2023 ICPC Sinchon Summer Algorithm Camp Contest Solution

Official Solutions

신촌지역 대학생 프로그래밍 동아리 연합

2023년 8월 20일

240

문제		의도한 난이도	출제자
1A	Toe Jumps	**Beginner**	정회윤 yunny_world
1B	Potato	**Easy**	채성우 lem0nad3
1C	피보나치 사각형	**Easy**	이협 dlguq0107
1D	좋은 팀이란?	**Easy**	길수민 2093ab
1E	Spell Cards	**Medium**	채성우 lem0nad3
1F	납이야 나비야	**Hard**	최우현 starwh03
1G	Equilibruim Points	**Hard**	정회윤 yunny_world
2A	슥 빡! 빡 슥!	**Easy**	정회윤 yunny_world
2B	W키가 빠진 성원이	**Easy**	이도훈 dhlee1031
2C	Minimax Tree	**Medium**	길수민 2093ab
2D	도로 위의 표지판	**Medium**	이도훈 dhlee1031
2E	피보나치 서로소	**Hard**	이협 dlguq0107
2F	편광판	**Hard**	최우현 starwh03
2G	나비야 나비야	**Challenging**	채성우 lem0nad3

240

1A. Toe Jumps

`implementation`

출제진 의도 – **Beginner**

- ✓ 제출 18번, 정답 11명 (정답률 61.111%)
- ✓ 처음 푼 사람: **변미담**, 14분
- ✓ 출제자: `yunny_world`

1A. Toe Jumps

240

- ✓ 문자열을 이용해 모든 경우를 구현하면 됩니다.
- ✓ 4개의 방향과 3개의 점프가 있으니 12가지를 구분하면 됩니다.

240

1B. Potato

`sorting, greedy`
출제진 의도 – **Easy**

- ✓ 제출 20번, 정답 13명 (정답률 65.000%)
- ✓ 처음 푼 사람: **변미담**, 23분
- ✓ 출제자: `lem0nad3`

1B. Potato

240

- 남아있는 접시들 중, 박 모 씨는 최대의 a 값을 가진 접시를 가져가는 것이 이득입니다.
- 성우는 최소의 a 값을 가진 접시를 가져가는 것이 이득입니다.
- 박 모 씨가 먼저 접시를 가져가기 때문에, 박 모 씨는 $\left\lceil \frac{N}{2} \right\rceil$ 개의 접시를 가져갈 수 있습니다.
- a 를 오름차순으로 정렬합시다.
- 왼쪽 $N - \left\lceil \frac{N}{2} \right\rceil$ 개는 성우가 가져갑니다.
- 오른쪽 $\left\lceil \frac{N}{2} \right\rceil$ 개는 박 모 씨가 가져갑니다.
- 이를 각각 출력하면 됩니다.

240

1C. 피보나치 사각형

`divide and conquer`

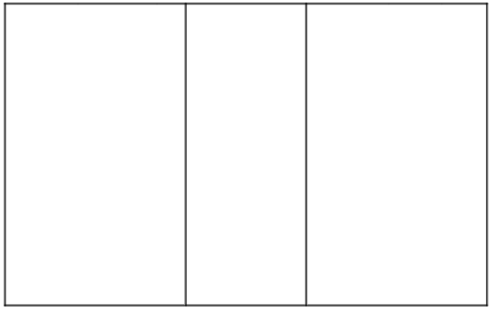
출제진 의도 – **Easy**

- ✓ 제출 4번, 정답 0명 (정답률 0%)
- ✓ 처음 푼 사람: -, -분
- ✓ 출제자: `dlguq0107`

1C. 피보나치 사각형

240

- ✓ 우선 그림이 세로가 긴 경우 90도 회전하여 가로가 길도록 만들 수 있습니다.
- ✓ i번째 피보나치 사각형을 한번 살펴봅시다.



- ✓ 사각형을 관찰해보면 가운데 영역에 점이 포함되는 경우 현재 피보나치 사각형이 최소임을 알 수 있습니다.
- ✓ 이외의 경우, $i-1$ 번째 피보나치 사각형에 포함되고 이는 재귀적으로 해결할 수 있습니다.
- ✓ 중앙 부분의 영역을 식으로 쓰면

$$F_{i-1} \le x \le F_i, \quad 0 \le y \le F_i$$

이므로 이를 통해 중앙에 점이 포함되는지 안 되는지 판별이 가능합니다.

