2025 ICPC Sinchon SUAPC Winter Solution

신촌지역 대학교 프로그래밍 동아리 연합

2025년 2월 22일 토요일

	문제	의도한 난이도	출제자
A	노트북 세 대를 가지고 왔다	Easy	김형진plast
B	skeep 문자열	Medium	박주형we12223
C	징검다리 게임	Medium	유상혁golazcc83
D	도로 공사	Medium	허준영jyheo98
E	Andrew the Diver	Challenging	이현빈my3
F	초코바 만들기	Easy	김형진plast
G	장비 강화하기	Challenging	유상혁golazcc83
H	완전 그래프와 쿼리	Medium	박**isekaijoucho
I	체리 컴퍼니	Hard	박진한jinhan814
J	blobnom.xyz	Easy	장래오leo020630
K	학식 뭐 먹지	Medium	김영인young_out
L	Anti-Fan Death	Easy	정치훈wjdclgns12
M	신촌 길찾기 서비스	Medium	유상혁golazcc83

A. 노트북 세 대를 가지고 왔다.

input, output

출제진 의도 - **Easy**

- 제출 23번, 정답 20팀 (정답률 : 86.957%)
- 처음 푼 팀 : **석우**[연세대학교], 1분
- 출제자 : **김형진**[plast]

A. 노트북 세 대를 가지고 왔다.

- 두 수를 입력받고 둘 중 더 작은 값을 출력합니다.

B. skeep 문자열

string, stack

출제진 의도 - **Medium**

- 제출 92번, 정답 12팀 (정답률 : 13.043%)
- 처음 푼 팀 : **calientacabezas**[서강대학교, 연세대학교], 9분
- 출제자 : **박주형**[we12223]

B. skeep 문자열

- s를 제외한 소문자 알파벳을 ?로 두고 한번의 작업으로 skeep 문자열을 ?로 바꾸어 넣어봅시다.
- 한번의 작업을 할때 skeep 문자열을 만들 수 있는 경우는 ...skeep... ...s????... 등 s기준 오른쪽으로 skeep 문자열이 가능합니다.
- 이때, ?는 s를 제외한 소문자 알파벳이므로 skeep 문자열은 서로 겹치지 않고 독립적입니다.
- 따라서 s를 제외한 소문자 알파벳을 ?로 두고 작업을 진행하면 작업을 하는 위치순서에 상관없이 최대횟수만큼 했을 때 생기는 문자열은 유일합니다.
- 그러면 스택을 이용하여 작업 후 문자간의 순서를 유지하면서 개수를 셀 수 있습니다.

B. skeep 문자열

- 문자열을 순회하면서 스택에 각 문자를 넣어줍니다.
- 이때 가장 마지막에 넣은 문자부터 5개의 문자를 확인했을 때, skeep 문자열이 가능하면 5개의 문자를 pop 후 ?를 push해서 개수를 세는 과정을 반복합니다.
- 총 시간복잡도는 $\mathcal{O}(N)$ 입니다.

C. 징검다리 게임

simulation, case_work

출제진 의도 - **Medium**

- 제출 44번, 정답 15팀 (정답률 : 36.364%)
- 처음 푼 팀 : **wowSpirit**[홍익대학교], 53분
- 출제자 : **유상혁**[golazcc83]

C. 징검다리 게임

- 점프(J) 명령어가 없다면 마지막 칸으로 이동할 수 없으므로 NO를 출력하면 됩니다.
- 명령어를 하나씩 처리할 경우, 각 칸을 이동할 때마다 곰의 체력만큼 명령어를 수행하여 시간 초과를 받을 수 있습니다.
- 징검다리의 i번째 칸 위에서 수행하는 공격(A), 지뢰 제거(D)로 구성된 부분 명령어를 X_i라고 가정하겠습니다. 이 때 각 X_i마다 처리할 수 있는 곰의 체력, 지뢰 제거 가능 여부를 전처리하여 수행 시간을 줄일 수 있습니다
- 명령어 안의 J의 개수를 C라고 하면
  - X_1은 입력으로 주어진 문자열의 첫 번째 명령어부터 첫 번째 J 사이의 부분 명령어입니다.
  - $2 \le i \le C$일 때 X_i는 $i-1$번째 J와 i번째 J 사이의 부분 명령어입니다.
  - X_{C+1}는 마지막 J와 첫 번째 J 사이의 환형을 이루는 부분 명령어로 주의하면서 구현해야 합니다.
  - $i \ge C+2$일 때 X_i는 $X_{(i-2 \mod C)+2}$와 동일합니다.

C. 징검다리 게임

- 각 X마다 처리할 수 있는 정보는 아래와 같습니다.
  - 첫 번째 명령어가 D라면 다음 칸의 지뢰를 제거할 수 있습니다. 만약 첫 번째 명령어가 A라면 다음 칸에 지뢰가 없어야 합니다.
  - X 안에 있는 A의 개수만큼 다음 칸의 곰의 체력을 깎을 수 있습니다.
- 각 칸을 지날 때마다 명령어를 하나씩 수행하는 대신 위에서 전처리한 정보를 바탕으로 $\mathcal{O}(1)$에 다음 칸으로 이동할 수 있는지 계산합니다.
- 총 시간복잡도는 $\mathcal{O}(N+K)$입니다.

