

ICPC Sinchon Summer Camp 2020 Final Exam

Official Solutions

Table of Contents

문제		의도한 난이도	출제자	
2A	요요 시뮬레이션	Easy	백성익홍익대	906bc
2B	IF 문 좀 대신 써줘	Easy	백성익홍익대	906bc
2C	1A 센티와 마법의 뽕망치	Easy	백성익홍익대	906bc
2D	화장실의 규칙	Medium	백성익홍익대	906bc
2E	중첩 집합 모델	Medium	이재열홍익대	malkoring
2F	TORIE	Hard	백성익홍익대	906bc
2G	좀비 떼가 전역 때보다 먼저 오다니	Hard	백성익홍익대	906bc
2H	좀비 떼가 기관총 진지에도 오다니	Medium	백성익홍익대	906bc

Table of Contents

문제	의도한 난이도	출제자
1A 2C 센티와 마법의 뽕망치	Easy	백성익 ^{홍익대} 906bc
1B 배틀로얄	Easy	이준석 ^{서강대} semteo04
1C 햄치 몇?	Medium	임지환 ^{서강대} raararaara
1D Random Generator	Medium	이국렬 ^{연세대} lky7674
1E Unique Solution	Medium	이국렬 ^{연세대} lky7674
1F 미하일 2마리	Medium	이국렬 ^{연세대} lky7674
1G 미담 전하기	Hard	임지환 ^{서강대} raararaara
1H 재미여행	Hard	이준석 ^{서강대} semteo04

1A/2C. 센티와 마법의 뽕망치

Priority queue

출제진 의도 : Easy

- 제출 18번, 맞은 사람 7명 (정답률 38.89%)
- First Solve : pjh6792^{서강대학교}
- 제출 35번, 맞은 사람 13명 (정답률 37.14%)
- First Solve : leeju1013^{홍익대학교}
- 출제자 : 906bc^{홍익대학교}

1A/2C. 센티와 마법의 뽕망치

- 사실 이 문제는 초급 스터디 1주차에 업로드했던 시간 복잡도 분석용 코드 중 하나를 살짝 변형한 문제입니다.
- 뽕망치를 사용할 때마다 정렬을 하여 가장 큰 거인을 찾는다면 $O(TN \log N)$ 이므로 시간 초과입니다.
- 뽕망치를 사용할 때마다 선형 탐색으로 가장 큰 거인을 찾는다면 $O(TN)$ 이므로 시간초과입니다.
- 문제를 해결하기 위해서는 거인들을 우선순위 큐에 넣은 뒤 가장 큰 거인을 뽑아서 때리고, 다시 집어넣으면 $O((T+N)\log N)$ 으로 해결할 수 있습니다.

1G. 미담 전하기

Strongly Connected Component, Topological Sort, Dynamic Programming

출제진 의도 : Hard

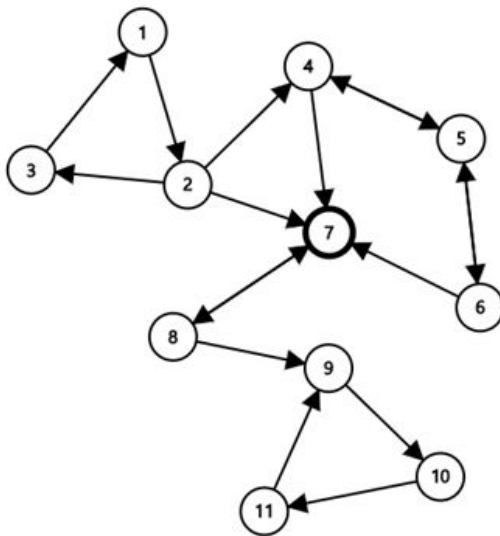
- 제출 20번, 맞은 사람 1명 (정답률 5.00%)
- First Solve : rhksdn6227^{서강대학교}
- 출제자 : raararaara^{서강대학교}

1G. 미담 전하기

- 방향 그래프가 주어졌고, 미담은 당사자에게 전달되어야 합니다.
- 역방향 간선들로 구성된 그래프를 만든다면, 미담 당사자로부터 출발하여 도달할 수 있는 사람들은 직접 미담 전파자의 후보가 될 수 있습니다.
- 특정 미담 전파 경로 상에 사이클이 존재한다면, 사이클 내의 모든 사람들은 모두 간접 미담 전파자가 될 수 있습니다. Strongly Connected된 관계를 떠올릴 수 있습니다.

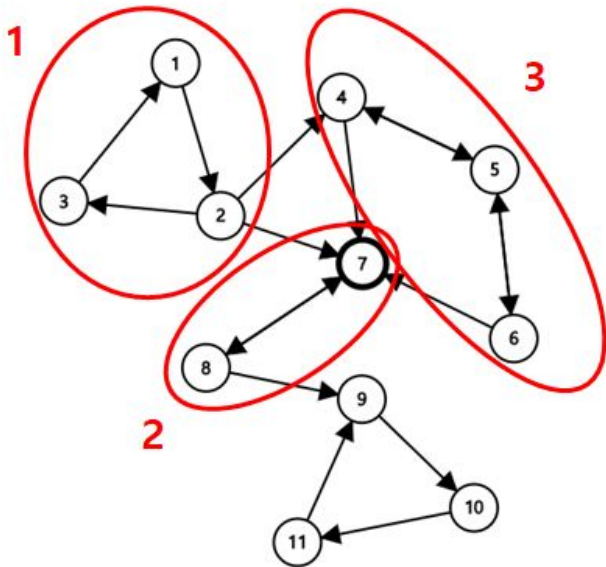
1G. 미담 전하기

- 입력 예시를 SCC단위로 묶어봅시다.