1D. 좋은 팀이란?

`bruteforcing`
출제진 의도 – **Easy**

- ✓ 제출 11번, 정답 1명 (정답률 9.091%)
- ✓ 처음 푼 사람: **김현서**, 152분
- ✓ 출제자: `2093ab`

- naive한 접근인 $\mathcal{O}(N^3)$ 은 시간 초과를 받게 될 것입니다.
- 나올 수 있는 일주는 총 60 가지입니다.
- 따라서 **비둘기집의 원리**에 따라, 주어지는 입력이 61 가지 이상이면 같은 일주가 적어도 하나 이상 존재합니다.
- 각 일주마다 주어지는 실력 값을 모두 저장하여 정렬해 놓습니다.
- 팀원에 포함될 일주를 $\mathcal{O}(60^3)$ 혹은 $\mathcal{O}(120^3)$ 으로 선택합니다.
- 한 케이스마다 각 일주별의 관계 점수를 계산하고, 선택된 일주마다 실력 값이 큰 것부터 꺼내서 더합니다.

1D. 좋은 팀이란?

240

- 만약 선택된 일주에서 꺼내야 하는 실력 값의 개수보다 저장된 값의 개수가 작은 경우는 계산하지 않고 넘어갑니다.
  - 같은 일주가 여러 번 선택될 수 있음에 유의하시기 바랍니다.
- 계산된 값들 중 최댓값을 찾아 출력하면 됩니다.
- 이외에도 일주, 실력 값을 모두 선택하는 $\mathcal{O}(300^3)$ 풀이도 가능합니다.

240

1E. Spell Cards

dp

출제진 의도 – **Medium**

- ✓ 제출 3번, 정답 0명 (정답률 0%)
- ✓ 처음 푼 사람: -, -분
- ✓ 출제자: `lem0nad3`

- $f(l, r) =$ (구간 $[l, r]$의 모든 카드를 하나로 합칠 때 드는 마력 소모량의 최솟값)으로 정의합시다.
- $r \leq l$인 경우, $f(l, r) = 0$입니다. $l < r$인 경우를 생각해봅시다.
- $g(l, r) = \max_{l \leq i \leq r} a_i$로 정의합시다.
- $f(l, r) = \min_{l \leq i \leq r-1} (f(l, i) + f(i + 1, r) + g(l, i) + g(i + 1, r))$으로 두고, 메모이제이션을 이용해 재귀적으로 해결할 수 있습니다.

- ✓ $g(l, r)$의 값을 필요할 때마다 매번 구하게 되면 $\mathcal{O}(N^4)$이므로, 시간 초과를 받습니다.
- ✓ $g(l, r)$을 $\mathcal{O}(N^2), \mathcal{O}(N^3)$ 등에 전처리하면 됩니다.
- ✓ 전체 시간복잡도는 $\mathcal{O}(N^3)$입니다.

- ✓ kipa00님께서 제공해주신 그리디 풀이도 소개하겠습니다. 다음과 같습니다.
  1. 현재 남아있는 카드 중 마력 소모량 값이 가장 작은 카드를 고른다.
  2. 이 카드와 맞닿은 양 옆의 두 카드 중, 마력 소모량 값이 더 작은 쪽과 합친다.
  3. 1.과 2.를 카드가 단 하나만 남을 때까지 반복한다.
- ✓ 이 과정을 naive하게 수행하면 $\mathcal{O}(N^2)$ 입니다.

