D \to A$로 구성되며 $B \to C$와 $D \to A$ 사이 교점이 존재한다고 가정합시다.
- 교점을 Q로 두면 삼각부등식에 의해 $\overline{BD} < \overline{BQ} + \overline{DQ}$과 $\overline{AC} < \overline{AQ} + \overline{CQ}$가 성립합니다.
- 따라서 $\overline{BD} + \overline{AC} < \overline{BC} + \overline{AD}$가 성립하여 $A \to \cdots \to B \to D \to \cdots \to C \to A$가 더 작은 비용을 가지게 됩니다.
- 이는 가정에 모순입니다.

- 삼각부등식이 성립하지 않기 위해서는 $\triangle BDQ$ 또는 $\triangle ACQ$가 한 직선 위에 존재하여 부등호가 등호로 바뀌어야 합니다.
- 그러나 정의상으로 Q는 $\overline{BC}$ 위와 $\overline{AD}$ 위를 동시에 차지합니다.
- 따라서 $\triangle BDQ$가 한 직선 위에 있다면 B, C, D 세 점이 한 직선 위에 있게 됩니다.
- 마찬가지로 $\triangle ACQ$가 한 직선 위에 있다면 A, C, D 세 점이 한 직선 위에 있게 됩니다.
- 이는 문제 조건에 모순입니다.

- 따라서 TSP의 최적해는 두 번째 조건을 이미 만족합니다.
- 두 번째 조건을 무시하고 TSP의 최적해를 구해주면 문제가 해결됩니다.

F. 연결된 지배 집합

`ad_hoc, constructive, implementation`

출제진 의도 – **Medium**

- 제출 77번, 정답 32명 (정답률 41.6%)
- 처음 푼 참가자: **ibm2006**, 19분
- 출제자: `hjroh0315`

- 존재성에 대한 정답은 항상 YES입니다.
- 조건을 만족시키는 구성 방법은 여러 가지 있지만, 그 중 한 가지만 서술합니다.
- 본 해설에서는 일반성을 잃지 않고 $n \geq m \geq 2$를 가정합니다.

- 격자 그래프의 첫 두 행에 해당하는 영역은 A라고 부릅니다.
- A를 제외한 나머지 영역은 B라고 부릅니다.
- A와 B에서 각각 부분집합 A', B'를 찾아서 그 합집합이 연결된 지배 집합이 되도록 합니다.

- A'는 격자 그래프의 두 번째 행에 해당하는 정점만 포함합니다.
- B'는 B의 정점 중 짝수 번째 열에 해당하는 정점만 포함합니다.
- 이때 $|A'| = \frac{|A|}{2}$이고 $|B'| \le \frac{|B|}{2}$임을 어렵지 않게 보일 수 있습니다.
- 따라서 $|A' \cup B'| \le \frac{nm}{2}$가 성립합니다.
- $A' \cup B'$가 연결된 지배 집합이라는 것도 어렵지 않게 보일 수 있습니다.
- 따라서 $A' \cup B'$는 문제의 조건을 만족하는 집합이 됩니다.

F. 연결된 지배 집합

- 논문 “Connected domination in grid graphs”에서 아이디어를 얻어 만들어진 문제입니다.
- 링크된 논문에서는 격자 그래프의 연결된 지배 집합의 최소 크기를 찾고 최소임을 증명합니다.
- n과 m에 따라 정의된 격자 그래프에 대해 연결된 지배 집합의 최소 크기는 $\frac{nm}{3}$ 내외입니다.
- 본 문제를 해결하기 위해 논문에 대한 지식은 요구되지 않았습니다.
- 물론, 누군가 논문을 읽고 내용에 따라 코드를 3시간 안에 구현할 수 있을 것이라고 예상하지 않기 때문에, 논문이 요구되는 문제도 내지 않았습니다.

G. 어려운 정수 맞히기 게임

`binary_search, math`

출제진 의도 – **Hard**

- 제출 103번, 정답 31명 (정답률 30.1%)
- 처음 푼 참가자: **ncy09**, 22분
- 출제자: `hjroh0315`

G. 어려운 정수 맞히기 게임

- 함수 $\sqrt{x-a}-b$가 단조증가하므로 이분 탐색을 할 수 있을 것 같습니다.
- 하지만 b를 조작하여 단순하게 이분 탐색을 하면 $\mathcal{O}\left(\sqrt{x}\right)$ 만큼의 차이가 발생할 수 있습니다.
- a를 조작하여 이분 탐색하는 것은 $x-a<0$일 가능성 때문에 할 수 없습니다.
- a를 조작하여 선형 탐색하는 것은 당연히 할 수 없습니다.

