# 2020 서강 프로그래밍 대회 풀이

Official Solutions

by

2020 SPC 출제진



2020년 11월 28일

| 문제 | | 의도한 난이도 | 출제자 |
|---|---|---|---|
| **mA** | 3대 측정 | **Easy** | 임지환raararaara |
| **mB** | 서강근육맨 | **Easy** | 안향빈mic1021 |
| **mC** | 반전 요세푸스 | **Medium** | 이기현sbrus_1213 |
| **mD** | 민트 초코 | **Medium** | 이상원gumgood |
| **mE** | 할로윈의 양아치 | **Hard** | 이기현sbrus_1213 |
| **mF** | 비밀번호 제작 | **Hard** | 안향빈mic1021 |
| **mG** | 피보나치와 수열과 쿼리 | **Challenging** | 임지환raararaara |
| **mH** | 블랙홀 | **Challenging** | 이상원gumgood |

# mA. 3대 측정

`implementation`
출제진 의도 – **Easy**

- ✓ 제출 61번, 정답 27명 (정답률 44.26%)
- ✓ 처음 푼 사람 : **강효규**, 3분
- ✓ 출제자 : 임지환`raararaara`

## **mA**. 3대 측정

- ✓ 세 팀원 모두에 대해 레이팅이 $L$ 이상인지 확인합니다.
- ✓ 세 팀원의 레이팅의 합이 $K$ 이상인지 확인합니다.
- ✓ 입력받은 순서대로 출력함에 유의합니다.

# mB. 서강근육맨

`greedy`, `binary_search`
출제진 의도 – **Easy**

- ✓ 제출 18번, 정답 18명 (정답률 32.76%)
- ✓ 처음 푼 사람 : **이민희**, 11분
- ✓ 출제자 : 안향빈`mic1021`

$N$ 개의 수를 두 개씩 짝지어서 각 그룹의 합의 최대값이 최소가 되게 하는 문제입니다. 문제를 단순화시켜서, 답이 $m$ 이 되도록 $2$ 개씩 짝지을 수 있는지를 확인해봅시다.

✓ $N$ 개의 수 중에 임의로 하나를 골라서 $a$ 라고 합니다. $a$ 와 짝이 되는 수를 $b$ 라고 하면, $b$ 는 $m - a$ 보다 작거나 같은 수 중에서 가장 큰 수입니다.

이렇게 $b$를 확정할 수 있는 근거는 다음과 같습니다.

- ✓ $m-a$보다 작거나 같은 수 중에서 가장 큰 수를 $b$라고 하고, 그보다 작은 임의의 수를 $c$라고 합시다.
- ✓ 그리고 $a$와 $b$를 짝지었을 때 아직 선택되지 않은 수들의 집합을 $S$라 하고, $a$와 $c$를 짝지었을 때 아직 선택되지 않은 수들의 집합을 $T$라고 합시다.
- ✓ 그러면 $S$와 $T$는 $N-2$개의 원소 중 $N-3$개가 일치하고, 나머지 한 원소에 대해 $c \in S$이고 $b \in T$입니다.
- ✓ 이 상황에서 두 개씩 짝지어서 각 그룹의 합의 최대값이 최소가 되도록 했을 때, $S$에서 구한 최소는 $T$에서 구한 최소보다 작거나 같습니다.

- 왜냐하면, $T$에서 짝지은 그룹에서 $b$를 $c$로 바꾸면 $c < b$이므로 $T$에서 구한 최소보다 작거나 같아지기 때문입니다. 따라서 $a$와 $b$를 짝짓는 것이 최선의 방법임을 알 수 있습니다.
- $m$은 $0$과 $10^{18}$ 사이에서 이분 탐색으로 최솟값을 구할 수 있습니다. 우선 $N$개의 원소를 오름차순으로 정렬한 다음, 특정 $m$에 대해 이분 탐색을 $N/2$번 수행해야 하므로, 전체 시간 복잡도는 $\mathcal{O}\left(N \log_2 N + N \log_2 N \log_2 10^{18}\right) = \mathcal{O}\left(N \log N\right)$이 됩니다.

# mC. 반전 요세푸스

`data_structures`, `deque`, `implementation`

출제진 의도 – **Medium**

- 제출 57번, 정답 18명 (정답률 31.58%)
- 처음 푼 사람 : **권지은**, 19분
- 출제자 : 이기현 `sbrus_1213`

- ✓ Deque을 이용한 구현 문제입니다.
- ✓ Queue를 이용하는 요세푸스 문제 (BOJ 1158) 를 참고하면 좋습니다.

- 문제에서 요구한 동작은 다음과 같이 진행할 수 있습니다.
  - **원을 따라 이동** — deque의 한쪽 끝에서 pop을 하고 pop 된 수를 반대 끝에 push를 해줍니다.
  - $K$ **번째 사람을 제거** — deque의 한쪽 끝에서 pop만 하고 push는 해주지 않습니다.
  - $M$ **명의 사람이 제거될 때마다 방향을 바꾸는 동작** — deque에서 pop & push를 하는 각 위치를 바꿔줍니다.