D. 도로 공사

dijkstra

출제진 의도 - **Medium**

- 제출 15번, 정답 6팀 (정답률 : 40.000%)
- 처음 푼 팀 : **대충 26년 전역 어쩌구**연세대학교, 120분
- 출제자 : **허준영**jyheo98

D. 도로 공사

- 어떤 i번 정점에 대하여, 자기보다 tuple (D_i, i)가 작은 정점들 j 중 적어도 하나는 $D_j + dis = D_i$를 만족해야 합니다.
- tuple로 비교하는 이유는, D_i가 같은 정점을 2번 비교하지 않기 위해서입니다.
- 이는 여사건으로 $D_j + dis \geq D_i$의 가짓수에서 $D_j + dis > D_i$인 경우를 빼서 구해줄 수 있습니다.
- 만약 $D_j + \max_{dis} < D_i$인 정점이 하나라도 있으면 경우의 수는 즉시 0가지가 됩니다.

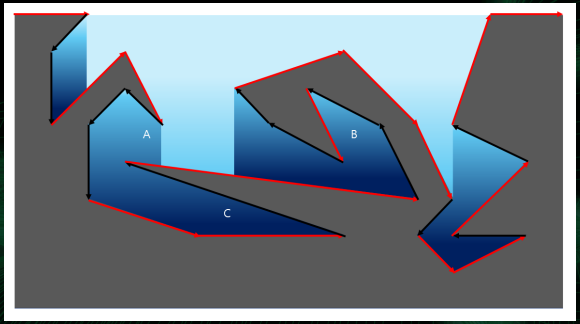
E. Andrew the Diver

geometry, lazyprop, tree_set

출제진 의도 - **Challenging**

- 제출 14번, 정답 0팀 (정답률 : 0.000%)
- 처음 푼 팀 : **?**$^{?}$, $?$분
- 출제자 : **이현빈**^{my3}

E. Andrew the Diver

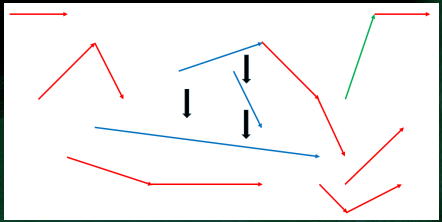


- 입력된 그림을 벡터로 나타내었으며, 이들 중 x방향 변위가 양수인 벡터는 빨간색으로 나타내었습니다.
- 수면 위 방향으로부터 보이는 모든 벡터는 빨간색 벡터임을 쉽게 알 수 있습니다.

E. Andrew the Diver

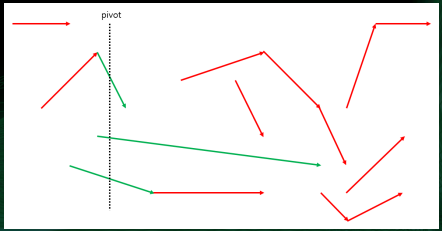
- 동굴의 어두운 영역에는 반드시 하나 이상의 꼭짓점이 포함됩니다.
- 어두운 영역에 포함되지 않는 꼭짓점들을 입력 순서대로 나열했을 때, 나열된 두 점이 연속한 점이 아니라면 두 점 사이에 하나 또는 두 개의 어두운 영역이 존재합니다.
- 따라서 어두운 영역에 포함되지 않는, 동굴 밖에서 관찰할 수 있는 꼭짓점을 구해야 합니다.
- 이러한 꼭짓점들을 찾기 위해, 동굴의 벽을 이루는 벡터들 중 수면 위에서 관찰 가능한 벡터들을 구할 것입니다.
- 각 벡터들을 그래프의 정점으로 보고, 위에서 아래를 보는 방향으로 위상 정렬하겠습니다.

E. Andrew the Diver



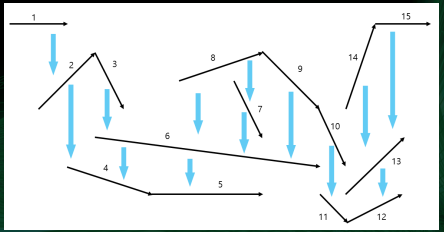
- 위 그림에서 파란색 벡터들간에는 어떤 벡터가 위에 있는지 쉽게 판단할 수 있지만, 파란색 벡터들과 초록색 벡터는 어떤 벡터가 위에 있는지 판단하기 어렵습니다.
- 두 벡터가 공유하는 x좌표가 존재할 때에만 어떤 벡터가 더 위에 있는지 판단하는 것이 가능합니다.

E. Andrew the Diver



- Shamos-Hoey 알고리즘처럼 특정 x값에 대응하는 y좌표를 기준으로 set에 벡터를 저장할 수 있습니다.
- 점들을 x좌표가 증가하는 순서로, 같은 x좌표를 가지면 시작점이 먼저 오도록 정렬합니다.
- 점들을 순서대로 검사하면서 시작점인 경우 set에 벡터를 넣고, 끝점인 경우 벡터를 제거하면서 위, 아래 벡터를 찾아 그래프에 간선을 추가합니다.

E. Andrew the Diver

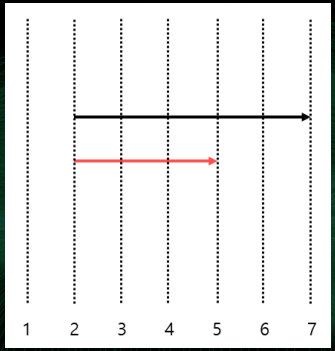


- 위상 정렬된 그래프를 생성하면 위 그림과 유사한 모습을 가지게 됩니다.
- 위 그림과는 달리 실제로는 두 벡터가 x좌표를 한 점만 공유하더라도 그래프의 간선으로 나타내어야 합니다.