G. 어려운 정수 맞히기 게임

- 풀이를 설명하기 전 식을 간단히 조작합시다.
- 부등식을 다음과 같이 수정할 수 있습니다.
- 제곱근 기호 안의 $x - a$가 항상 음이 아니므로, 다음과 같은 조작이 가능합니다.

$$\sqrt{x-a} - b < 0 \iff \sqrt{x-a} < b \iff x - a < b^2 \iff x - a - b^2 < 0$$

- 이를 활용하면 실수 오차를 배제하고 구현하는 것이 가능합니다.
- 실제 채점기도 위의 방법을 활용해 구현되었습니다.

G. 어려운 정수 맞히기 게임

- 이분 탐색 전후에 x가 있을 수 있는 구간을 관리합시다.
- 초기에 구간은 $[0, 10^{18}]$ 입니다.
- 현재 구간의 크기가 S라고 합시다.
- 이분 탐색으로 x가 있을 수 있는 크기 $2\left\lfloor\sqrt{S}\right\rfloor$ 이하의 구간을 찾을 수 있습니다.
- 이 구간의 왼쪽 끝을 a로 두고 다시 이분 탐색합니다.
- 이 과정을 반복하면 $\mathcal{O}(\log\log S)$ 번의 이분 탐색으로 S가 $\mathcal{O}(1)$로 줄어듭니다.
- S가 $\mathcal{O}(1)$로 줄어들고 나면 선형 탐색이 가능합니다.
- 이때 질문을 얼마나 사용하게 될까요?

- 각 이분 탐색에서 $\left\lfloor \log_2(\sqrt{S}) \right\rfloor = \lfloor \log_2(S)/2 \rfloor$ 개의 질문을 사용합니다.
- 이분 탐색 후에는 $\log_2(S)$ 가 $1 + \lfloor \log_2(S)/2 \rfloor$ 이하가 됩니다.
- 이분 탐색 중에 x 의 값이 특정 가능하게 되는 경우를 배제하고 최악의 경우를 가정합시다.

G. 어려운 정수 맞히기 게임

- 초기에 $0 \le x \le 10^{18}$ 이므로 $\log_2(S) = 60$ 으로 둡니다.
- 첫 단계에는 $\lfloor \log_2(S)/2 \rfloor = 30$ 번의 질문을 사용하며, $\log_2(S)$ 가 31 로 바뀝니다.
- 이를 반복하면 $S = 4$ 일 때까지 질문의 수가 $30 + 15 + 8 + 4 + 2 + 2 = 61$ 이 됩니다.
- $S = 4$ 가 되면 $S = 2\left\lfloor \sqrt{S} \right\rfloor$ 이므로 기본적으로 구간이 작아지지는 않습니다.
- 이 지점부터 선형 탐색하면 항상 70 번 이하의 질문을 사용해 문제를 해결할 수 있습니다.

- 이분 탐색에서 선형 탐색으로 전환을 일찍 하는 등 풀이에 미세한 차이가 있을 수 있습니다.
- 여러 가지 풀이를 허용하기 위해 질문 개수의 제한을 최대 75번으로 조금 크게 두었습니다.

H. 수열과 어렵지 않은 쿼리

data_structures, segment_tree
출제진 의도 – **Hard**

- 제출 39번, 정답 29명 (정답률 74.4%)
- 처음 푼 참가자: **ncy09**, 8분
- 출제자: hjroh0315

H. 수열과 어렵지 않은 쿼리

- $\mathcal{O}\left((n+q)\log n\right)$ 시간 복잡도의 매우 다양한 풀이가 있습니다.
- GNU PBDS를 사용해서 이 문제를 풀 수 있다는 사실은 이해하기 어렵지 않습니다.
- 문제는 GNU PBDS를 사용하는 풀이가 생각보다 복잡한 구현을 요구한다는 것입니다.
- 흔히 "금광 세그먼트 트리"라고 부르는 방법을 응용해도 되지만, 여전히 어렵습니다.
- 본 에디토리얼에서는 익숙한 구간 합 자료구조를 사용해 문제를 해결하는 방법을 제시합니다.

- 중요한 연속 일치 구간의 개수 자체를 세지 않아도 됩니다.
- 중요한 연속 일치 구간이 바뀐 횟수의 시점에서 문제를 접근합시다.
- 중요한 연속 일치 구간의 개수는 바뀐 횟수보다 정확히 1 더 많게 됩니다.
- 중요한 연속 일치 구간이 바뀌는 것은 인접한 두 위치 사이에서만 발생합니다.
- 따라서 인접한 두 위치가 다른지를 구간 합 자료 구조로 관리하면 됩니다.