- 총 시간 복잡도는 $\mathcal{O}(NK)$ 입니다.

# mD. 민트 초코

`number_theory`
출제진 의도 – **Medium**

- 제출 182번, 정답 8명 (정답률 4.40%)
- 처음 푼 사람: **이민희**, 65분
- 출제자: 이상원`gumgood`

- ✓ 수식의 결과를 분수로 나타내봅시다.
- ✓ 분모가 분자로 나눠떨어지는지 확인해야 합니다.

결과에 부호는 영향을 주지 않기 때문에 모든 수를 양수로 만든 뒤 생각해 봅시다.

- ✓ 수식에 등장하는 수의 절대값은 $10$ 만을 넘지 않습니다. 따라서 모든 수를 $10$ 만 이하 소수의 곱으로 나타낼 수 있습니다.
- ✓ 각 소수에 대해 분모에 몇 번 곱해졌는지, 분자에 몇 번 곱해졌는지 관리합니다.
- ✓ 어떤 소수가 분자에 분모보다 더 많이 곱해진 경우, 수식의 결과는 정수가 아닌 유리수가 됩니다.
- ✓ 그렇지 않은 경우, 수식의 결과는 정수가 됩니다.

- ✓ 각 수를 $\mathcal{O}\left(\sqrt{X}\right)$ 안에 소인수분해하면, 총 시간복잡도는 $\mathcal{O}\left(N\sqrt{X}\right)$ 입니다.
- ✓ $0$을 곱하는 경우를 주의합시다.

# mE. 할로윈의 양아치

`disjoint_set`, `dp`
출제진 의도 – **Hard**

- ✓ 제출 32번, 정답 3명 (정답률 9.38%)
- ✓ 처음 푼 사람 : **강효규**, 66분
- ✓ 출제자 : 이기현 `sbrus_1213`

## mE. 할로윈의 양아치

- ✓ 각 아이의 사탕 수와 친구 관계가 주어졌을 때, 얻을 수 있는 최대 사탕을 구하는 문제입니다.

- ✓ 문제 해결을 위해서 두 단계로 나눌 수 있습니다.
  1. 친구 관계를 이용해 친구 그룹을 생성해야 합니다.
  2. 각 그룹의 인원수와 사탕 개수를 이용해 얻을 수 있는 최대 사탕의 개수를 구합니다.

1. 친구 관계를 이용해 친구 그룹을 생성해야 합니다.

- ✓ 지문의 “친구의 친구는 친구다?!” 라는 부분과 예제를 통해서 친구 관계가 있는 아이들을 그룹으로 묶어야 한다는 것을 알아야 합니다.
- ✓ Disjoint set을 이용하여 각 친구 관계에 대해서 union 연산으로 친구 그룹을 생성해줍니다.
- ✓ 그 방법 외에도 그래프 탐색을 이용해서 connected component를 구할 수 있습니다.

2. 각 그룹의 인원수와 사탕 개수를 이용해 얻을 수 있는 최대 사탕의 개수를 구합니다.

✓ 제한된 인원 내에서 최대 사탕을 뺏어야 하므로 Knapsack DP를 이용해 해결 가능합니다.

$dp[i][j] = i$번째 친구 그룹까지 확인해 최대 $j$명까지의 사탕을 뺏었을 때,
얻을 수 있는 최대 사탕의 개수

$c_i$를 $i$ 번째 그룹의 아이 수, $v_i$를 $i$ 번째 그룹의 사탕 수라고 할 때

$$dp[i][j] = \max_{j<i} \{dp[i-1][j], dp[i-1][j-c_i] + v_i\}$$

- ✓ disjoint set의 find 연산에 path compression 기법을 사용하지 않으면 TLE 가 날 수 있습니다.
- ✓ $K$ 명의 사탕을 뺏는 순간 공명을 하므로 최대 $K-1$ 명의 사탕을 뺏을 때의 최댓값이 답이 됨을 알아야합니다.

1. Disjoint set을 이용해 친구 그룹 생성 - $\mathcal{O}(N)$
2. Knapsack DP를 통해 최대 사탕 구하기 - $\mathcal{O}(NK)$

- ✓ 최종 시간복잡도는 $\mathcal{O}(N+NK)$ 가 됩니다.

# mF. 비밀번호 제작

`graph_traversal`, `bfs`

출제진 의도 – **Hard**

- 제출 6번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 안향빈`mic1021`

## mF. 비밀번호 제작

- 가능한 비밀번호는 $0$과 $N$ 사이의 정수입니다. 각각을 노드로 생각하여 그래프를 구성해 봅시다.
- 임의의 두 비밀번호를 XOR한 결과에서 $1$인 비트 갯수가 $1$개일 때 ($=$ 두 비밀번호의 안전 거리가 $1$일 때) 두 노드를 이어 줍니다.
- 그리고 로그인 시도에 사용된 비밀번호들을 전부 시작점으로 해서 다중 시작점multi-source BFS를 진행합니다.
- 정답은 가장 마지막에 방문한 노드까지의 최단 거리가 됩니다.