E. Andrew the Diver

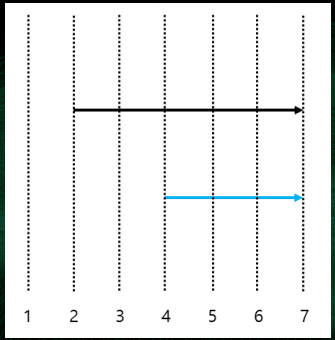
- 위상정렬된 그래프를 위에서부터 탐색하면서 노드(벡터)가 포함하는 x좌표를 확대해나가겠습니다.
- 이를 위해 좌표압축을 한 뒤 lazy propagation segment tree를 사용합니다.

E. Andrew the Diver



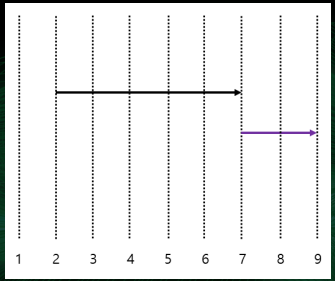
- 기존에 검은색 벡터와 같이 $[2, 7]$ 구간을 포함하고 있다고 가정해보겠습니다.
- 빨간색 벡터와 같이 $[2, 5]$ 구간을 포함하는 벡터가 주어지면 빨간색 벡터의 시작점은 동굴 입구에서 관측됩니다.

E. Andrew the Diver



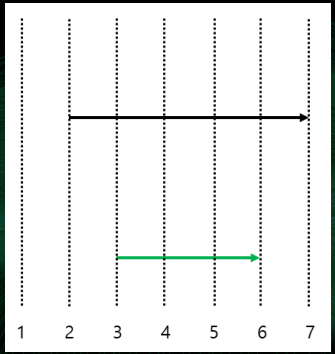
- 파란색 벡터와 같이 $[4, 7]$ 구간을 포함하는 벡터가 주어지면 파란색 벡터의 끝점은 동굴 입구에서 관측됩니다.

E. Andrew the Diver



- 보라색 벡터와 같이 $[7, 9]$ 구간을 포함하는 벡터가 주어지면 보라색 벡터의 시작점과 끝점은 모두 동굴 입구에서 관측됩니다.

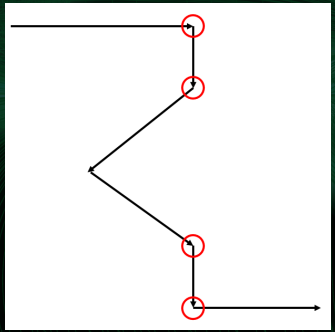
E. Andrew the Diver



- 초록색 벡터와 같이 $[3, 6]$ 구간을 포함하는 벡터가 주어지면 초록색 벡터의 시작점과 끝점은 모두 동굴 입구에서 관측되지 않습니다.

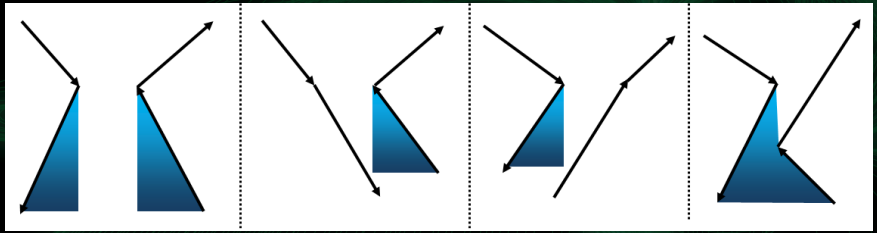
E. Andrew the Diver

- 이처럼 위상정렬된 순서대로 벡터가 포함하는 영역을 확장해나갈 수 있으며, 동시에 동굴 입구에서 보이는 꼭짓점을 찾을 수 있습니다.
- 동굴 입구에서 보이는 꼭짓점과 이웃하며 같은 x좌표를 가진 꼭짓점 또한 동굴 입구에서 보인다고 할 수 있습니다.



E. Andrew the Diver

- 동굴 입구에서 보이는 모든 꼭짓점을 입력 순서대로 정렬한 뒤, 단절된 부분의 두 꼭짓점을 조사하면 몇 개의 어두운 영역이 존재하는지 알 수 있습니다.



F. 초코바 만들기

greedy

출제진 의도 - Easy

- 제출 31번, 정답 19팀 (정답률 : 61.290%)
- 처음 푼 팀 : **우리팀명 뭐로할래**[서강대학교], 5분
- 출제자 : **김형진**[plast]

F. 초코바 만들기

- 모든 초코바의 변 중 가장 긴 변의 길이를 X라고 해봅시다.
- 반드시 틀의 한 변은 길이가 X 이상이여야 합니다. 일반성을 잃지 않고, 틀의 가로 길이가 X 이상이라고 하겠습니다.
- 그렇게 되면 아래와 같은 그리디 전략이 생기게 됩니다.
- 이미 틀의 가로 길이는 X 이상입니다. 초코바의 어느 변을 가로에 두더라도, 가로 길이는 초코바의 어느 변의 길이보다 길거나 같습니다.
- 따라서, 모든 초코바는 두 변의 길이 중 큰 변을 가로에 두고, 작은 변을 세로에 두는것이 반드시 좋습니다.
- 그러면 답은 (모든 초코바의 더 큰 변의 길이 중 최대) * (모든 초코바의 더 작은 변의 길이 중 최대) 가 됩니다.