H. 수열과 어렵지 않은 쿼리

- 구간 합 세그먼트 트리 또는 펜윅 트리를 관리합니다.
- 인접한 두 값이 같다면 0을, 다르다면 1을 트리의 원소로 저장해 줍니다.
- 이제 1번 쿼리는 이 자료 구조에서 원소를 최대 2개 변경해 주는 연산이 됩니다.
- 2번 쿼리는 구간 합 연산이 됩니다.
- 이제 세그먼트 트리나 펜윅 트리를 그대로 사용할 수 있어 어렵지 않게 구현이 가능합니다.

I. 선인장 접기

bcc, dp, knapsack, bitset, math, cactus
출제진 의도 – **Hard**

- 제출 38번, 정답 11명 (정답률 31.6%)
- 처음 푼 참가자: **yijw0930**, 72분
- 출제자: hjroh0315

- 일반적인 그래프에 대해서는 이 문제를 다항 시간에 풀 수 없을 가능성이 높습니다.
- 선인장 그래프에 한하여, 다음 조건은 “그래프를 접어서 1차원으로 만들 수 있다”의 필요충분조건입니다.

“각 사이클에 대해서 간선의 길이의 합이 S 라면,
간선의 길이의 합이 $S/2$ 인 간선의 부분집합을 찾을 수 있다.”

- 두 조건이 필요충분조건임은 K번의 에디토리얼에서 증명합니다.
- I번에서는 이것이 사실이라고 믿고 푸는 것으로도 충분합니다.

- 이제 각 사이클에 대해 Subset Sum 문제를 해결하면 됩니다.
- 문제는 시간 복잡도입니다.
- 기본적인 다이나믹 프로그래밍 풀이는 시간 복잡도가 $\mathcal{O}\left(m^2 l\right)$ 로 너무 느립니다.

- 길이가 l 인 간선이 k 개 있다고 가정하면, k 를 적당히 $\mathcal{O}(\log k)$ 개로 쪼갤 수 있습니다.
- 개수가 a 로 가장 작은 덩어리를 고릅니다.
- 이를 $\left\lfloor \frac{a}{2} \right\rfloor$ 개와 $\left\lceil \frac{a}{2} \right\rceil$ 개로 쪼갭니다.
- 가장 작은 덩어리가 원소 1 개짜리 덩어리가 될 때까지 반복합니다.
- 이렇게 얻은 $\mathcal{O}(\log k)$ 개의 덩어리로 Subset Sum을 수행하면 결과는 기존과 같습니다.

- 이제 원소의 개수가 $\mathcal{O}(l \log m)$ 개가 되었으므로 시간 복잡도가 $\mathcal{O}(ml^2 \log m)$ 이 됩니다.
- 이 풀이를 비트 집합으로 최적화하면 시간 복잡도가 $\mathcal{O}\left(\frac{ml^2 \log m}{w}\right)$ 이 됩니다.
- 이는 문제를 해결하기에 충분히 빠릅니다.
- 전체 시간 복잡도는 $\mathcal{O}\left(n + \frac{ml^2 \log m}{w}\right)$ 입니다.

- 이 문제에서 비트 집합을 사용할 때 동적 비트 집합이 필요할 수 있습니다.
- C++의 경우는 GCC의 `std::tr2::dynamic_bitset<>`을 쓸 수 있습니다.
- 또는 $len = 2^k$ 인 `bitset<len>`을 사용할 수도 있습니다.
- Python 등 다른 언어의 경우 임의 정밀도 정수를 동적 비트 집합으로 사용할 수도 있습니다.

J. 혼합 정수 이차 계획법

`flow`, `mcmf`, `math`
출제진 의도 – **Challenging**

- 제출 26번, 정답 5명 (정답률 19.2%)
- 처음 푼 참가자: **sharaelong**, 65분
- 출제자: `hjroh0315`

- ✓ 이 문제의 제약은 볼록 함수의 특성과 기존 MCMF 문제의 특성을 정교하게 이해하고 응용해야 해결 가능하도록 설계되었습니다.

J. 혼합 정수 이차 계획법

- $f(x) = ax^2 + bx$는 $a \geq 0$에 의해 아래로 볼록하며, 이 사실은 매우 중요합니다.
- 그 이유 중 하나로 $a < 0$일 경우 문제가 NP-Hard가 된다는 점이 있습니다.
- 반대로, 볼록성을 잘 응용하면 MCMF의 구현체를 블랙박스로 사용해 풀 수도 있게 됩니다.