이해를 돕기 위한 부연 설명입니다.

- ✓ 모든 시작점으로부터의 최소 안전 거리가 $d$인 노드까지 BFS를 진행했다고 가정해 봅시다.
- ✓ 그러면 아직 방문하지 않은 정점은 어느 시작점으로부터도 안전 거리가 $d$보다 크다는 사실을 알 수 있습니다. 따라서 해당 정점의 안전도는 $d$보다 큽니다.
- ✓ 따라서 가장 마지막에 방문한 정점의 안전도가 최댓값입니다.

모든 $N$개의 정점에 대해 최대 $20$개의 인접한 정점을 탐색하므로, 시간 복잡도는 $\mathcal{O}(20N)$입니다.

# mG. 피보나치와 수열과 쿼리

`prefix_sum`, `lazyprop`
출제진 의도 – **Challenging**

- 제출 30번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 임지환 `raararaara`

✓ $[l, r]$ 이 업데이트될 경우

$a[l] = f_1, a[l+1] = f_2, ..., a[r] = f_{r-l+1}$ 이 됩니다. 피보나치 수열의 점화식에 따라, $a[k] = a[k-2] + a[k-1]$ 이 성립합니다. $[1, 5]$ 가 업데이트된다면 다음과 같습니다.

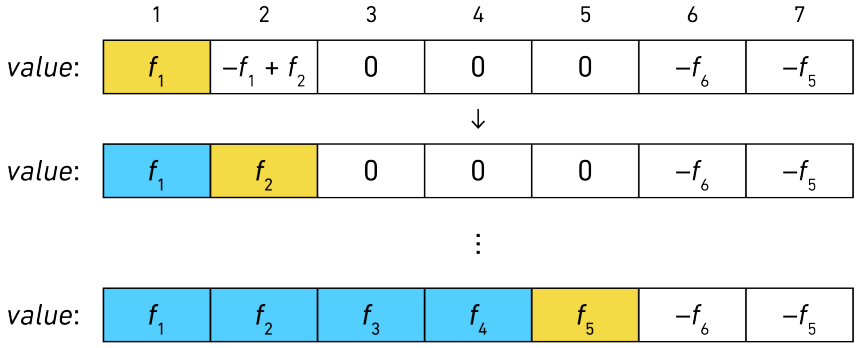
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| *value*: | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | 0 | 0 |

하지만 $a[k] = a[k-2] + a[k-1]$ 을 일반화하여 적용하면 $6$ 번째와 $7$ 번째에도 각각 $f_6, f_7$ 이 들어가게 됩니다. 구간 업데이트를 오프라인으로 처리하기 위해 아래와 같은 방식을 이용합니다.

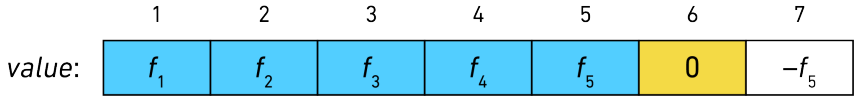
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| *value*: | | | | | | | |
| *add*: | $f_1$ | $-f_1 + f_2$ | | | | $-f_6$ | $-f_5$ |

## mG. 피보나치와 수열과 쿼리

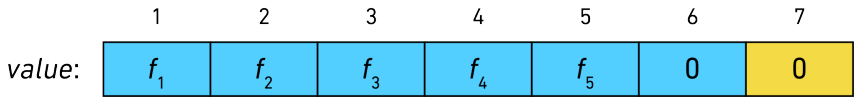
구간 $[l, r]$ 이 주어졌을 때, $l, l+1, r+1, r+2$ 위치에 알맞는 값을 대응시킨 후 $idx = l$ 부터 스위핑을 합니다. 이때 업데이트 규칙은 $a[i] \leftarrow a[i] + a[i-1] + a[i-2]$ 로 합니다.



$a[6] = a[6] + a[5] + a[4] = -f_6 + (f_4 + f_5) = -f_6 + f_6 = 0$ 이 됩니다.




$a[7] = a[7] + a[6] + a[5] = -f_5 + (f_5 + 0) = 0$ 이 됩니다.