G. 장비 강화하기

greedy, segtree, pbs

출제진 의도 - **Challenging**

- 제출 16번, 정답 2팀 (정답률 : 12.500%)
- 처음 푼 팀 : **대충 26년 전역 어쩌고**연세대학교, 240분
- 출제자 : **유상혁**golazcc83

G. 장비 강화하기

- N개의 장비 중 K개의 장비를 강화한다고 했을 때, B_i개의 강화석을 사용하는 대신 1개의 금화를 추가로 사용할 수 있다는 점을 이용하여 강화할 장비에 우선순위를 정할 수 있습니다.
- i번 장비와 j번 장비의 강화 재료를 비교했을 때
  - $A_i = A_j, B_i < B_j$일 때, i번 장비를 강화하는 것이 이득입니다.
  - $A_i < A_j$일 때, i번 장비를 강화하는 것이 이득입니다. i번 장비에 금화 1개를 추가로 사용하면 $A_i + 1 \leq A_j, 0 < B_j$이기 때문입니다.
- 따라서 장비를 A_i가 증가하는 순으로, A_i가 같다면 B_i가 증가하는 순으로 정렬하여 우선순위가 높은 K개의 장비를 선택할 수 있습니다.

G. 장비 강화하기

- N개의 장비 중 K개의 장비를 강화한다고 했을 때, 강화석을 사용할 장비와 금화 1개를 추가로 사용할 장비를 선택해야 합니다.
- 강화석을 사용할 장비는 강화석이 적게 필요한 장비부터 가능한 만큼 선택하면 됩니다.
- Y개의 강화석을 보유하고 있을 때 강화석을 사용한 장비의 개수를 C라고 하면 필요한 금화의 수는 $K - C + \sum_{i=1}^{K} A_i$ 입니다.
- 이 정보를 바탕으로 X개의 금화와 Y개의 강화석을 가지고 있을 때 K개의 장비를 강화할 수 있는지 계산할 수 있습니다.
- X개의 금화와 Y개의 강화석으로 강화할 수 있는 장비의 수는 K를 이분 탐색하여 구할 수 있습니다. 하지만 구해야 하는 질문의 수는 Q개입니다.

G. 장비 강화하기

- B_i 기준으로 정렬된 자료 구조 S가 있다고 가정합니다. 이 자료 구조는 아래의 쿼리를 수행해야 합니다.
  1. S에 B_i를 추가합니다.
  2. Y가 주어졌을 때, $\sum_{i=1}^{C} S_i \leq Y$인 C의 최댓값을 반환합니다.
- 위의 자료구조는 Segment Tree를 사용하여 구현할 수 있습니다. 먼저 입력으로 주어진 모든 장비를 B_i 기준으로 정렬하고, 각 노드에 장비의 수, 강화석의 수를 관리하면 1번 쿼리를 $\mathcal{O}(\log N)$에 수행할 수 있습니다.
- 2번 쿼리는 아래의 과정을 통해 $\mathcal{O}(\log N)$에 수행할 수 있습니다.
  - 현재 노드를 기준으로 왼쪽 노드의 강화석의 수가 Y보다 크면 왼쪽으로 이동합니다.
  - 아니면 쿼리의 답에 왼쪽 노드의 장비의 수를 더하고, Y를 왼쪽 노드의 강화석의 수만큼 빼고 오른쪽으로 이동합니다.

G. 장비 강화하기

- Q개의 질문에 대한 이분 탐색을 한꺼번에 처리하는 Parallel Binary Search와 앞서 정의한 Segment Tree를 활용해봅시다. 각 질문에 대한 정답의 최솟값, 최댓값을 저장하면서 아래의 과정을 수행합니다.
  1. 답이 정해지지 않은 질문의 중간값을 전처리합니다.
  2. $K = 0$부터 1씩 더하면서 1번 쿼리를 수행하고 K와 중간값이 일치하는 질문에 대해 2번 쿼리를 수행하여 최솟값 또는 최댓값을 변경합니다.
- 위의 과정은 1회당 $\mathcal{O}((N+Q)\log N)$에 수행할 수 있으며, $\log N$에 비례한 횟수만큼 반복되므로 총 시간복잡도는 $\mathcal{O}((N+Q)\log^2 N)$입니다.
- 별해 : Persistent Segment Tree 등의 자료 구조를 사용한다면 $K = 1, 2, \cdots, N$일 때의 상태를 저장할 수 있고, 2번 쿼리를 온라인으로 $\mathcal{O}(\log N)$에 수행할 수 있으므로 각 질문을 $\mathcal{O}(\log^2 N)$에 해결할 수 있습니다.

H. 완전 그래프와 쿼리

math, number_theory, ad_hoc, constructive

출제진 의도 - **Medium**

- 제출 47번, 정답 7팀 (정답률 : 14.894%)
- 처음 푼 팀 : **배신저스**서강대학교, 64분
- 출제자 : **박****isekaijoucho

H. 완전 그래프와 쿼리

- 초기 상태에서 정점의 개수가 N인 간선이 없는 그래프가 주어져 있습니다.
- 현재 상태에서 수행할 수 있는 쿼리 중, 가장 많은 정점을 연결하는 쿼리를 선택하여 수행합니다. 간선으로 연결되지 않은 두 정점 쌍이 존재한다면 이 과정을 반복합니다.
- 그래프의 모든 정점이 서로 연결되었다면, 수행을 종료합니다.
- 이렇게 선택된 쿼리에서 1번 쿼리의 개수는 N 이하의 홀수의 개수, 2번 쿼리의 개수는 N 이하의 합성수의 개수라는 규칙을 찾을 수 있습니다.
- 실제로 N 이하의 모든 홀수에 대해 1번 쿼리를, 모든 합성수에 대해 2번 쿼리를 수행하면 완전 그래프가 된다는 것을 관찰할 수 있습니다.
- 그러나, 여전히 모든 N에 대해 이 풀이가 항상 성립하는지 확신할 수 없는 상태입니다.