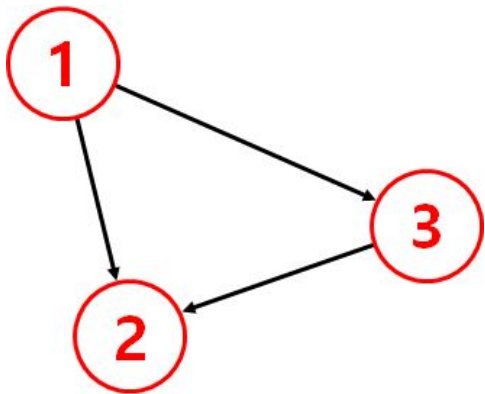
1G. 미담 전하기

- 묶은 SCC를 다음과 같이 라벨링하고, 필요 없는 부분을 제거해봅니다.



1G. 미담 전하기

- 목적지는 2번 라벨인데, 1번과 3번 컴포넌트의 개수에 상관없이 $2 \rightarrow 3 \rightarrow 1$ 순으로 방문하는 것이 유리합니다.
- 각 컴포넌트마다의 개수를 저장하고, 컴포넌트 단위로의 역간선 그래프를 만들어줍니다.

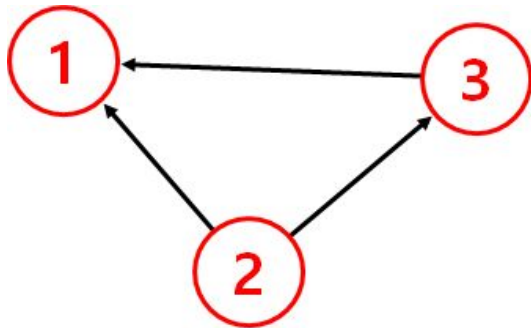


1G. 미담 전하기

- 정방향 간선 그래프에서 현재 컴포넌트를 cur , 다음 전파될 컴포넌트를 nxt 라 하고 $d[idx]$ 를 “ idx 번째 컴포넌트에서 미담 당사자가 있는 컴포넌트로 전파를 시도했을 때 생기는 간접 미담 전파자의 수”라 하면

$$d[nxt] = \max(d[nxt], d[cur] + scc_size[nxt])$$

가 성립합니다.



1G. 미담 전하기

$$d[nxt] = \max(d[nxt], d[cur] + scc_size[nxt])$$

- 역방향 간선 그래프로 구성을 하였기 때문에 위와 같은 관계식을 풀기 편합니다.
- 위상정렬을 이용하여 해결합니다.

1G. 미담 전하기

- 직접 미담 전파자는 간접 미담 전파자가 될 수 없습니다.
- 미담 당사자에 의해서도 간접 미담 전파자가 생길 수 있습니다. 즉, 미담 당사자가 포함된 사이클도 고려합니다.
- 미담 당사자는 직접 미담 전파자가 될 수 없음에 유의합니다.

2A. 요요 시뮬레이션

Simulation

출제진 의도 : Easy

- 제출 42번, 맞은 사람 13명 (정답률 30.95%)
- First Solve : yibin99^{숙명여자대학교}
- 출제자 : 906bc^{홍익대학교}

2A. 요요 시뮬레이션

- 제목에 적힌 것처럼 시뮬레이션을 하면 되는 문제입니다.
- 다이어트 에너지 섭취량 - 다이어트 에너지 소비량 $\leq$ 기초 대사량 변화 역치 인 경우, 요요가 발생하지 않습니다.
- 단, 힌트에서 언급한 것처럼 C++에서는 $-5 / 2$ 의 값이 -2 가 나오므로, 이를 별도 처리해줘야 합니다.

2F. TORIE

String, Parsing, Tree, DFS, Stack

출제진 의도 : Hard

- 제출 0번, 맞은 사람 0명 (정답률 0.00%)
- First Solve : —
- 출제자 : 906bc^{홍익대학교}

2F. TORIE

- 문자열을 파싱하여 트리를 만들고, 이 트리를 순회하면서 키워드를 검색하는 문제입니다.
- 첫 번째 줄에 주어지는 문자열의 길이를 L , 키워드 문자열의 길이를 K 라고 합시다.
- 트리를 만든 후 모든 TORIE의 영어 대문자 문자열의 각 글자에서 키워드 문자열 검색을 거꾸로 한다면 이 때의 시간 복잡도는 $O(LK)$ 입니다. 시간 제한을 충분히 통과하므로 여기서 추가적인 문자열 검색 알고리즘을 적용할 필요가 없습니다.
- 그 후 키워드가 탐색된 TORIE는 순회에서 제외하고 출력을 하면 됩니다.

2F. TORIE

- 파이썬으로 문제를 풀 경우 Python3로는 시간 초과가 발생하므로 PyPy3로 해결해야 하는데, PyPy3에서는 `setrecursionlimit`이 제대로 작동하지 않으므로 DFS를 스택으로 구현하여 해결해야 합니다.