- ✓ 이 풀이가 왜 되는 걸까요?
- ✓ 문제의 조건을 다음과 같이 고쳐 봅시다.
  - 인접한 두 카드를 골라, 둘 중 작은 쪽의 카드를 없애는 비용이 **두 카드에 적힌 수의 합**
  - 카드를 하나로 만드는 데에 드는 비용을 최소화
- ✓ 위 조건의 굵은 글씨를 **큰 쪽의 카드에 적힌 수**로 바꾸어도 됩니다.
  - 적힌 수가 가장 큰 카드를 빼고 모든 카드는 정확히 한 번씩 없어집니다.
  - 따라서 조건을 바꾸어 답을 구한 뒤,
    - ▶ 맨 처음에 모든 카드에 적혀 있던 수의 합을 더하고,
    - ▶ 맨 처음에 모든 카드에 적혀 있던 수 중 가장 큰 값을 빼면,

    그 값이 답이 됩니다.

- ✓ 문제를 이렇게 바꾸고, i 번째 카드에 적힌 수를 a_i, i 번째 카드가 없어질 때 들었던 비용을 c_i 라고 합시다.
- ✓ **모든** 없애는 순서에 대해, $c_i \geq \min(a_{i-1}, a_{i+1})$ 입니다. ($a_0 = a_{N+1} = \infty$)
  - 어떤 카드는 인접한 두 쌍 중에서 적힌 수가 작았기 때문에 사라집니다.
  - 따라서 i 번째 카드 주위가 사라졌다면, 그 카드는 적힌 수가 더 크거나 같은 카드로 대체되었을 것입니다.

- ✓ 이제 카드가 사라지는 어떤 순서가 주어지면, 동일한 순서로 없애되 최솟값이 적힌 i번째 카드를 가장 먼저 없앨 때 모든 c_j가 같거나 줄어듦을 보입니다.
- ✓ 일단 이 과정이 가능할까요?
  - 다른 모든 카드는 i번째 카드와 함께 골라져서 사라질 수 없습니다.
  - i번째 카드가 사라지고 그 카드를 건너는 두 카드를 골라서 사라질 수는 있습니다.
  - 따라서 i번째 카드의 제거 순서를 **앞으로** 옮기는 것은 상관없습니다.
- ✓ 맨 앞으로 옮겼다고 합시다.
  - c_i는 이론상 최솟값이 되었으므로, 당연히 이전보다 같거나 줄어듭니다.
  - 나머지 값들은 기존의 카드를 그대로 선택해 제거될 수 있으므로, c_j 값은 최소한 유지됩니다.
- ✓ 이 말은 i번째 카드가 가장 먼저 사라지는 순서 중 비용이 최소가 되는 배치가 있다는 뜻입니다!

1. 현재 남아있는 카드 중 마력 소모량 값이 **가장 작은 카드를 고른다**.
2. 이 카드와 맞닿은 **양 옆의 두 카드** 중, 마력 소모량 값이 더 작은 쪽과 합친다.
3. 1.과 2.를 카드가 단 하나만 남을 때까지 반복한다.

- ✓ 우리는 이 풀이의 정당성을 증명했습니다.
- ✓ 이제 이 풀이를 $\mathcal{O}(N \log N)$에 구현하는 방법을 소개합니다.
- ✓ 1.의 과정은 a_i가 작은 순서대로 정렬한 순서입니다.
- ✓ 없어진 카드를 실제로 없애는 대신 -1을 집어넣으면, 2.의 굵은 글씨 부분이 문제가 됩니다.
- ✓ 해결하는 한 가지 방법을 소개합니다.

- union-find에는 두 가지 최적화가 있습니다.
  - path compression: 나의 루트를 찾을 때 등장했던 모든 간선을 루트로 직접 연결한다.
  - rank compression: 노드의 개수가 적은 트리를 많은 트리에 합친다.
- 둘 중 하나만 해도 $\mathcal{O}(N \log N)$ 시간 복잡도임이 잘 알려져 있습니다.
  - 뒤엣것을 포기하면, 합칠 때 부모를 마음대로 지정해 줄 수 있습니다!

- ✓ 내 수 앞/뒤로 -1이 이어지는 부분수열을 앞/뒤-run이라고 합시다.
- ✓ 내 앞-run과 뒤-run을 뭉치는 union find 자료구조를 각각 하나씩 관리합니다.
  - 이때 부모는 끝쪽 -1로 합니다.
- ✓ 예를 들어 현재 수열이 -1 3 -1 -1 5 -1 2 -1이면,
  - 앞-run: [**-1** 3] [**-1** -1 5] [**-1** 2] [**-1**]
  - 뒤-run: [**-1**] [3 -1 **-1**] [5 **-1**] [2 **-1**]
- ✓ 이제 union find에서 내 부모를 찾고 ± 1을 하면 나와 인접한 카드도 빠르게 찾을 수 있습니다.
- ✓ 이렇게 하여 전체 시간복잡도 $\mathcal{O}(N \log N)$을 달성할 수 있습니다.