H. 완전 그래프와 쿼리

- 문제를 엄밀히 증명하기 위해 해결해야 하는 내용은 다음과 같습니다.
  1. "N 이하의 모든 홀수에 대해 1번 쿼리를, 모든 합성수에 대해 2번 쿼리를 수행하여 완전 그래프를 구성할 수 있는가?"
  2. "그렇게 선택된 쿼리들이 가장 적은 방법으로 완전 그래프를 구성하는가?"
- 이제부터 이 두 가지 의문을 엄밀하게 증명하고자 합니다.

H. 완전 그래프와 쿼리

- 간선들의 집합 $p_N(v)$와 $q_N(v)$를 다음과 같이 정의하겠습니다.
  - $p_N(v) :=$ N개의 정점을 갖는 그래프에서 $(1,\ v)$ 쿼리로 생성되는 간선들의 집합
  - $q_N(v) :=$ N개의 정점을 갖는 그래프에서 $(2,\ v)$ 쿼리로 생성되는 간선들의 집합
- 가장 먼저, 두 쿼리를 이용하여 완전 그래프를 항상 구성할 수 있음을 보이겠습니다.
- **Theorem 1** $p_N(v)$와 $q_N(v)$는 다음을 만족합니다.
  - $\bigcup_{i=1}^{N} p_N(i) \cup \bigcup_{i=1}^{N} q_N(i) = K_N$ (정점의 개수가 N인 완전 그래프)
  - $\bigcup_{i=1}^{N} p_N(i) \cap \bigcup_{i=1}^{N} q_N(i) = \emptyset$

H. 완전 그래프와 쿼리

- **Proof:**
  1. 임의의 $1 \leq i \leq N$에 대하여, 쿼리의 정의에 의해 $p_N(i) \cap q_N(i) = \emptyset$ 이고,
     $p_N(i) \cup q_N(i) =$ (정점 i와 연결된 모든 간선들의 집합) 입니다.
  2. 따라서, $\bigcup_{i=1}^{N} p_N(i) \cap \bigcup_{i=1}^{N} q_N(i) = \emptyset$ 이고, $\bigcup_{i=1}^{N} p_N(i) \cup \bigcup_{i=1}^{N} q_N(i) = K_N$ 입니다.
- **Theorem 1**에 의해 두 종류의 쿼리로 완전 그래프를 구성할 수 있음을 보였습니다.
- 추가로 1번 쿼리로 생성된 간선들의 집합과 2번 쿼리로 생선된 간선들의 집합은 서로 독립적인 것을 확인할 수 있습니다.
- 따라서 그래프의 구성에 대한 증명을 두 가지 주장으로 나누어 해결할 수 있습니다.

H. 완전 그래프와 쿼리

- **Claim 1** $\bigcup_{i=1}^{N} p_N(i) = \bigcup_{i \text{ is odd}} p_N(i)$
- **Proof:**
  1. $\bigcup_{i \text{ is odd}} p_N(i) \subseteq \bigcup_{i=1}^{N} p_N(i)$ 임은 자명합니다.
  2. $\{a,\ b\} \in \bigcup_{i=1}^{N} p_N(i)$ 라 하겠습니다. $\gcd(a,\ b) = 1$ 이므로, (WLOG) 만약 a가 짝수이면, b는 반드시 홀수입니다.
  3. 따라서 $\{a,\ b\} \in \bigcup_{i \text{ is odd}} p_N(i)$ 입니다.

H. 완전 그래프와 쿼리

- **Claim 2** $\bigcup_{i=1}^{N} q_N(i) = \bigcup_{i \text{ is composite}} q_N(i)$
- **Proof:**
  1. $\bigcup_{i \text{ is composite}} q_N(i) \subseteq \bigcup_{i=1}^{N} q_N(i)$ 임은 자명합니다.
  2. $\{a,\ b\} \in \bigcup_{i=1}^{N} q_N(i)$ 라 하겠습니다. $\gcd(a,\ b) \neq 1$ 이므로, (WLOG) 만약 a가 소수이면, b는 a를 인수로 갖는 합성수입니다.
  3. 따라서 $\{a,\ b\} \in \bigcup_{i \text{ is composite}} q_N(i)$ 입니다.

H. 완전 그래프와 쿼리

- **Theorem 1**과 **Claim 1, 2**에 의해 다음이 성립합니다.
  - $\bigcup_{i \text{ is odd}} p_N(i) \cup \bigcup_{i \text{ is composite}} q_N(i) = K_N$
  - $\bigcup_{i \text{ is odd}} p_N(i) \cap \bigcup_{i \text{ is composite}} q_N(i) = \emptyset$
- 따라서 "N 이하의 모든 홀수에 대해 1번 쿼리를, 모든 합성수에 대해 2번 쿼리를 수행하여 완전 그래프를 구성할 수 있는가?"에 대한 증명이 완료되었습니다.
- 이제 완전 그래프를 구성하도록 하는 쿼리의 최소 수행 횟수의 하한이 (N 이하의 홀수의 개수) + (N 이하의 합성수의 개수) 인지 증명하고자 합니다.
- 그러기에 앞서, 두 개의 정리를 증명해야 합니다.

H. 완전 그래프와 쿼리

- **Theorem 2** 양의 정수 N에 대하여, $a_1, a_2, \ldots, a_k$는 서로 다른 양의 정수로 이루어진 수열이며, 모든 a_i는 N 이하입니다. 이때, 서로 다른 i와 j에 대하여, $\gcd(a_i, a_j) \neq 1$을 만족하는 수열의 길이를 k라고 하겠습니다.
  - k의 최댓값은 N 이하의 짝수의 개수와 같습니다.
  - 이 수열은 N 이하의 서로 다른 모든 짝수들로 이루어져 있습니다.
- **Proof:**
  1. 조건을 만족하는 수열의 존재성은 자명합니다.
  2. 길이가 $m > \lfloor \frac{N}{2} \rfloor$ 인 더 긴 수열 $b_1, b_2, \ldots, b_m$이 존재한다고 가정하겠습니다.
  3. N이 짝수인 경우를 생각해봅시다. m개의 정수를 선택하면 비둘기집의 원리에 의해, $|b_i - b_j| = 1$인 어떤 $i, j (1 \leq i, j \leq m, i \neq j)$가 존재하게 됩니다.