2D. 화장실의 규칙

Priority queue

출제진 의도 : Medium

- 제출 25번, 맞은 사람 6명 (정답률 24.00%)
- First Solve : ystaeyoon113^{연세대학교}
- 출제자 : 906bc^{홍익대학교}

2D. 화장실의 규칙

- M명의 선두에서 화장실을 이용할 사람을 선형 탐색으로 찾는다면 총 $O(NM)$ 이므로 시간 초과입니다.
- M개의 줄을 우선순위 큐에 넣으면서 우선순위의 기준을 선두로 잡아 우선순위 큐에서 화장실을 이용할 사람을 찾는다면 $O((N+M) \log M)$ 으로 해결할 수 있습니다.

1H. 재미 여행

기하 + Dijkstra

출제진 의도 : Hard

- 제출 0번, 맞은 사람 0명 (정답률 0.00 %)
- First Solve : —
- 출제자 : semteo04^{서강대학교}

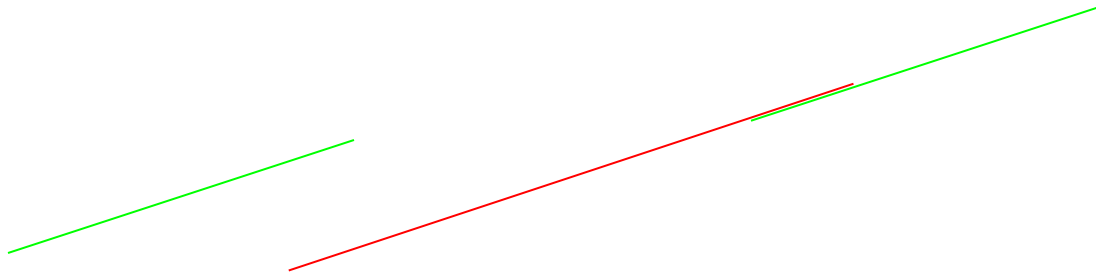
1H. 개미 여행

각 그룹별로 컨벡스 헐을 구합니다. 컨벡스 헐 위의 점과 시작점, 끝점으로 만들 수 있는 모든 선분에 대해서 이동할 수 있는지 판단합니다.

이 점들을 정점으로 하고 이동할 수 있는 정점들끼리 연결해 줍니다. 이때, 가중치는 선분의 길이입니다. 시작점부터 끝점까지 다익스트라를 돌리면 해결할 수 있습니다.

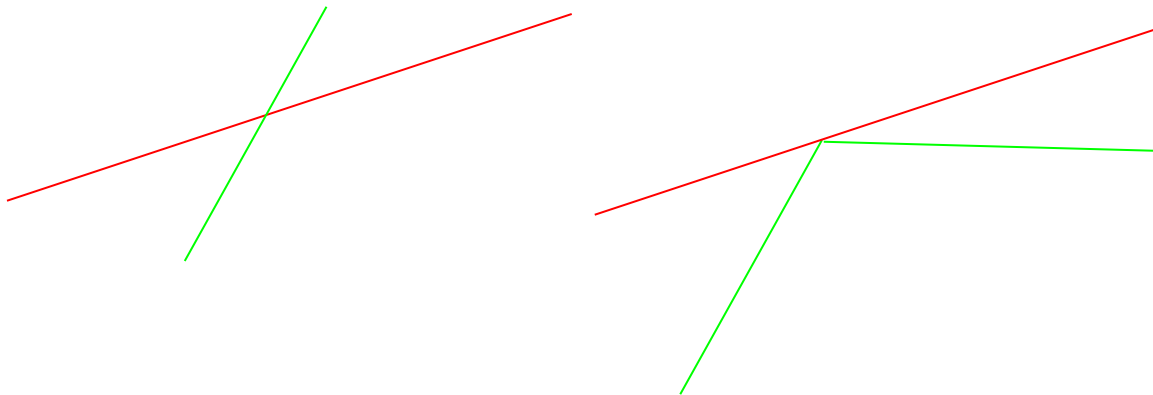
1H. 개미 여행

어떤 선분이 있을 때, 이 선분과 평행한 선분은 모두 통과할 수 있기 때문에 무시합니다.



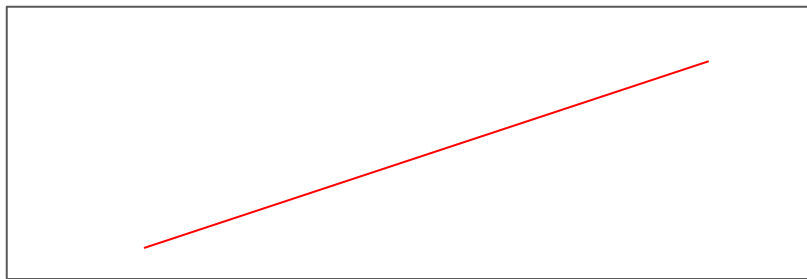
1H. 개미 여행

만약 이 선분과 양 끝이 아닌 다른 곳에서 교차한다면 이동할 수 없다고 판단합니다.



1H. 재미 여행

이 선분의 중점을 잡은 다음 이 점이 다른 컨벡스 헐 안에 포함되면 이동할 수 없다고 판단합니다. 모든 과정을 통과하면 이동할 수 있는 선분입니다.



1D. Random Generator

Segment Tree

출제진 의도 : Medium

- 제출 9번, 맞은 사람 5명 (정답률 55.55%)
- First Solve : pjh6792^{서강대학교}
- 출제자 : lky7674^{연세대학교}

1D. Random Generator

- 우선 segment tree에 각 $p[i]$ 들을 i 번째 노드들에 update 해준다.
- 예제 : $n=8$, $p=[10,9,8,7,1,2,3,4]$, $w=[19,23,28,8,9,9,3,2]$

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

1D. Random Generator

- k번째까지의 Prefix sum이 $w[i]$ 이상이 되는 최소 k를 구한다.
- 이후 k번째 원소를 0으로 초기화 해준다.