240

1F. 납이야나비야

`graph`, `inclusion and exclusion`
출제진 의도 – **Hard**

- ✓ 제출 0번, 정답 0명 (정답률 0%)
- ✓ 처음 푼 사람: -, -분
- ✓ 출제자: `starwh03`

1F. 납이야 나비야

240

- ✓ 우선 나비의 형태를 생각해 봅시다.
- ✓ 나비는 간선을 서로 공유하지 않는 두 개의 길이 3 사이클과 서로 다른 두 개의 간선이 딱 하나의 정점을 공유하고 있습니다.
- ✓ 한 정점에서 찾을 수 있는 사이클 쌍의 개수와 사이클에 포함된 간선을 제외하고 남은 간선 중 2 개를 고르는 경우의 수를 곱한다면 한 정점에서 만들 수 있는 나비의 개수를 알 수 있습니다.
- ✓ 그렇다면 사이클 쌍의 개수는 어떻게 알 수 있을까요?

- ✓ 우선 존재할 수 있는 사이클의 개수를 먼저 세어 봅시다.
- ✓ 정점으로부터 인접한 다른 정점 두 개를 골랐을 때 그 두 정점이 연결되어 있다면 사이클 하나가 존재하게 됩니다.
  - 이 방식으로 $\mathcal{O}(V^2)$에 사이클의 개수 c를 알 수 있습니다.
  - 서로 다른 두 쌍의 개수는 $\frac{c(c-1)}{2}$와 같습니다.
- ✓ 하지만 여기에는 사이클이 하나의 정점을 공유하는 경우도 포함되어 있습니다.

240

- ✓ 이 경우는 처음 인접한 정점을 방문할 때 정점이 세어진 횟수를 통해 알 수 있습니다.
- ✓ 한 정점이 세어진 횟수 t 라고 한다면 $\frac{t(t-1)}{2}$ 개의 사이클 쌍이 겹친다는 것을 알 수 있습니다.
- ✓ 모든 인접한 정점에서 겹치는 횟수를 빼 준다면 한 정점에서 나비의 개수를 알 수 있습니다.
- ✓ 모든 정점에 대해 위 과정을 진행하면 되므로 총 시간복잡도는 $\mathcal{O}(V^3)$ 이 됩니다.

240

1G. Equilibrium Points

`binary_search`
출제진 의도 – **Hard**

- ✓ 제출 0번, 정답 0명 (정답률 0%)
- ✓ 처음 푼 사람: -, -분
- ✓ 출제자: `yunny_world`

- 전하의 크기가 $+1$인 점전하와 모든 고정된 점전하 사이의 작용하는 전기력이 0이라면 $+1$ 크기의 점전하는 움직이지 않는다.
- $+1$ 크기의 점전하를 좌표 x에 놓았을 때, 해당 점전하가 받는 힘의 크기를 식으로 나타내면 다음과 같다.

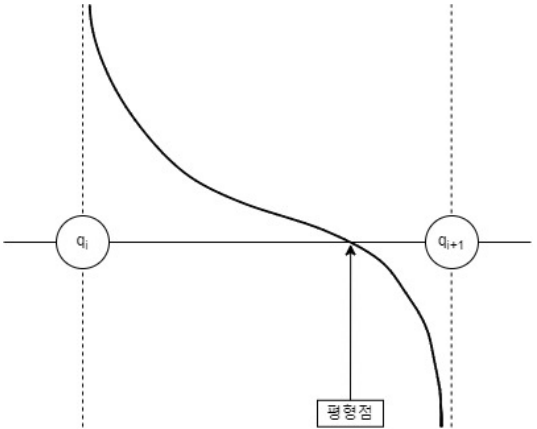
$$\sum_{i=1}^{j} \frac{q_i}{(x-x_i)^2} - \sum_{i=j+1}^{N} \frac{q_i}{(x-x_i)^2} \quad (x_j \le x \le x_{j+1})$$

- 이 식의 값이 0이 되는 x 좌표를 '평형점'이라 하자.