H. 완전 그래프와 쿼리

4. 연속한 두 정수는 항상 서로소이므로, N이 짝수이면 가정에 의해 모순입니다.
5. N이 홀수인 경우를 생각해봅시다. m이 N 이하의 홀수의 개수, $\lceil \frac{N}{2} \rceil$ 보다 크면, 연속한 두 수를 포함하게 되므로, $m \leq \lceil \frac{N}{2} \rceil$ 입니다.
6. $\lfloor \frac{N}{2} \rfloor + 1 = \lceil \frac{N}{2} \rceil$ 이고, $m = \lceil \frac{N}{2} \rceil$ 인 연속된 정수를 포함하지 않는 수열은 홀수인 수열이 유일합니다.
7. 이 수열은 1을 포함하고, 1은 모든 수와 서로소이므로, N이 홀수이면 가정에 의해 모순입니다.
8. 따라서, 최대 길이 k는 $\lfloor \frac{N}{2} \rfloor$ 이며, 이는 N 이하의 짝수로 이루어진 수열입니다.

H. 완전 그래프와 쿼리

- **Theorem 3** 양의 정수 N에 대하여, $a_1,\ a_2,\ \ldots,\ a_k$는 서로 다른 양의 정수로 이루어진 수열이며, 모든 a_i는 N 이하입니다. 이때, 서로 다른 i와 j에 대하여, $\gcd(a_i,\ a_j) = 1$을 만족하는 수열의 길이를 k라고 하겠습니다.
  - k의 최댓값은 1을 포함한 N 이하의 소수의 개수 $\pi(N) + 1$과 같습니다.
  - 이 수열은 1과 N 이하의 모든 소수들로 이루어져 있습니다.
- **Proof:**
  1. 조건을 만족하는 수열의 존재성은 자명합니다.
  2. 길이가 $m > \pi(N) + 1$ 인 더 긴 수열 $b_1,\ b_2,\ \ldots,\ b_m$이 존재한다고 가정하겠습니다.
  3. 서로 다른 i와 j에 대하여, b_i와 b_j가 공통 소인수 p를 공유하면 $\gcd(b_i,\ b_j) \geq p > 1$ 이므로 각 소인수는 수열 내에서 최대 한 번만 등장할 수 있습니다.

H. 완전 그래프와 쿼리

4. N 이하의 자연수 c에 대하여, c를 자기 자신의 최소 소인수에 대응시키는 함수 f를 생각해봅시다.
5. c가 합성수이면, $f(c)$는 c의 최소 소인수에 대응됩니다. c가 소수이면, $f(c)$는 자기 자신에 대응됩니다. c가 1이면 1에 대응됩니다.
6. 즉, 수열의 총 원소 수는 1을 포함한 N 이하의 소수 개수를 초과할 수 없습니다. 이는 $m > \pi(N) + 1$이라는 가정에 모순입니다.
7. 따라서 최대 길이 k는 $\pi(N) + 1$이며, 이는 1과 N 이하의 소수로 이루어진 수열입니다.

H. 완전 그래프와 쿼리

- P_N, Q_N 그리고 $\#$-function을 다음과 같이 정의하겠습니다.
  - $P_N = \bigcup_{i \text{ is odd}} p_N(i)$
  - $Q_N = \bigcup_{i \text{ is composite}} q_N(i)$
  - $\# : \bigcup_{i \in A} x(i) \to A$
- **Theorem 1**에 의해 쿼리의 최소 수행 횟수 또한 1번 쿼리와 2번 쿼리로 나누어 생각해볼 수 있습니다.

H. 완전 그래프와 쿼리

- **Claim 3** 임의의 $S_N = \bigcup_{i \in A} p_N(i)$에 대하여, $S_N = \bigcup_{i=1}^{N} p_N(i)$ 일 때, $|\#(P_N)| \leq |\#(S_N)|$
- **Proof:**
  1. $|\#(P_N)| > |\#(S_N)|$ 라 가정하겠습니다.
  2. 가정에 의해, 집합 A에 대하여, $A = \{1, 2, \ldots, N\} \setminus B$이고 $|B| > \lfloor \frac{N}{2} \rfloor$을 만족하는 어떤 집합 B가 존재합니다.
  3. $S_N = \bigcup_{i=1}^{N} p_N(i)$이므로 집합 B의 서로 다른 임의의 두 원소 $b_i,\ b_j (i \neq j)$가 $\gcd(b_i,\ b_j) \neq 1$를 만족해야 합니다.
  4. **Theorem 2**에 의해, $|B| \leq \lfloor \frac{N}{2} \rfloor$이므로 이러한 집합 B가 존재한다는 것은 모순입니다.
  5. 따라서 **Claim 3**이 성립합니다.

H. 완전 그래프와 쿼리

- **Claim 4** 임의의 $T_N = \bigcup_{i \in A} q_N(i)$에 대하여, $T_N = \bigcup_{i=1}^{N} q_N(i)$ 일 때, $|\#(Q_N)| \leq |\#(T_N)|$
- **Proof:**
  1. $|\#(Q_N)| > |\#(T_N)|$ 라 가정하겠습니다.
  2. 가정에 의해, 집합 A에 대하여, $A = \{1, 2, \ldots, N\} \setminus B$이고 $|B| > \pi(N) + 1$을 만족하는 어떤 집합 B가 존재합니다.
  3. $T_N = \bigcup_{i=1}^{N} q_N(i)$이므로 집합 B의 서로 다른 임의의 두 원소 $b_i,\ b_j (i \neq j)$가 $\gcd(b_i,\ b_j) = 1$를 만족해야 합니다.
  4. **Theorem 3**에 의해, $|B| \leq \pi(N) + 1$이므로 이러한 집합 B가 존재한다는 것은 모순입니다.
  5. 따라서 **Claim 4**가 성립합니다.