35							
25				10			
10		15		3		7	
10	0	8	7	1	2	3	4

1D. Random Generator

- 루트 노드에서 출발한다.

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

1D. Random Generator

- 루트 노드에서 출발한다.
 - 현재 노드의 왼쪽 노드의 값이 $w[i]$ 이상이라면 왼쪽으로 이동

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

1D. Random Generator

- 루트 노드에서 출발한다.
 - 현재 노드의 왼쪽 노드의 값이 $w[i]$ 이상이라면 왼쪽으로 이동

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

1D. Random Generator

- 루트 노드에서 출발한다.
 - 현재 노드의 왼쪽 노드의 값이 $w[i]$ 이상이라면 왼쪽으로 이동
 - 그렇지 않다면 오른쪽으로 이동하며 $w[i]$ 에 왼쪽 노드의 값을 빼준다.

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

1D. Random Generator

- 루트 노드에서 출발한다.
 - 현재 노드의 왼쪽 노드의 값이 $w[i]$ 이상이라면 왼쪽으로 이동
 - 그렇지 않다면 오른쪽으로 이동하며 $w[i]$ 에 왼쪽 노드의 값을 빼준다.
- 리프 노드에 도달하면 해당 위치를 return 한다.
- 비슷한 문제인 1849 순열, 1777 순열 복원을 풀어보자.

44							
34				10			
19		15		3		7	
10	9	8	7	1	2	3	4

2E. 중첩 집합 모델

DFS, DFS Numbering, Euler tour technique

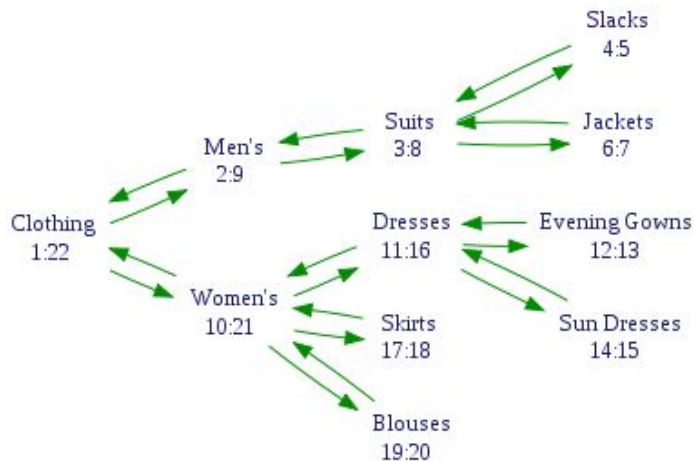
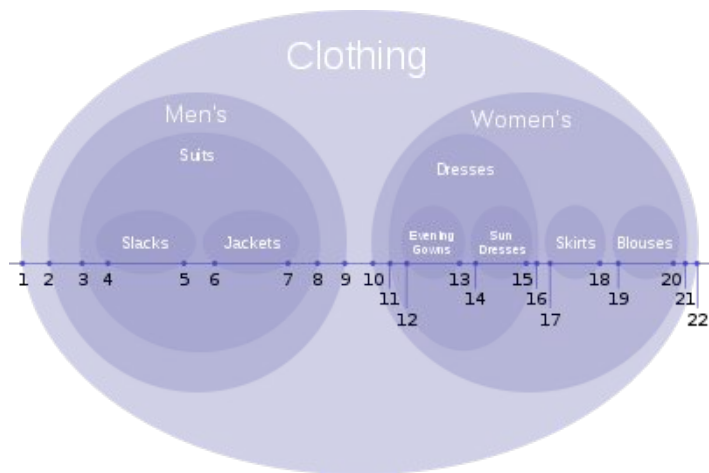
출제진 의도 : Medium

- 제출 5번, 맞은 사람 2명 (정답률 40.00%)
- First Solve : ystaeyoon113^{연세대학교}
- 출제자 : malkoring^{홍익대학교}

2E. 중첩 집합 모델

- https://en.wikipedia.org/wiki/Nested_set_model

개발 분야에서 쓰이긴 하는 알고리즘이라는 주제로 끼워넣고 싶었습니다.