1G. Equilibrium Points

240

- 앞서 나온 식을 어떤 두 연속된 점전하 사이에서 '위치에 따른 받는 힘' 함수의 개형을 그리면 다음 그림과 같이 나타난다.



- 따라서, 모든 두 연속한 점전하 사이에서 평형점이 한 개씩 존재하게 된다.
- 여러 평형점 중 좌표의 크기가 최소인 곳의 위치를 출력해야 하므로, 첫 번째와 두 번째 사이에 존재하는 평형점의 좌표를 출력하면 된다.
- 두 연속한 점전하 사이의 구간에서 받는 힘의 합은 단조 감소하므로, 이분 탐색을 통해 찾을 수 있다.

- ✓ 만약, C++의 `float`과 같은 single-precision 실수 자료형을 사용하여 이분 탐색 시 무한 루프에 빠지게 된다.
- ✓ $[\mathrm{lo}, \mathrm{hi}]$ 범위에서 이분 탐색 시, $(\mathrm{lo} + \mathrm{hi})/2$가 lo와 같은 값을 가지게 되어 무한 루프에 빠지게 된다.
- ✓ 따라서, 이분 탐색 시 C++의 `double`과 같은 double-precision 이상의 실수 자료형을 사용해야 된다.

- ✓ 예를 들어 $N = 2, x_1 = 0, x_2 = 100\,000\,000, q_1 = 1, q_2 = 1$을 생각하자.
  - 위의 예시와 같이 10^8 정도의 크기에서 `float`을 다루면, 오차가 1의 자리에서 발생하게 된다.
  - (`float`의 상대 오차는 약 10^{-7} 정도이고, `double`의 상대 오차는 약 10^{-15} 정도이다)
  - $\mathrm{lo} = 50\,000\,000, \mathrm{hi} = 50\,000\,004, (\mathrm{lo} + \mathrm{hi})/2 = 50\,000\,000$이 되어 무한 루프에 빠진다.
  - 구현에 따라 자세한 값은 바뀔 수 있다.
- ✓ 또한, double-precision 이상의 실수 자료형을 사용하더라도 이분 탐색 시 사용하는 ε 값을 너무 작게 잡아도 위와 같은 현상이 발생할 수 있으니, 10^{-6} 정도의 적당한 ε 값을 통해 이분 탐색해야 한다.

2A. 쓱~빡! 빡~쓱!

`greedy`
출제진 의도 – **Easy**

- ✓ 제출 71번, 정답 8명 (정답률 11.268%)
- ✓ 처음 푼 사람: **원도연**, 22분
- ✓ 출제자: `yunny_world`

- i번째 공격과 회피 시의 체내 아드레날린 변화량은 $\lfloor A_i \cdot K_i \rfloor - B_i$ 또는 $A_i - \lfloor B_i \cdot K_i \rfloor$ 이다.
- 경기 종료 시 최대 체내 아드레날린 양을 구해야 하므로, 항상 둘 중 큰 값을 취하면 된다.
- $\lfloor A_i \cdot K_i \rfloor - B_i \geq A_i - \lfloor B_i \cdot K_i \rfloor$ 을 정리하면, $K_i \geq 1$ 이다.
- i번째 공격과 회피 시에, $K_i \geq 1$ 이면 답에 $\lfloor A_i \cdot K_i \rfloor - B_i$ 를 더해주고, 아니면 답에 $A_i - \lfloor B_i \cdot K_i \rfloor$ 를 더하면 된다.