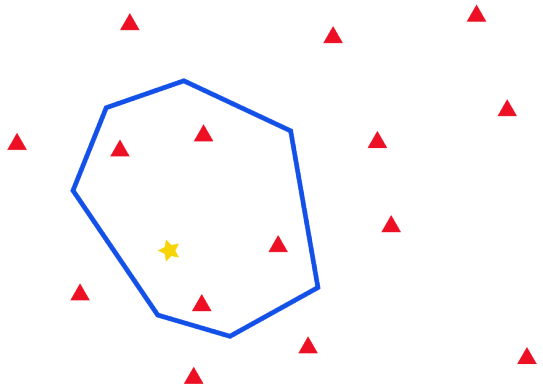
# mH. 블랙홀

`geometry`, `sorting`
출제진 의도 – **Challenging**

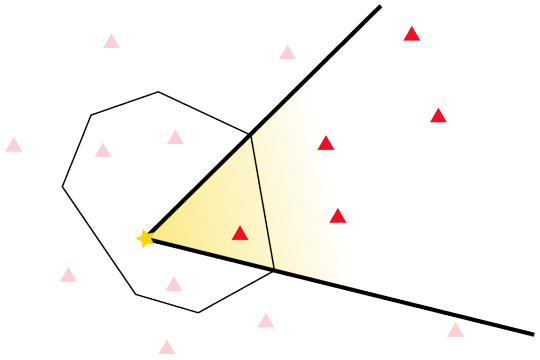
- 제출 1번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 이상원 `gumgood`

- ✓ 특정 $t$ 초에 블랙홀의 좌표를 직접 구하여 집어삼킨 건물의 수를 세는 경우를 생각해봅시다. $t$ 에 대해 parametric search 하여 해결할 수 있을 것 같습니다.

- ✓ 네 어림도 없습니다. 가능한 $t$ 의 범위는 $4 \cdot 10^{18}$ 이고, 이에 따라 블랙홀의 좌표는 대략 $8 \cdot 10^{27}$ 까지 갈 수 있습니다. 이는 일반적인 자료형에 담을 수 없습니다.
- ✓ 블랙홀의 좌표를 확대하지 않고 해결하는 방법이 필요해 보입니다.

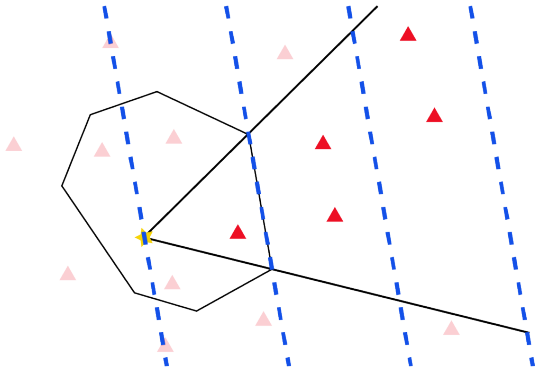


각 건물이 몇 초 뒤에 블랙홀에 집어삼켜지는지 알 수 있다면, $K$ 번째로 집어삼켜지는 건물의 집어삼켜진 시간은 쉽게 구할 수 있습니다.



평면을 특이점과 각 변을 기준으로 분할하여 생각해봅시다.

- ✓ 총 $N$ 개의 분할된 영역이 생깁니다.



각 영역에 속한 건물은 해당 변과 맞닿게 됩니다.

✓ 각 건물과 변 사이 거리를 계산하여 최소 몇 초가 지나야 집어삼켜지는지 구할 수 있습니다.

- 점과 직선 사이 거리를 실수 자료형으로 구하는 경우, 부동 소수점 오차를 피할 수 없습니다.
- 정수 좌표를 가지는 벡터들 간의 외적의 크기는 항상 정수임을 이용하여 부동 소수점 오차 없이 계산할 수 있습니다.
- 총 시간복잡도는 $\mathcal{O}\left(N + M \log M\right)$ 입니다.
- 블랙홀이 생긴 시점 $t = 0$에 이미 건물이 블랙홀에 집어삼켜진 경우에 주의합시다.

| 문제 | | 의도한 난이도 | 출제자 |
|---|---|---|---|
| **cA** | 파일 정리 | **Easy** | 이기현 sbrus_1213 |
| **cB** | 컨설팅 | **Easy** | 김주현 woonikim |
| **cC** | 연료가 부족해 | **Medium** | 안향빈 mic1021 |
| **cD** | 에어컨 설치 | **Medium** | 김주현 woonikim |
| **cE** | 사탕 배달 | **Hard** | 이윤제 yjyj1027 |
| **cF** | 폰친구 | **Hard** | 이윤제 yjyj1027 |
| **cG** | Confuzzle | **Challenging** | 임지환 raararaara |
| **cH** | 파인애플 피자 | **Challenging** | 이상원 gumgood |

# cA. 파일 정리

`sorting`
출제진 의도 – **Easy**

- ✓ 제출 28번, 정답 16명 (정답률 57.14%)
- ✓ 처음 푼 사람 : **박준성**, 3분
- ✓ 출제자 : 이기현`sbrus_1213`

## cA. 파일 정리

단순 정렬 문제입니다.

- ✓ 파일의 확장자를 해싱 또는 적절한 자료 구조 (`std::map` 등) 을 이용하여 각각 몇 번 등장하였는지 셉니다.
- ✓ 카운팅 후, $\{$확장자, 등장 횟수$\}$의 쌍을 확장자 이름을 기준으로 사전순으로 정렬을 진행해줍니다.
- ✓ 확장자의 개수가 최대 $5$만개이므로 $\mathcal{O}\left(N \log N\right)$ 정렬을 이용하여야 합니다.