H. 완전 그래프와 쿼리

- **Claim 3, 4**에 의해 다음이 성립합니다.
  - $\bigcup_{i\in A} p_N(i) = \bigcup_{i=1}^{N} p_N(i)$을 만족하는 $|A|$의 최솟값은 $|\#(P_N)|$
  - $\bigcup_{i\in A} q_N(i) = \bigcup_{i=1}^{N} q_N(i)$을 만족하는 $|A|$의 최솟값은 $|\#(Q_N)|$
- 따라서 "그렇게 선택된 쿼리들이 가장 적은 방법으로 완전 그래프를 구성하는가?"에 대한 증명이 완료되었습니다.
- 이로써 문제의 정해에 대한 정당성이 모두 증명되었습니다.
- 에라토스테네스의 체를 이용하여 $\mathcal{O}(N \log\log N)$에 구현하거나 소수 판별 알고리즘으로 $\mathcal{O}(N\sqrt{N})$에 구현할 수 있습니다.

I. 체리 컴퍼니

offline query, segment tree, sweeping

출제진 의도 - **Hard**

- 제출 19번, 정답 8팀 (정답률 : 47.368%)
- 처음 푼 팀 : **나근육또녹을지도**[서강대학교], 125분
- 출제자 : **박진한**[jinhan814]

I. 체리 컴퍼니

- i번 직원을 i번 정점으로 볼 때 $2 \le i \le N$에 대해 (i, P_i) 간선을 추가해서 만들어지는 그래프는 트리입니다.
- L_i, R_i에 대한 정답은 그래프에서 양 끝 점의 인덱스가 모두 $[L_i, R_i]$ 범위에 속하는 간선의 개수를 $f(L_i, R_i)$라 할 때 $(R_i - L_i + 1) - f(L_i, R_i)$입니다.
- $f(L_i, R_i)$는 간선 (i, P_i)를 2차원 평면 위의 점으로 생각하면 $[L_i, R_i] \times [L_i, R_i]$ 영역의 점 개수와 같으니 2차원 구간 합 쿼리로 환원할 수 있습니다.
- 쿼리를 offline으로 처리하면 segment tree와 sweeping으로 2차원 구간 합 쿼리를 해결할 수 있습니다.
- 풀이의 시간복잡도는 $\mathcal{O}((N+Q)\log N)$입니다.
- 이외에도 Mo's를 이용한 $\mathcal{O}((N+Q)\sqrt{N})$ 풀이나 PST를 이용한 $\mathcal{O}((N+Q)\log N)$ 풀이가 가능합니다.

J. blobnom.xyz

binary_search, prefix_sum

출제진 의도 - Easy

- 제출 31번, 정답 18팀 (정답률 : 58.065%)
- 처음 푼 팀 : **나근육또녹을지도**[서강대학교], 8분
- 출제자 : **장래오**[leo020630]

J. blobnom.xyz

- 먼저 사용자의 실력 B가 주어졌을 때 해결할 수 있는 문제 수를 구해야 합니다.
- $S[x]$를 난이도가 x 이하인 문제의 수로 정의합시다. 이는 누적 합 배열을 통해 구성할 수 있습니다.
- 실력이 B인 사용자가 해결할 수 있는 문제의 수는 $S[B]$입니다.
- 만약 $S[B] = 0$이라면 답 역시 0입니다.
- 그렇지 않다면, $3k(k-1)+1 \leq S[B]$인 k의 최댓값을 구해야 합니다.

J. blobnom.xyz

- Solution 1.
  - $F[i] = 3i(i-1) + 1$로 정의된 배열을 생각해 봅시다.
  - 답은 이 배열에서 $S[B]$의 upper bound를 구한 뒤 1을 뺀 것과 같습니다.
  - 시간 복잡도는 $\mathcal{O}(N + M \log N)$ 입니다.
- Solution 2.
  - 문제의 제한에서 가능한 최대 k는 316입니다.
  - 따라서 각 사용자에 대한 정답을 나이브하게 구해도 시간 안에 문제를 해결할 수 있습니다.
  - 시간 복잡도는 $\mathcal{O}(N + Mk)$ 입니다.
- 이외에도 여러 방법으로 문제를 해결할 수 있습니다.

K. 학식 뭐 먹지

dp, knapsack, greedy

출제진 의도 - **Medium**

- 제출 41번, 정답 9팀 (정답률 : 21.951%)
- 처음 푼 팀 : **나근육또녹을지도**서강대학교, 49분
- 출제자 : **김영인**young_out

K. 학식 뭐 먹지

- 불만도를 최소로 하는 메뉴 번호들의 집합을 떠올려봅시다. 그중 가격이 최대인 메뉴의 번호를 k라 할 때, k번 메뉴는 B_k개 이하로 쓰이고 $i(i \neq k)$번 메뉴는 B_i개만큼 쓰이는 것이 최선임을 알 수 있습니다.
- 즉, 가격이 낮은 메뉴부터 최대한 많이 쓰는 것이 최선입니다.
- $dp[n][m] = n$명이 m개의 메뉴 중에 선택한 가격의 합의 최솟값이라고 정의합시다. 이때, 불만도는 $dp[n][m] \times m$이 됩니다.
- 아무것도 선택하지 않은 상태를 $dp[0][0] = 0$이라 할 수 있습니다. 또한 가격이 낮은 메뉴가 높은 메뉴보다 우선적으로 쓰이는 것이 최선이므로, 메뉴들을 가격에 대한 오름차순으로 정렬합시다.