2B. W키가 빠진 성원이

`dfs`, `bfs`

출제진 의도 – **Easy**

- ✓ 제출 32번, 정답 17명 (정답률 53.125%)
- ✓ 처음 푼 사람: **심동현**, 16분
- ✓ 출제자: `dhlee1031`

2B. W키가 빠진 성원이

240

- 성원이가 “시작점에서 출발하여 W키를 제외한 나머지 7개의 키만을 이용해 목적지에 도달할 수 있다”는 것은, “목적지에서 출발해 X키를 제외한 나머지 7개의 키만을 이용해 시작점으로 갈 수 있다”는 것과 동일합니다.
- 이를 이용하면, 역으로 목적지에서 출발해 X키를 제외한 나머지 7개의 키만을 이용해 갈 수 있는 빈 공간의 개수를 세면 됩니다.
- 이는 기본 DFS 또는 BFS로 구현할 수 있습니다.

240

2C. Minimax Tree

`dp_tree`

출제진 의도 – **Medium**

- 제출 20번, 정답 15명 (정답률 75.000%)
- 처음 푼 사람: **김명준**, 49분
- 출제자: `2093ab`

- 지문에 나와있는 대로 Minimax Tree를 구성하면 되는 문제입니다.
- $\mathrm{dp}[i]$를 노드 i에서의 값이라고 정의합니다.
- 만약 리프 노드인 경우 노드의 값은 입력에서 주어진 값으로 계산합니다.
- 리프 노드가 아닌 경우 루트 노드부터 MAX player로 시작하고 자손 노드로 내려갈 때마다 MAX player와 MIN player가 바뀝니다.

- 정점 i의 (직접) 자손들의 집합을 C_i라 하면,

$$\mathrm{dp}[i] = \begin{cases} \max\limits_{d \in C_i} \mathrm{dp}[d] & \text{노드 } i\text{가 MAX player인 경우} \\ \min\limits_{d \in C_i} \mathrm{dp}[d] & \text{노드 } i\text{가 MIN player인 경우} \end{cases}$$

  와 같이 계산할 수 있습니다.
- 쿼리가 들어올 때마다 저장된 노드의 값을 출력해주면 됩니다.
- 이때 각 노드의 값을 매번 계산하는 경우 시간 초과를 받게 됩니다.

240

2D. 도로 위의 표지판

MST

출제진 의도 – **Medium**

- ✓ 제출 13번, 정답 9명 (정답률 69.231%)
- ✓ 처음 푼 사람: **강진주**, 43분
- ✓ 출제자: `dhlee1031`

2D. 도로 위의 표지판

240

- 문제의 핵심은 “같은 도로를 두 번 이상 지날 때는 처음 한 번만 돈을 지불해도 된다.”와 “지난 적이 있는 도로의 표지판 숫자는 핸드폰에 다시 적지 않기로 했다.”입니다.
- 이 조건이 있기 때문에 주어진 그래프에서 최소 스패닝 트리(Minimum Spanning Tree, MST)를 만들고 그 비용을 구하면 정답을 구할 수 있습니다.
- 그래프에서 간선의 가중치를 {표지판 숫자, 통행료}인 pair로 만들고, Prim 알고리즘 또는 Kruskal 알고리즘을 이용하여 MST를 구한 뒤 간선의 가중치의 합을 출력하면 됩니다.

2E. 피보나치 서로소

`number theory`
출제진 의도 – **Hard**

- ✓ 제출 23번, 정답 8명 (정답률 34.783%)
- ✓ 처음 푼 사람: **박건휘**, 18분
- ✓ 출제자: `dlguq0107`

2E. 피보나치 서로소

240

- 우선 n의 범위가 10억이기 때문에, 모든 수를 구하고 비교하는 방식으로는 서로소의 개수를 알기 힘듭니다.
- 중급 2회차 정수론 시간에 보았던 피보나치 수의 성질을 다시 한 번 알아봅시다.

- ✓ 우선 인접한 피보나치 수는 서로소입니다.
  - 귀류법으로 인접한 피보나치 수가 서로소가 아니라고 한번 가정해봅시다.
  - 그렇다면 두 수의 최대공약수 $g = \gcd(F_i, F_{i+1}), g > 1$ 가 존재해야 합니다.
  - $F_{i+1} = F_i + F_{i-1}$ 이고 $g | F_i, F_{i+1}$ 이므로 $g \mid F_{i-1}$ 을 만족해야 합니다.
  - 같은 논리를 F_{i-1}, F_i 에서도 적용이 가능합니다.
  - 위 논리를 계속하여 적용하면 $g \mid F_1, F_2$ 이어야 하는데 $F_1 = F_2 = 1$ 이므로 모순입니다.