- 함수 $f(x) = ax^2 + bx$ 를 변형해 각 꼭짓점이 정수점인 함수 $g(x)$ 를 만듭시다.
- $x \in \mathbb{Z}$ 인 경우 $f(x) = g(x)$ 입니다.
- 그 외의 경우 $g(x) = f(\lfloor x \rfloor) + (f(\lceil x \rceil) - f(\lfloor x \rfloor))(x - \lfloor x \rfloor)$ 로 정의됩니다.
- 이는 piecewise linear 하며, 기울기는 $f(x+1) - f(x) = 2ax + a + b$ 로 증가합니다.
- 따라서 이 새로운 함수 $g(x)$ 는 $f(x)$ 와 마찬가지로 아래로 볼록합니다.

- ✓ 문제를 완전히 해결하기 위해서는 한 가지 관찰이 더 필요합니다.

- 어떠한 함수의 볼록성을 정의하는 방법은 여러 가지가 있습니다.
- 그 중 하나는 그 함수 위 또는 아래에 해당하는 집합이 볼록 집합이라는 특성에 따른 정의입니다.
- 따라서 볼록 집합의 특성을 볼록 함수에 적용할 수 있는 경우도 많습니다.

- 두 원소의 합이 잘 정의되는 모든 볼록 집합 A와 B가 있다고 합시다.
- 이때 두 집합의 민코프스키 합 $A + B = \{a + b \mid a \in A,\ b \in B\}$는 잘 정의됩니다.
- 이렇게 얻은 민코프스키 합의 결과 또한 볼록 집합이 됩니다.
- 반대로 다른 두 볼록 집합의 민코프스키 합으로 분해할 수 있는 볼록 집합도 있습니다.
- 다시 말해, $P = A + B$가 되는 두 집합 A와 B를 찾을 수 있는 볼록 집합 P가 존재합니다.

- 이 관찰을 새로운 함수 $g(x)$ 에 적용해 봅시다.
- 함수 $g(x)$ 의 볼록성은 $g(x)$ 위의 영역이 볼록 집합임으로 정의됩니다.
- 변형된 문제에서 관심이 있는 집합은 $C = (\{(x,y) \mid y \geq g(x)\} \cap ([0,c] \times \mathbb{R}))$ 입니다.
- 이는 볼록 집합과 볼록 집합의 교집합이므로 또한 볼록 집합입니다.

- 이 볼록 집합 C 를 단순한 볼록 집합 여러 개의 민코프스키 합으로 표현하는 것이 가능할까요?

- 실제로 이것은 가능합니다.
- $[0, c)$ 의 모든 정수 x 에 대해 $(0, 0)$ 과 $(1, f(x+1) - f(x))$ 를 연결하는 선분 L_x 를 정의합니다.
- S_x 를 L_x 자체를 포함해 L_x 보다 위쪽에 있는 영역으로 정의합니다.
- 이렇게 하면 $S_0 + S_1 + S_2 + \cdots + S_{c-1}$ 은 집합 C 와 같아집니다.

- 이제 MCMF 문제로 돌아옵시다.
- 어떤 간선의 비용이 $f(x) = ax^2 + bx$ 이고 용량은 c 단위입니다.
- 유량이 정수이므로 간선의 비용은 $g(x)$ 로도 표현 가능합니다.
- $a = 0$ 인 경우 간선을 그대로 두고, $a \neq 0$ 인 경우 간선을 중복 간선 c 개로 쪼갭니다.
- 쪼개진 간선 $E_0, E_1, E_2, \cdots, E_{c-1}$ 에 대해서, E_x 의 비용은 $f(x+1) - f(x)$ 이고 용량은 모두 1 입니다.
- 이 c 개의 간선을 따라서 x 단위의 자원을 보냈을 때 최소 비용은 $g(x)$ 가 됩니다.
- 이에 대한 증명은 이전의 관찰을 참조할 수도 있고, 탐욕적 접근을 직접 증명해도 됩니다.

- MCMF는 flow integrality theorem이 성립하므로 모든 변수가 정수인 최적해가 존재합니다.
- 그러한 최적해는 이 문제의 조건을 만족합니다.
- 따라서 설명한 대로 변형된 그래프에서 MCMF를 시행하면 문제가 해결됩니다.

- 다만 MCMF의 구현체가 변형된 그래프에서도 잘 작동하는지는 감안해야 합니다.
- 이 문제를 해결하기 위해서는 MCMF의 구현체가 중복 간선과 음수 사이클을 허용해야 함에 유의해야 합니다.
- 두 조건을 만족하면서 본 문제를 해결할 수 있는 것으로 확인된 알고리즘에는 다음이 있습니다.
- Cost Scaling은 $\mathcal{O}\left(V^3 \log VC\right)$의 복잡도를 가지며, 이때 C는 $\mathcal{O}(ac+b)$입니다.
- Capacity Scaling은 $\mathcal{O}\left(E^2 \log E \log U\right)$의 복잡도를 가지며, 이때 E는 최대 2400입니다.
- Minimum Cost b-flow (yosupo)를 200ms 이내에 통과하는 구현체라면 본 문제도 통과할 가능성이 높습니다.