# cB. 컨설팅

`greedy`
출제진 의도 – **Easy**

- ✓ 제출 35번, 정답 10명 (정답률 28.57%)
- ✓ 처음 푼 사람 : **고성빈**, 15분
- ✓ 출제자 : 김주현`woonikim`

같이 실행되면 안 되는 명령어들의 관계를 dependency라고 합시다.

- ✓ 아직까지 dependency가 발생하지 않은 명령어 집합 $X$ 를 생각해봅시다.
- ✓ 전체 명령어 집합 $C$에 대해, $X$ 와 $C - X$ 사이의 모든 dependency들을 최소한의 WAIT로 끊는 방법은 무엇일까요?

- ✓ $C - X$ 에 속해 있는 $i$ 번째 명령어 $c_i$ 가 $X$ 에 속해있는 명령어와 dependency가 생길 때, WAIT를 넣어주면 됩니다. ( $c_{i-1}$ 까지의 명령어는 $X$ 에 속해 있다고 가정합니다.)
- ✓ WAIT를 넣게 되면, $X$ 과 $C - X$ 사이의 모든 dependency가 끊기기 때문에, $X$ 를 초기화 시킨 후에 $c_i$ 를 $X$ 에 포함시켜야 합니다.
- ✓ 위 과정을 EXIT이 나올 때까지 반복하면 됩니다.
- ✓ 집합 $X$ 와 명령어 $c_i$ 사이에 dependency가 존재하는지 판단하는 것은 해싱이나 적절한 자료 구조 (`std::set`, `std::map` 등) 을 통해 해결할 수 있습니다.

# cC. 연료가 부족해

`dynamic_programming`, `binary_search`
출제진 의도 – **Medium**

- ✓ 제출 57번, 정답 5명 (정답률 8.77%)
- ✓ 처음 푼 사람 : **정성엽**, 39분
- ✓ 출제자 : 안향빈`mic1021`

출발점에서의 연료량이 $m$ 이라고 가정하면 피라미드까지 도달 가능한지의 여부는 다음과 같이 판단할 수 있습니다.

- ✓ 자동차가 오른쪽과 아래로만 이동할 수 있으므로, 우선 연료 보관소들을 $(r+c)$ 가 작은 순서로 오름차순 정렬합니다.

## cC. 연료가 부족해

$dp[i]$ 를 $i$ 번째 연료 보관소에 도착해서 보관된 연료를 충전했을 때, 현재 남은 자동차 연료량으로 정의합시다. 그러면 점화식을 다음과 같이 정의할 수 있습니다.

$$dp[i] = \max_{j<i} \left( dp[j] - dist\left(j, i\right) + fuel[i] \right)$$

- ✓ 여기서 $dist\left(j, i\right)$ 는 $j$ 번째 연료 보관소에서 $i$ 번째 연료 보관소까지의 맨해튼 거리이고, $fuel[i]$ 는 $i$ 번째 연료 보관소에 저장된 연료량입니다.
- ✓ $dp[j] \geq dist\left(j, i\right)$ 이어야 합니다. 이를 만족하는 $j$ 가 없을 경우, $i$ 번 연료 저장소에는 도달할 수 없게 됩니다.

## cC. 연료가 부족해

- $i = 1$ 부터 $N$ 까지 모든 연료 보관소를 차례대로 방문하면서 $dp$ 를 갱신했을 때, 피라미드를 방문할 수 없다면 출발점에서 $m$ 보다 많은 연료를 충전해야 한다는 뜻이 됩니다.
- 그렇지 않은 경우 $m$ 보다 적거나 같은 연료로도 피라미드에 도달할 수 있습니다.
- $m$ 하나에 대해 전체 $dp$ 배열은 $\mathcal{O}\left(N^2\right)$ 에 계산할 수 있으므로, $m$ 에 대한 이분 탐색으로 문제를 해결할 수 있습니다.

- ✓ 초기에 연료 보관소를 $(r+c)$ 순서로 정렬할 때 $\mathcal{O}\left(N \cdot \log_2 N\right)$,
- ✓ $N$ 개의 연료 보관소에 대해 $dp$ 점화식을 계산할 때 $\mathcal{O}\left(N^2\right)$,
- ✓ $m$ 에 대해 이분탐색을 진행하는 데 $m$ 의 상한값은 $R+C$ 이므로 $\mathcal{O}\left(\log_2\left(R+C\right)\right)$ 입니다.

  따라서 전체 시간복잡도는 $\mathcal{O}\left(N^2 \cdot \log_2\left(R+C\right)\right) \sim 10^6 \cdot \log_2 6\,000$ 입니다.