240

- ✓ $F_{m+n} = F_{m-1}F_n + F_mF_{n+1}$ 가 성립한다는 것을 캠프에서 배웠습니다.
  - 간단하게 보여봅시다.
  - $m = 2$ 일 때, $F_{n+2} = F_n + F_{n+1} = F_1F_n + F_2F_{n+1}$ 이므로 성립합니다.
  - $m \leq k$ 일 때 성립한다 가정하고 $m = k + 1$ 일 때를 확인해봅시다.

$$\begin{aligned} F_{n+k+1} &= F_{n+k} + F_{n+k-1} \\ &= F_{k-1}F_n + F_kF_{n+1} + F_{k-2}F_n + F_{k-1}F_{n+1} \\ &= (F_{k-2} + F_{k-1})F_n + (F_{k-1} + F_k)F_{n+1} \\ &= F_kF_n + F_{k+1}F_{n+1} \end{aligned}$$

  - 귀납적으로 식이 성립함을 알 수 있습니다.

- ✓ 다음으로 $n \mid m \Rightarrow F_n \mid F_m$ 가 성립합니다.
  - $F_{m+n} = F_{m-1}F_n + F_mF_{n+1}$ 이 식을 이용하여 간단히 증명이 가능합니다.
  - 우선 $m = n$ 일 때는 자명합니다.
  - $m = n(k-1)\,(k > 1)$ 일 때 성립한다고 가정하고. $m = nk$ 일 때를 살펴봅시다.
  - $F_{nk} = F_{n-1}F_{n(k-1)} + F_nF_{n(k-1)+1}$ 인데 $F_n \mid F_{n-1}F_{n(k-1)}$ 이고 $F_n \mid F_nF_{n(k-1)+1}$ 이므로 $F_n \mid F_{nk}$ 가 성립합니다.
  - 귀납적으로 $n \mid m \Rightarrow F_n \mid F_m$ 임을 알 수 있습니다.

- ✓ $\gcd(F_n, F_m) = \gcd(F_n, F_{m-n}), (m > n)$
  - 이 사실도 위 유도과정과 비슷하게 증명이 가능합니다.
  - $g = \gcd(F_n, F_m)$ 라 둡시다.
  - $F_m = F_{n+(m-n)} = F_{n-1}F_{m-n} + F_nF_{m-n+1}$
  - $g \mid F_m = F_{n+(m-n)} = F_{n-1}F_{m-n} + F_nF_{m-n+1}$ 인데 $g \mid F_n$ 이고 $g \nmid F_{n-1}$ 이므로 $g \mid F_{m-n}$ 가 성립해야 합니다.
- ✓ 위 사실들을 통해 피보나치 수의 인덱스에 유클리드 호제법을 적용할 수 있다는 사실을 알 수 있습니다.
- ✓ 따라서 $\gcd(F_n, F_m) = F_{\gcd(n,m)}$ 이 성립합니다.

- ✓ 두 피보나치 수가 서로소인 경우는 $\gcd(F_n, F_m) = 1$ 이고 $F_1 = F_2 = 1$ 이므로
- ✓ 주어진 숫자를 n 이라고 할 때 $\gcd(i, n) = 1$ 또는 $\gcd(i, n) = 2, (i < n)$ 인 i 를 찾으면 됩니다.
- ✓ $\gcd(i, n) = 1$ 인 경우는 $\Phi(n)$ 를 통해 쉽게 구할 수 있습니다.
- ✓ $\gcd(i, n) = 2$ 인 경우는 잘 생각해보면 $2 \mid n$ 일 때, $\frac{n}{2}$ 이하의 수 중 서로소인 개수와 같다는 것을 알 수 있고 이는 $\Phi\left(\frac{n}{2}\right)$ 와 같습니다.
- ✓ 따라서 두 수를 더해주면 서로소인 쌍의 개수를 구할 수 있고 전체 시간 복잡도는 $\mathcal{O}(Q\sqrt{N})$ 입니다.
- ✓ $n = 1, n = 2$ 인 경우는 예외적으로 처리를 해 줘야 합니다.