2E. 중첩 집합 모델

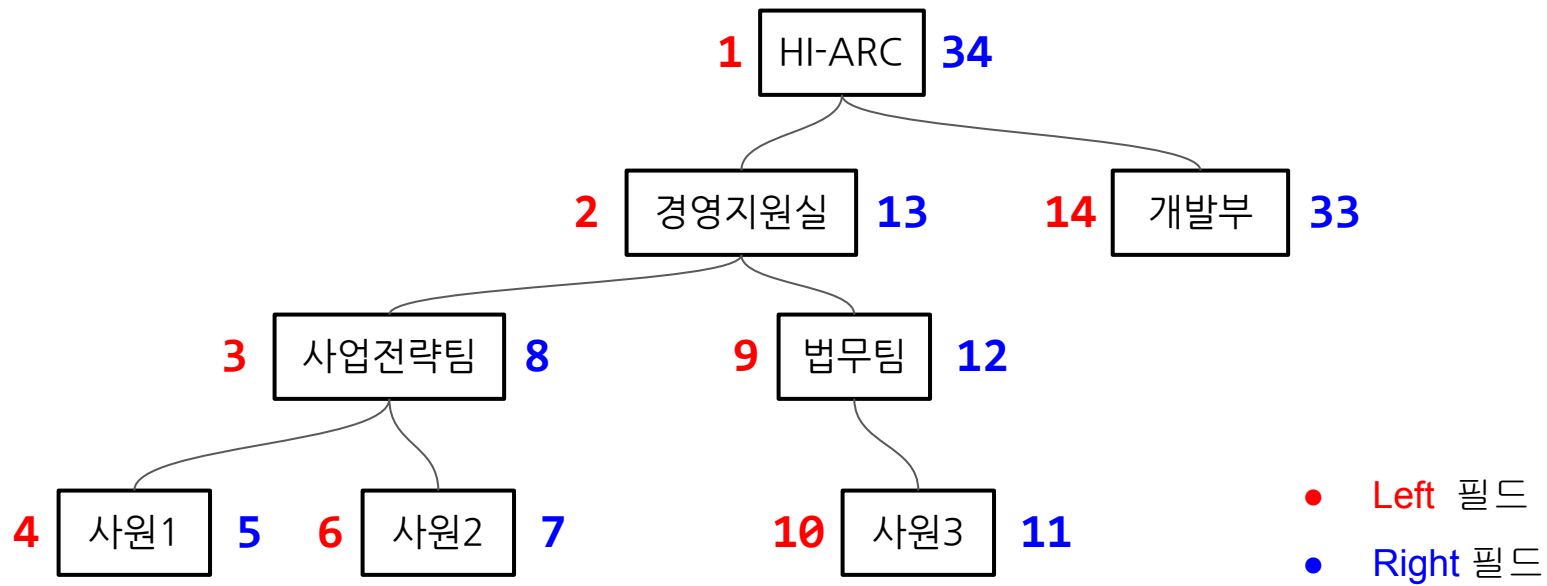
- 이 문제를 이해하려면,
엄청나게 긴 지문을 읽으려고 하기보다는
left/right 필드가 왜 저렇게 채워지는지를
이해하는 편이 좋습니다.

부서명/이름	left	right	level
HI-ARC	1	34	1
경영지원실	2	13	2
사업전략팀	3	8	3
사원1	4	5	4
사원2	6	7	4

- 다음 슬라이드에서 어떤 모양이 되는지 한 번 살펴보겠습니다.

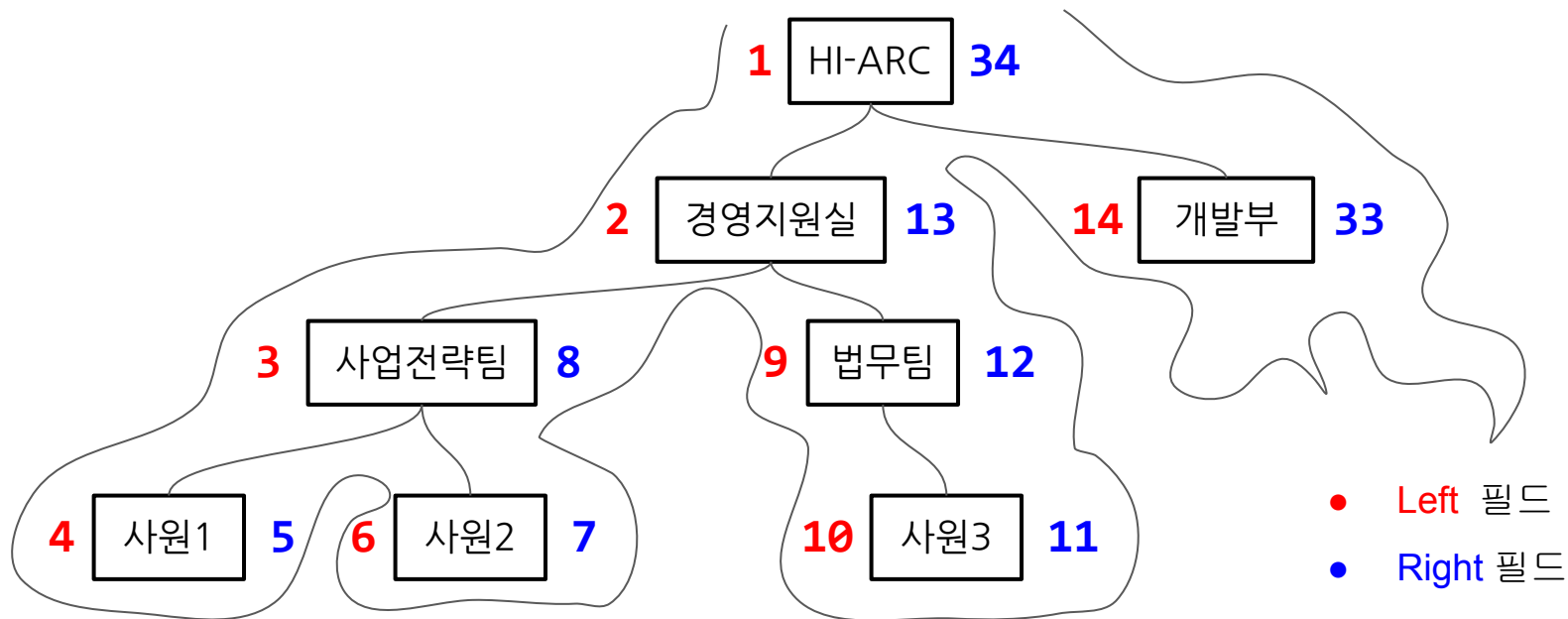
2E. 중첩 집합 모델

- 여백이 부족하기 때문에, 일부만 트리로 나타내겠습니다.