# cD. 에어컨 설치

`bipartite_matching`, `flow`
출제진 의도 – **Medium**

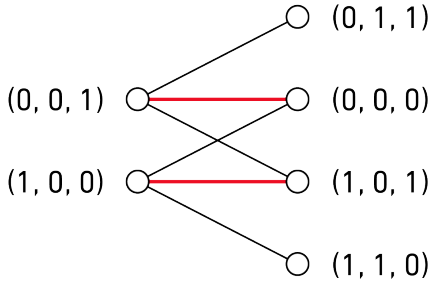
- ✓ 제출 11번, 정답 0명 (정답률 0.00%)
- ✓ 처음 푼 사람 : —
- ✓ 출제자 : 김주현`woonikim`

복도와 계단은 좌표 간 거리가 $1$인 두 방 사이에만 존재합니다.

- ✓ $(x_1, y_1, z_1)$에 있는 방과 $(x_2, y_2, z_2)$에 있는 방 사이의 거리가 $1$이라고 가정해봅시다. 이 경우, $|x_1| + |y_1| + |z_1|$와 $|x_2| + |y_2| + |z_2|$, 두 값의 차이는 항상 $1$입니다.

- ✓ 정수 좌표 $(x, y, z)$에 대해서 $(|x| + |y| + |z|) \equiv 0 \pmod 2$인 그룹을 $G_0$, $(|x| + |y| + |z|) \equiv 1 \pmod 2$인 그룹을 $G_1$라고 하면,
- ✓ 모든 간선 (복도나 계단) 은 $G_0$과 $G_1$ 사이에만 존재하고, 같은 그룹에 속해 있는 방 사이에는 존재하지 않는다는 것을 알 수 있습니다.

## cD. 에어컨 설치




- ✓ 따라서, 방을 정점으로 하고 간선을 복도와 계단으로 하는 그래프는 항상 **이분 그래프**가 된다는 것을 알 수 있습니다.
- ✓ 모든 복도/계단을 냉방하면 모든 방도 냉방이 되기 때문에, 이 문제는 이분 그래프의 모든 간선을 커버하는 **최소 버텍스 커버**의 크기를 구하는 것과 같습니다.

- ✓ Kőnig's Theorem에 의해, 이분 그래프에서의 최소 버텍스 커버의 크기는 최대 매칭의 크기와 동일합니다. 따라서, 이 문제는 이분 그래프의 **최대 매칭**의 크기를 구하는 것과 같습니다.
- ✓ 간선으로 연결되어있지 않은 방은 이분 매칭에 포함되지 않기 때문에, 별도로 처리해주어야 하는 것을 주의해 주세요.

# cE. 사탕 배달

`trees`, `lca`

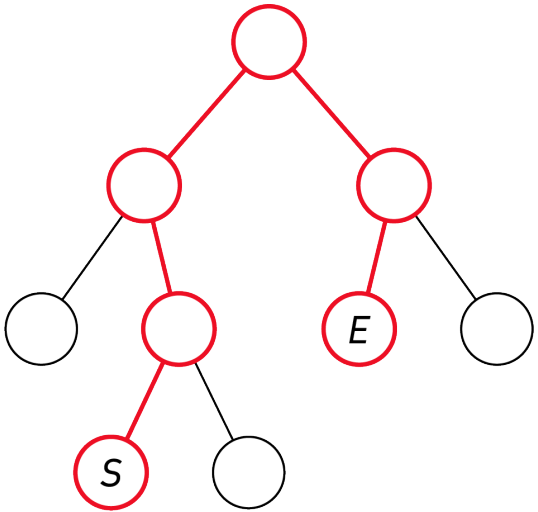
출제진 의도 – **Hard**

- 제출 24번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 이윤제 `yjyj1027`

친구들을 순서대로 찾아가는 길에 원하는 종류의 사탕을 사갈 수 있는지 확인해야 합니다.

- ✓ 처음에는 어디에서든 출발할 수 있으므로, 첫 번째 친구는 원하는 종류의 사탕 가게가 마을에 한 개 이상 존재하면 됩니다.
- ✓ 이후 $i$ 번째 친구에 대해 ($i-1$ 번째 친구의 위치) $\rightarrow$ ($i$ 번째 친구의 위치) 로 가는 경로상에 원하는 종류의 사탕을 파는 가게가 있어야 합니다.

## cE. 사탕 배달



- ✓ 사탕 종류가 최대 $5$ 가지이므로 각 사탕의 종류에 대해 존재하는지 여부를 배열로 관리할 수 있습니다.
  - 비트 연산을 활용하면 `int`형 배열에 최대 32종류의 사탕 정보를 담을 수 있습니다.
- ✓ 이는 쿼리당 $\mathcal{O}(\log N)$ 의 LCA 알고리즘을 이용해 해결할 수 있고,
- ✓ 이 때 시간 복잡도는 $\mathcal{O}(M \log N)$, 공간복잡도는 $\mathcal{O}(N \log N)$ 입니다.
- ✓ 같은 형태의 식을 heavy-light decomposition으로도 해결할 수 있습니다.