- 여담으로, 아직 이 문제가 P에 속함을 증명한 것은 아닙니다.
- 이 문제가 P에 속하기 위해서는 입력 크기에 다항 시간인 알고리즘이 있어야 합니다.
- 그런데 c에 대한 입력 크기는 $\mathcal{O}(\log c)$ 입니다.
- 이 문제의 풀이는 c에는 다항 시간이지만 $\log c$에는 지수 시간입니다.
- 따라서 이 풀이는 이 문제가 P에 속하는지의 여부를 증명하지 못합니다.

K. 선인장 접기 Plus

bcc, dp, dfs, cactus

출제진 의도 – **Challenging**

- 제출 20번, 정답 0명 (정답률 0.0%)
- 처음 푼 참가자: -, -분
- 출제자: hjroh0315

- I번의 해설에서 “그래프를 접어서 1차원으로 만들 수 있다”의 필요충분조건으로 다음을 제시했습니다.

“각 사이클에 대해서 간선의 길이의 합이 S 라면,
간선의 길이의 합이 $S/2$ 인 간선의 부분집합을 찾을 수 있다.”

- 왜 이것이 필요충분조건인지 증명합시다.

- 우선 충분조건임을 증명합니다.
- 합이 $S/2$ 인 부분집합에 포함되는 간선에는 1 을, 나머지 간선에는 -1 을 배정합니다.
- 그 뒤 임의의 정점에서 DFS를 시작합니다.
- 현재 정점이 x_v 에 있으며 간선 (u, v, l) 을 지나고 있다고 합시다.
- 1 이 배정된 간선을 지나면 $x_u = x_v + l$, -1 이 배정된 간선을 지나면 $x_u = x_v - l$ 로 둡니다.
- 이렇게 하면 back edge를 포함해 모든 간선에서 $|x_u - x_v| = l$ 이 성립합니다.
- 이는 DFS가 선인장에서 탐색을 할 때 모든 사이클을 한 방향으로만 돌기 때문입니다.
- 이러한 과정을 역으로 수행하면 선술한 조건이 필요조건이라는 것도 증명이 가능합니다.

- 이제 I번의 풀이를 적당히 최적화한 뒤 역추적을 하면 될 것 같습니다.
- I번의 풀이에서 역추적을 구현할 수는 있지만 구현이 매우 귀찮아집니다.
- 심지어 $\mathcal{O}\left(\frac{ml^2 \log m}{w}\right)$ 라는 시간 복잡도에 $l = 500$ 라는 빠듯한 제한 때문에 시간 제한 내에 통과하기도 어려워졌습니다.

- 논문 “Linear Time Algorithms for Knapsack Problems with Bounded Weights”에서는 $\mathcal{O}(nW)$의 시간에 Subset Sum을 해결하는 법을 제시합니다.
- 또한, 역추적도 수월하게 할 수 있습니다!
- 역추적 방법은 논문에 제시되어 있는 DP를 그대로 응용 가능하기 때문에 생략합니다.
- Subset Sum의 시간 복잡도는 논문의 알고리즘 그대로 $\mathcal{O}(ml)$이 됩니다.
- 전체 시간 복잡도는 $\mathcal{O}(n+ml)$입니다.

- 역추적을 했다면 이제 증명의 내용 그대로 DFS를 수행해 각 정점에 x_i 의 값을 배정하면 됩니다.
- I번에서 BCC를 모두 나열하기 위해 템플릿을 사용했다면 그래프의 연결 컴포넌트의 개수에 상관없이 풀이가 가능했던 것에 비해, 본 문제에서는 연결 컴포넌트가 여러 개 있을 수 있음에 유의해야 합니다.
- DFS를 시작하는 지점 v에 대해 x_v 의 값을 0으로 두었다면, 크기 a인 같은 연결 컴포넌트 안에서 $-al \le x \le al$가 됩니다.
- $al \le 10^5 \times 500 = 5 \times 10^7$ 이므로 모든 x_i 는 $[-10^9, 10^9]$ 구간 안에 있도록 배정이 가능합니다.
- 단, 메모리 사용량에 주의하여 구현을 해야 합니다. 정해는 1024MB의 메모리 제한 중 약 420MB 정도를 사용합니다.