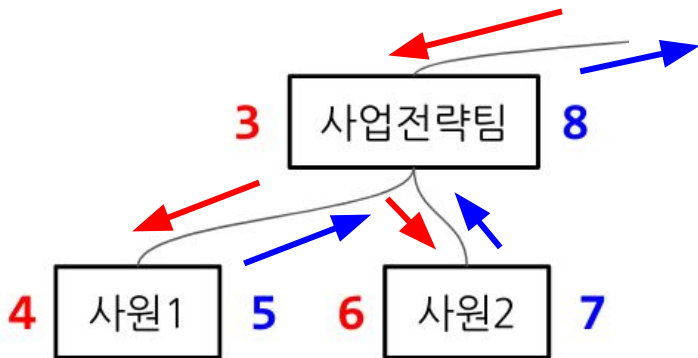
2E. 중첩 집합 모델

- 트리의 방문순서대로 left 필드와 right 필드가 어떻게 채워지는지 확인해봅시다.



2E. 중첩 집합 모델

- 루트 노드에서부터 DFS를 진행할 때,
노드에 진입하는 과정에서 1 증가해서 left 필드를 채우고
노드를 빠져나가는 과정에서 1 증가해서 right 필드를 채웁니다.



```
void dfs(int u) {  
    if (visited[u]) return;  
  
    visited[u] = true;  
    L[u] = cnt++;  
  
    for (auto v : graph[u]) {  
        if (!visited[v])  
            dfs(v);  
    }  
  
    R[u] = cnt++;  
}
```

2E. 중첩 집합 모델

- 트리의 방문 순서대로 번호를 매길 때,
번호가 가장 낮은 노드부터 오름차순으로 방문해야 하기 때문에
트리를 방문하기 전에 정점에 연결된 모든 간선을 오름차순으로 정렬해야 합니다.

```
for (int i = 1; i <= N; ++i) {  
    sort(graph[i].begin(), graph[i].end());  
}
```


2E. 중첩 집합 모델

- 위에서 사용된 방법은 최소 공통 조상(Least Common Ancestor)을 세그먼트 트리로 구현할 때 사용되는 테크닉과 유사한 방법이니 알아두는게 도움될 수 있습니다.

2B. IF문 좀 대신 써줘

Binary Search

출제진 의도 : Easy

- 제출 56번, 맞은 사람 9명 (정답률 16.07%)
- First Solve : ystaeyoon113^{연세대학교}
- 출제자 : 906bc^{홍익대학교}

2B. IF문 좀 대신 써줘

- 문제 본문의 pseudo code처럼 칭호를 선형 탐색한다면 $O(NM)$ 으로 시간 초과가 발생합니다.
- 칭호는 전투력 상한의 비내림차순으로 주어지므로 정렬 할 필요는 없으며, 주어지는 캐릭터의 전투력으로 이분 탐색(lower_bound)을 하면 $O(N \log M)$ 으로 해결할 수 있습니다.

1 C. 햄치 몇?

Dynamic Programming

출제진 의도 : Medium

- 제출 2번, 맞은 사람 0명 (정답률 0.00%)
- First Solve : —
- 출제자 : raararaara^{서강대학교}

1C. 햄치 몇?

- 대회에 참가한 관우와 길원이는 이 문제를 풀지 못했습니다.
- 전체 효용의 합을 3등분하되, 가장 작은 값이 최대가 되도록 해야 합니다.
- N 은 충분히 작고($1 \leq N \leq 50$), 햄버거 각각의 효용도 작습니다. ($1 \leq a_i \leq 50$)

1C. 햄치 몇?

- 0-1 Knapsack인데 두명이 아닌 세명이 나눠 가진다고 생각을 해봅니다.
- 0~idx까지의 햄버거를 모두 분배하였을 때 효용의 합을 T 라 하면

$$dp[idx][x_1][x_2] = T - (x_1 + x_2)$$

가 성립합니다. $x_1, x_2, T - (x_1 + x_2)$ 중 길원이는 가장 작은 값을 취할 수 있습니다.

- 효용의 합은 최대 2500이므로, 한명이 가져갈 수 있는 최대 효용은 $2500/3 = 833.33$ 입니다.
- $O(840^2N)$

1 B. 배틀로알

Greedy

출제진 의도 : Easy

- 제출 24번, 맞은 사람 7명 (정답률 29.17%)
- First Solve : wnstjd13245^{서강대학교}
- 출제자 : semteo04^{서강대학교}

1 B. 배틀로얄

만약 (회복할 수 있는 체력의 총 합 + 최대 체력 $\leq$ 싸움으로 잃게 되는 체력의 합)이면
게임에서 이길 수 없습니다.

어떤 순서로 싸우고 아이템을 먹어도 마지막에는 체력이 음수가 됩니다.

그렇지 않은 경우 항상 싸움에서 이길 수 있습니다.

1 B. 배틀로알

현재 체력 $> M/2$ 이면 적과 싸우고 현재 체력 $\leq M/2$ 면 회복 아이템을 먹습니다.

이렇게 하면 최대 체력을 넘어서 회복하지 않고 싸워서 질 일도 없습니다.

예외적으로 싸울 적이 없을 땐 회복 아이템을 먹어야 하고, 회복 아이템이 없을 땐 적과 싸워야 합니다.