# cF. 폰친구

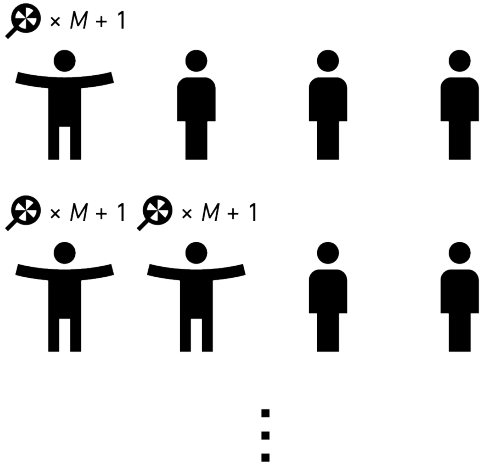
`combinatorics`, `inclusion_and_exclusion`
출제진 의도 – **Hard**

- ✓ 제출 7번, 정답 1명 (정답률 14.29%)
- ✓ 처음 푼 사람: **고성빈**, 89분
- ✓ 출제자: 이윤제`yjyj1027`

각 친구에게 $m$ 개 이상 $M$ 개 이하의 사탕을 나누어 주어야 합니다.

- ✓ 하한을 먼저 처리합니다. 우리는 모든 친구에게 $m$ 개를 나누어 주고 시작해도 무방합니다.
- ✓ 이제 $(K - Nm)$ 개의 사탕을 $N$ 명의 친구들에게 나누어 주는 문제가 되었습니다.

- ✓ 상한이 없다면 간단한 중복조합 문제입니다. 빼주어야 할 경우를 생각합시다.
  - 어떤 친구가 $M$ 개 초과의 사탕을 갖게 되는 경우입니다.


× M + 1
× M + 1
× M + 1

$$\sum_{i=0}^{K/(M+1)} (-1)^i \cdot ({}_N\mathrm{C}_i) \cdot \left({}_N\mathrm{H}_{K-i(M+1)}\right)$$

✓ 팩토리얼의 모듈로 역원을 이용해 조합 연산을 빠르게 계산할 수 있습니다.

# cG. Confuzzle

`case_work`, `sqrt_decomposition`
출제진 의도 – **Challenging**

- 제출 0번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 임지환 `raararaara`

✓ 서로 같은 값을 갖는 정점들의 집합에 대하여, 두 가지 방법을 생각해볼 수 있습니다.

1. 모든 정점 쌍에 대하여 일일이 거리 구하기
2. 각 정점들을 출발점으로 하는 multi-source BFS 수행

하지만 두 경우 모두 비효율적인 상황이 등장합니다.

1. 모든 정점 쌍에 대하여 일일이 거리 구하기
   $\rightarrow$ 트리 상의 모든 정점이 같은 값을 갖는다면 $\mathcal{O}\left(N^2 \log N\right)$ 의 시간이 걸립니다.
   두 정점 사이의 거리를 $\mathcal{O}\left(1\right)$ 에 구할 수 있다 하더라도 $\mathcal{O}\left(N^2\right)$ 입니다.

2. 각 정점들을 출발점으로 하는 multi-source BFS 수행
   $\rightarrow$ 모든 정점의 값이 모두 다르다면 BFS를 $N$ 번 수행하여, $\mathcal{O}\left(N^2\right)$ 의 시간이 걸립니다.

두 가지 방법을 합쳐봅시다.
같은 값을 갖는 정점의 개수에 따라 case work를 할 수 있습니다.

- ✓ 개수가 적은 경우 $\rightarrow$ 완전탐색
- ✓ 개수가 많은 경우 $\rightarrow$ multi-source BFS

많고 적음의 기준은 어떻게 할까요? $\rightarrow \sqrt{N}$!

- 개수가 $\sqrt{N}$ 보다 적은 경우
  $\sqrt{N}$ 개 이하의 정점들에 대한 완전탐색은 $\mathcal{O}(N \log N)$ 입니다.
  최악의 경우는 $\sqrt{N}$ 개의 정점 집합이 $\sqrt{N}$ 개 등장하는 경우로, $\mathcal{O}\left(N\sqrt{N}\log N\right)$ 입니다.
  두 정점 사이의 거리를 $\mathcal{O}(1)$ 에 구한다면 $\mathcal{O}\left(N\sqrt{N}\right)$ 입니다.

- 개수가 $\sqrt{N}$ 보다 많은 경우
  이 경우는 $\sqrt{N}$ 번 보다 많이 등장할 수 없습니다. 따라서 $\mathcal{O}\left(N\sqrt{N}\right)$ 입니다.

별해로, centroid decomposition을 사용할 수 있습니다.
이 경우 $\mathcal{O}\left(N \log N\right)$으로, 출제자 의도인 $\sqrt{N}$에 따른 case work보다 더 빠릅니다.(...)