K. 학식 뭐 먹지

- 특정한 메뉴의 가격 A_i와 수량 B_i가 주어졌을 때, 현재 상태 $dp[n][m]$에 대해 다음을 수행할 수 있습니다.
- $n_{new} = min(n + B_i, N)$
- $dp[n_{new}][m+1] = min(dp[n_{new}][m+1], dp[n][m] + (n_{new} - n) \times A_i)$
- 이러한 작업을 가격이 낮은 메뉴부터 높은 메뉴까지 모든 메뉴에 대해 수행하면 됩니다. 같은 메뉴에 대해서는 한 번만 수행하도록 뒤에서부터 Knapsack 방식으로 업데이트합니다.
- 따라서 불만도는 $dp[N][m] \times m (1 \le m \le M)$의 최솟값입니다.
- M개의 메뉴에 대해 $dp[n][m]$을 업데이트해야 하므로 시간복잡도는 $\mathcal{O}(NM^2)$입니다.

L. Anti-Fan Death

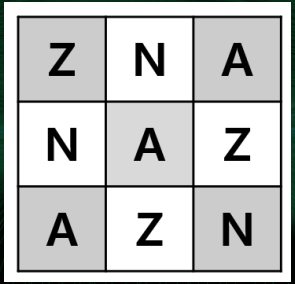
Ad-hoc

출제진 의도 - Easy

- 제출 47번, 정답 16팀 (정답률 : 34.043%)
- 처음 푼 팀 : **대충 26년 전역 어쩌고**연세대학교, 12분
- 출제자 : **정치훈**wjdclgns12

L. Anti-Fan Death

- 우선 가장 작은 $3N \times 3N$ 크기의 답을 찾아봅시다.
- 완전탐색 등으로 쉽게 답을 찾을 수 있습니다.



L. Anti-Fan Death

- 이를 기반으로 일반 해를 찾으면 됩니다.
- 예를 들어, 각 칸의 크기가 1×1보다 커도, 문제의 조건을 여전히 만족함을 알 수 있습니다.

Z	Z	N	N	A	A
Z	Z	N	N	A	A
N	N	A	A	Z	Z
N	N	A	A	Z	Z
A	A	Z	Z	N	N
A	A	Z	Z	N	N

M. 신촌 길찾기 서비스

shortest_path

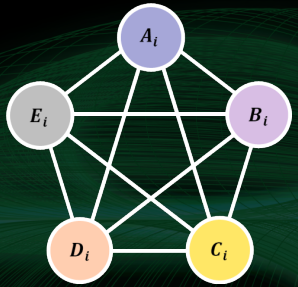
출제진 의도 - **Medium**

- 제출 19번, 정답 10팀 (정답률 : 52.632%)
- 처음 푼 팀 : **나근육또녹을지도**[서강대학교], 88분
- 출제자 : **유상혁**[golazcc83]

M. 신촌 길찾기 서비스

- 두 정류장 사이를 이동하는 경로는 (정류장 → 버스 → 정류장 → 버스 → ... → 정류장 → 버스 → 정류장)과 같이 나타낼 수 있습니다.
- 정류장을 정점으로, 버스를 간선으로 생각한다면 정점의 개수는 최대 $100\,000$개이므로, 각 질문이 주어질 때마다 해당 정점들을 방문하며 최단 거리를 계산하므로 시간 초과를 받습니다.
- 버스 노선의 개수는 최대 500개라는 점을 이용해서 버스를 정점으로 하여 그래프를 구성해봅시다.

M. 신촌 길찾기 서비스



- i번 정류장을 지나는 버스노선 A_i, B_i, C_i, D_i, E_i는 i번 정류장을 통해 환승할 수 있으므로, 위 그림과 같이 길이 1의 간선으로 서로 연결되어 있다고 볼 수 있습니다. 이를 활용하여 버스를 정점으로 그래프를 구성하면 정점이 $5X$개인 그래프를 구성할 수 있습니다.

M. 신촌 길찾기 서비스

- 서로 다른 두 버스 사이의 최단 거리를 미리 계산할 수 있습니다.
- 미리 계산하는 방법으로는 각 정점을 시작점으로 하는 너비 우선 탐색을 $5X$번 수행거나, 플로이드 워셜을 수행하는 방법이 있습니다. 두 방법 모두 $\mathcal{O}(125X^3)$의 시간 복잡도를 가지며 주어진 제한에서 대략 1억이므로 제한 시간 내에 통과합니다.
- 각 정점을 시작점으로 하는 다익스트라를 $5X$번 수행하는 방법으로도 제한 시간 내에 통과할 수 있습니다.

M. 신촌 길찾기 서비스

- 두 버스 사이의 최단 거리를 미리 계산했다면, 두 정류장 사이의 최단 거리도 계산할 수 있습니다.
- 각 정류장을 지나는 5개의 버스를 각각 시작, 도착 정점으로 했을 때 25가지 경우의 수의 최단 거리 중 가장 짧은 거리를 출력하면 됩니다.
- 총 시간복잡도는 $\mathcal{O}(25N + 125X^3 + 25Q)$ 입니다.
- 별해로 정류장 N개와 버스 $5X$개를 정점으로 하고, 정류장과 버스를 간선 $5N$로 연결하여 너비 우선 탐색을 $5X$번 수행하는 풀이가 있습니다.