1 E. Unique Solution

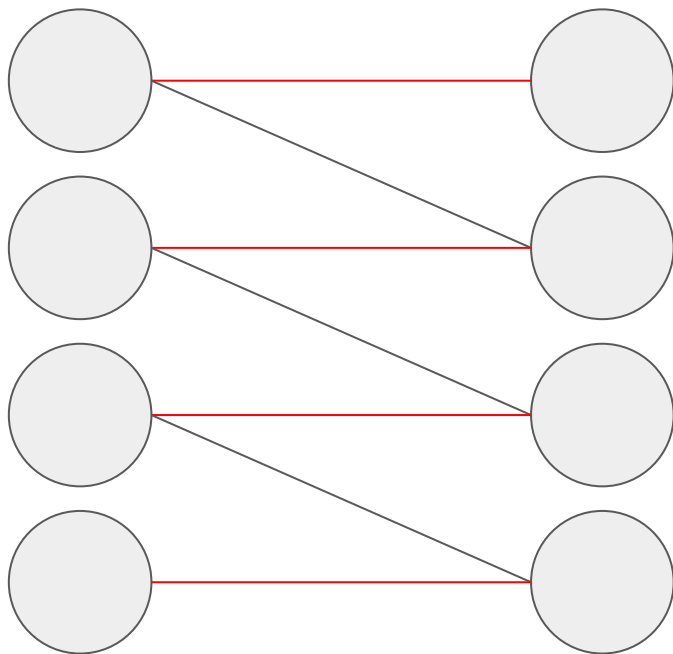
Bipartite Matching, Topological Sorting

출제진 의도 : Medium

- 제출 25번, 맞은 사람 3명 (정답률 12.00%)
- First Solve : wnstjd13245^{서강대학교}
- 출제자 : lky7674^{연세대학교}

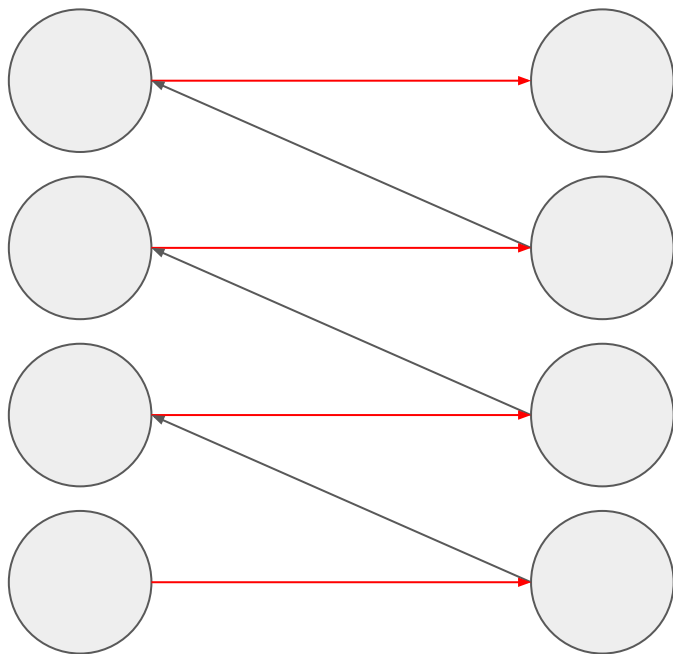
1E. Unique Solution

- Perfect Matching을 하나 찾는다.



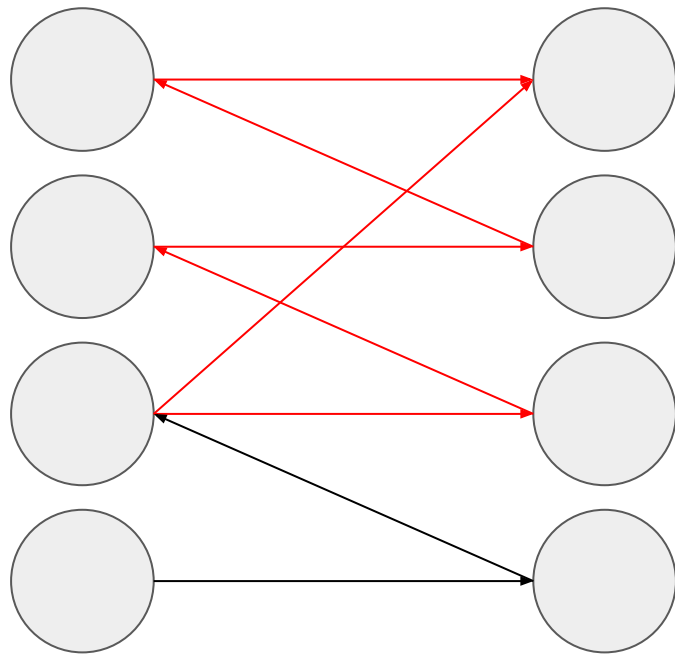
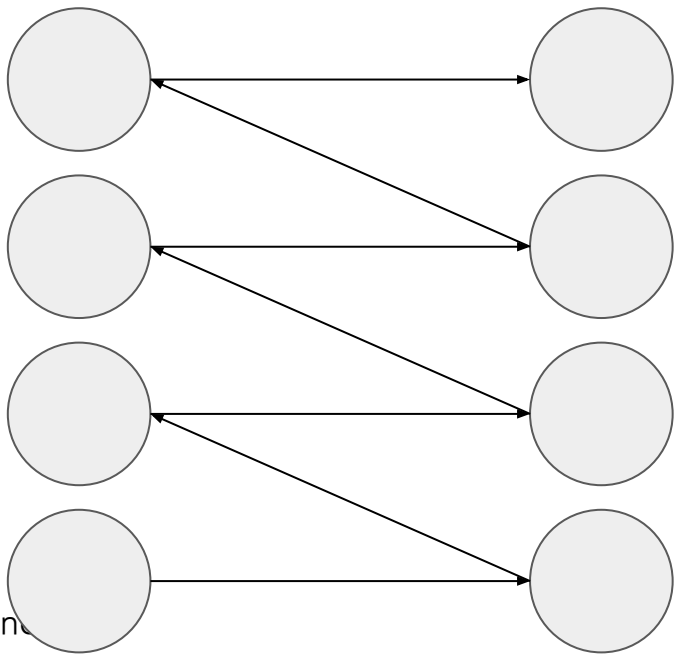
1E. Unique Solution