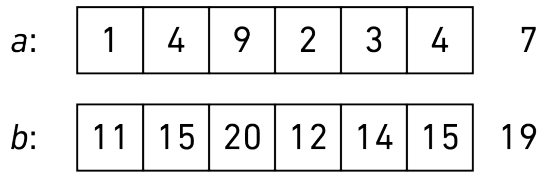
# cH. 파인애플 피자

`kmp`, `segment_tree`
출제진 의도 – **Challenging**

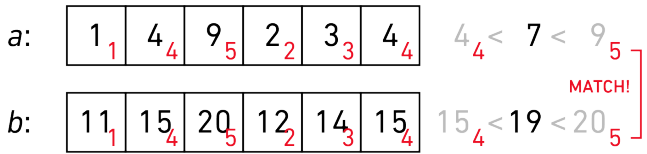
- 제출 3번, 정답 0명 (정답률 0.00%)
- 처음 푼 사람 : —
- 출제자 : 이상원 `gumgood`

Circular sequence가 주어졌을 때, 주어진 패턴이 등장하는 위치를 세는 문제입니다.

- ✓ 패턴과 정확히 일치하는 위치를 세는 경우, 수열을 두 번 이어 붙여 KMP 알고리즘으로 해결할 수 있습니다.
- ✓ 두 수열을 각각 좌표압축한 결과가 일치하는 경우, 두 수열이 **"비슷하다"**라고 합시다. 패턴과 비슷한 수열의 위치는 어떻게 찾아야 할까요?

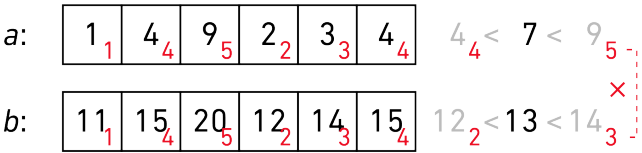



- 길이가 $6$인 두 비슷한 수열 $a, b$을 생각해 봅시다.
- 각 수열의 뒤에 $7$과 $19$를 추가하는 경우, 여전히 비슷한지 어떻게 확인할 수 있을까요?




각 원소 옆에 속한 수열에서의 등수를 표시했습니다.

- ✓ 수열 $a$에서, 추가하려는 수 $7$은 $4$등보다 크고 $5$등보다 작습니다.
  - 이를 통해 $7$을 추가하게 되면 $5$등이 된다는 사실을 알 수 있습니다.
- ✓ 수열 $b$에서도 마찬가지로 $19$를 추가하면 $5$등이 됩니다.
- ✓ 따라서 두 수열 뒤에 각각 원소 $7$과 $19$를 추가해도 두 수열은 여전히 비슷합니다.




각 수열 뒤에 $7, 13$를 추가하는 경우를 생각해봅시다.

- ✓ 수열 $a$에서 $7$은 앞에서와 같이 $5$등이 됩니다.
- ✓ 수열 $b$에서, 추가하려는 수 $13$은 $2$등보다 크고 $3$등 보다 작아 $3$등이 됩니다.
- ✓ 각 수열에 원소를 추가했을 때, 등수가 일치하지 않습니다. 따라서 두 수열 뒤에 각각 원소 $7$과 $13$을 붙이면 두 수열은 비슷하지 않게 됩니다.

이를 일반화하여 KMP 알고리즘을 수행할 수 있습니다.

- ✓ 두 원소가 같은지 확인하는 것 대신 두 원소가 두 비슷한 수열에 각각 추가 되었을 때 같은 등수를 가지는지 확인해야 합니다.
- ✓ 어떤 수에 대해 수열에서의 등수는 해당 수보다 큰 수의 개수, 작은 수의 개수로 구할 수 있습니다.
  - 이는 수열의 크기가 최대 $10$ 만이므로 좌표압축 후 segment tree 등과 같은 자료구조로 해결할 수 있습니다.
- ✓ 따라서 총 시간복잡도는 $\mathcal{O}\left((N+K)\log K\right)$ 입니다.

## 출제진

- 김주현 `woonikim` 서강대학교 컴퓨터공학과
- 안향빈 `mic1021` 서강대학교 컴퓨터공학과
- 이기현 `sbrus_1213` 서강대학교 수학과
- 이상원 `gumgood` 서강대학교 컴퓨터공학과
- 이윤제 `yjyj1027` 서강대학교 수학과
- 임지환 `raararaara` 서강대학교 컴퓨터공학과

## 검수진

| | | | |
|---|---|---|---|
| ✓ | 강한필 ho94949 | NEXON | KAIST 전산학부 |
| ✓ | 김영현 kipa00 | NEXON | 서울대학교 컴퓨터공학부 |
| ✓ | 나정휘 jhnah917 | | 선린인터넷고등학교 |
| ✓ | 류호석 rhs0266 | Lunit Inc. | 서울대학교 컴퓨터공학부 |
| ✓ | 박수현 shiftpsh | NEXON | 서강대학교 컴퓨터공학과 |
| ✓ | 이태한 sogangcse | | 서강대학교 컴퓨터공학과 |
| ✓ | 정기웅 QuqqU | | 서강대학교 컴퓨터공학과 |
| ✓ | 홍은기 pichulia | 삼성전자 | 고려대학교 컴퓨터학과 |