2F. 편광판

`segment tree`
출제진 의도 – **Hard**

- ✓ 제출 12번, 정답 2명 (정답률 16.667%)
- ✓ 처음 푼 사람: **박건휘**, 141분
- ✓ 출제자: `starwh03`

- ✓ 우선 인접한 편광판이 $90°$ 차이가 나는 경우 빛이 통과하지 않음을 쉽게 알 수 있습니다.
- ✓ 범위가 주어졌을 때 그냥 구하게 된다면 시간이 $\mathcal{O}(N)$ 만큼 걸리고 쿼리가 Q 개 존재하므로 총 $\mathcal{O}(QN)$ 이 되어 시간 안에 해결할 수 없습니다.
- ✓ 범위가 주어졌을 때 어떻게 하면 통과할 수 있는지 없는지를 빠르게 판별할 수 있을까요?
- ✓ 우리는 캠프에서 배웠던 세그먼트 트리를 이용하여 쿼리 하나를 $\mathcal{O}(\log N)$ 으로 줄일 수 있습니다.

2F. 편광판

240

- 인접한 두 개의 편광판을 새로운 하나의 노드로 봅시다. 두 편광판이 90° 차이가 난다면 1, 아니면 0 으로 둘 수 있습니다.
- 최댓값 세그먼트 트리를 이용한다면 어떤 범위에서 1 이 있는지 없는지를 쉽게 판별할 수 있고 만약 구간의 값이 1 이라면 빛이 통과하지 못한다는 사실도 알 수 있습니다.
- 그럼 업데이트는 어떨까요? 하나의 편광판을 바꾸게 된다면 최대 인접한 2 개의 노드에 영향을 주기 때문에 마찬가지로 $\mathcal{O}(\log N)$ 시간에 처리가 가능합니다.
- 이를 통해서 시간복잡도 $\mathcal{O}(N + Q \log N)$ 에 해결 가능합니다.

240

2G. 나비야 나비야

Geometry, Sweeping
출제진 의도 – **Challenging**

- 제출 1번, 정답 0명 (정답률 0%)
- 처음 푼 사람: -, -분
- 출제자: lem0nad3

- ✓ 어떤 네 점을 뽑는 경우, 이 점들은 볼록 사각형 또는 오목 사각형을 이룹니다.
- ✓ 볼록 사각형의 경우, 나비를 2마리 그릴 수 있습니다.
- ✓ 오목 사각형의 경우는 나비를 그릴 수 없습니다.

- ✓ 어떤 네 점을 순서에 상관없이 뽑았을 때, 이 점들이 볼록 사각형을 이루는 경우의 수를 X 라고 하면 답은 $2X$ 입니다.
- ✓ 어떤 네 점을 순서에 상관없이 뽑았을 때, 이 점들이 오목 사각형을 이루는 경우의 수를 Y 라고 합시다.
- ✓ $X + Y = \binom{N}{4}$ 입니다.
- ✓ 식을 하나 더 구하여 연립방정식을 해결합시다. 여기에서는 $4X + 3Y$ 를 구하는 방법을 소개하겠습니다.

- ✓ $1 \leq i \leq N$ 인 모든 i 에 대해 다음을 시행하면 됩니다.
  1. i 번 점을 점 A 로 잡고, 나머지 점들을 점 A 를 중심으로 각도 정렬을 합니다.
  2. 또 다른 점 B 에 대하여, 점 A, 점 B, 점 C 의 ccw 값이 음수가 되도록 하는 점 C 의 개수를 셉니다.
- ✓ 네 점이 볼록 사각형을 이루는 경우, 동일한 점 집합에서 이는 4 번 더해집니다.
- ✓ 네 점이 오목 사각형을 이루는 경우는 3 번 더해집니다.
- ✓ 앞서 점을 정렬했기 때문에, 투 포인터로 점 C 의 개수를 관리하며 답을 갱신할 수 있습니다.
- ✓ 즉, 전체 시간복잡도는 $\mathcal{O}(N^2 \log N)$ 입니다.