- Matching에 속하면 정방향, 아니면 역방향 간선으로 만들어준다.



1E. Unique Solution

- Cycle이 없으면 유일한 해가 있고, 있으면 다른 해가 존재한다.
- Cycle 유무는 Topological Sort로 확인하면 된다.



1F. 미하일 2마리

Exponential by Squaring, Floyd-Warshall, ~~Berlekamp-Massey~~

출제진 의도 : Medium

- 제출 0번, 맞은 사람 0명 (정답률 0.00%)
- First Solve : —
- 출제자 : lky7674^{연세대학교}

1F. 미하일 2마리

- 그렇다고 합니다.



미하일 둘이서 신전사냥해도 서로피해1도안감

1F. 미하일 2마리

- 각 위치 쌍의 최단 경로를 Floyd-Warshall로 구한다.
 - 미하일의 위치 상태를 13X13개, 총 169개로 구분한다.
 - 기존에 구한 최단 경로를 이용해서 169 X 169 인접 Matrix를 만든다.
 - 해당 Matrix의 거듭제곱을 $O(169 \times 169 \times 169 \times \log(n))$ 에 답을 구할 수 있다.
-
- 정해 : C++ 기준 2900ms, Pypy3 기준 4660ms.

1F. 미하일 2마리

- 해당 최적화를 진행하지 않아도 통과하게 만들었습니다.
- 최적화 1
 - 불가능한 상태 개수가 많다.
 - 불가능한 상태 개수를 줄여서 Matrix 크기를 줄인다.
 - 해당 방법으로 32ms로 줄일 수 있다.
- 최적화 2
 - Berlekamp Massey로 매우 간단한 점화식을 얻어낼 수 있다.
 - 해당 방법으로 0ms로 줄일 수 있다.

2G. 좀비 떼가 전역 때보다 먼저 오다니

Backtracking

출제진 의도 : Hard

- 제출 0번, 맞은 사람 0명 (정답률 0.00%)
- First Solve : —
- 출제자 : 906bc^{홍익대학교}

2G. 좀비 떼가 전역 때보다 먼저 오다니

- 브루트 포스를 하는 경우 $O(B3_{\text{length}} 3^L)$ 로 시간 초과가 발생합니다.
- 백트래킹으로 가지치기를 하여 유망하지 않은 분기는 탐색에서 제외해야 합니다.
- 먼저 좀비가 0m에 접근하여 킬로와 헥토가 사망한 이후는 탐색에서 제외합니다.
- 추가적인 가지치기가 더 필요한데, 맨 앞의 좀비를 소총만 쏘았을 때 제압이 가능한 경우에만 소총을 쏘는 방법, 남은 사격수보다 산탄이 더 많은 경우 소총을 생략하고 산탄총을 쏘는 방법 등이 있습니다.

2G. 좀비 떼가 전역 때보다 먼저 오다니

- 시간 복잡도를 더 줄이기 위한 기법으로는 저격 소총의 관통 처리를 슬라이딩 윈도우로 하여 관통 처리를 $O(B_{\text{length}})$ 에서 $O(1)$ 로 줄이는 방법이 있으며, 또한 맨 앞의 좀비를 잡을 때는 사용할 탄종과 그 수가 중요하지 사격 순서는 중요하지 않으므로 소총-산탄총-저격 소총을 순서대로 쏘는 3단 DFS, 또는 조합을 먼저 구성한 후 사격하는 방법도 가능합니다.

2H. 좀비 떼가 기관총 진지에도 오다니

Greedy, Sliding window

출제진 의도 : Medium

- 제출 1번, 맞은 사람 0명 (정답률 0.00%)
- First Solve : —
- 출제자 : 906bc^{홍익대학교}

2H. 좀비 떼가 기관총 진지에도 오다니

- 1m의 좀비를 기관총 사격을 해도 제압할 수 없는 경우에만 수평 세열 지향성 지뢰를 격발하고, 그 외의 경우에는 기관총 사격을 하는 간단한 탐욕법을 사용할 수 있습니다.
- 기관총 사격 시 좀비의 체력을 반복문으로 하나 하나 감소한다면 관통 처리 한 번에 $O(M_L)$, 전체 시간 복잡도는 $O(M_L L)$ 이 되므로 시간 초과가 발생합니다.
- 이때 관통 처리를 슬라이딩 윈도우로 한다면 $O(M_L)$ 에서 $O(1)$ 로 줄일 수 있는데, 전체 시간 복잡도는 $O(L)$ 이 되므로 시간 제한을 통과할 수 있습니다.