PPC 2023

Official Solutions

PPC 2023 - 출제진 및 검수진



✓ 출제진

- 권찬우 kwoncycle
- 김범수 qjatn0120baek
- 오승혁 menborong
- 장래오 leo020630
- 정윤성 slah007

✓ 검수진

- eaststar
- hyperbolic
- jthis
- kohandy
- pasa3232
- tlsdydaud01
- tony9402

문제		의도한 난이도	출제자
A	모범생 포닉스	**Easy**	장래오leo020630
B	폭발 속에서 살아남기	**Hard**	오승혁menborong
C	이상한 배열	**Medium**	김범수qjatn0120baek
D	보물 사냥	**Challenging**	오승혁menborong
E	지구평면설	**Medium**	권찬우kwoncycle
F	단순한 그래프와 이상한 쿼리	**Hard**	장래오leo020630
F2	평범한 그래프와 이상한 쿼리	**Challenging**	장래오leo020630
G	대회 상품 정하기	**Easy**	오승혁menborong
H	삼각형 모험	**Hard**	장래오leo020630
I	분탕	**Challenging**	권찬우kwoncycle
J	나무 타기	**Medium**	정윤성slah007
J2	나무 타기 (Hard)	**Hard**	정윤성slah007
K	회전초밥	**Easy**	김범수qjatn0120baek
L	라이벌	**Medium**	장래오leo020630
M	약속 장소 2	**Medium**	장래오leo020630
N	마지막 문제	**Easy**	장래오leo020630

A. 모범생 포닉스

`arithmetic`
출제진 의도 – **Easy**

- 제출 254번, 정답 202명 (정답률 84.646%)
- First Solve: **riroan**, 1분
- 출제자: 장래오 `leo020630`

A. 모범생 포닉스



- 포닉스의 공부 계획에 사용된 총 공부 시간은 $\sum_{i=1}^{N} T_i$ 입니다.
- 포닉스의 공부 계획에 사용된 총 휴식 시간은 $8 \times (N-1)$ 입니다.
- 따라서 $\sum_{i=1}^{N} T_i + 8 \times (N-1)$ 이 며칠 몇 시간인지 구해야 합니다.
- 일 수는 정답을 24 로 나눈 몫, 시간은 나머지에 해당합니다.



B. 폭발 속에서 살아남기

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

- ✓ 제출 29번, 정답 5명 (정답률 17.241%)
- ✓ First Solve: **aeren**, 21분
- ✓ 출제자: 오승혁 `menborong`



- 원점과 정점을 이은 선의 수직 이등분선을 그어봅시다.
- 이 선 위의 점들은 모두 원점과 정점과의 거리가 같습니다.
- 따라서 원점에서 출발해서 어떠한 벙법으로도 이 수직 이등분선은 넘어갈 수 없습니다.



- 이러한 수직 이등분선들을 모두 그어봅시다.
- 만약 수직 이등분선들이 원점을 둘러싼다면, 어떠한 방법으로 도망쳐도 그 둘러싸인 공간을 벗어날 수 없습니다.
- 반대로, 수직 이등분선들이 둘러싸지 못한 열린 공간이 있다면, 그 방향으로 직진하면 어떠한 정점보다 해당위치를 항상 먼저 도착하므로 도망칠 수 있습니다.
- 이는 Voronoi diagram 상에서 원점을 site로 하는 cell이 unbounded 된다는 것과 동치입니다.

B. 폭발 속에서 살아남기



- 원점 O를 중심으로 정점들을 시계 반대방향 순서대로 $P_1, P_2, \ldots, P_N, P_{N+1} = P_1$ 이라고 합시다.
- 이때, $\angle P_i O P_{i+1} \geq \pi$ 인 i가 존재한다면 이러한 열린 공간이 존재하게 됩니다.
- 따라서 정점들을 360도 정렬하고, 각 i에 대해서 $\angle P_i O P_{i+1} \geq \pi$ 인지 확인하면 되고, 이는 외적(ccw)을 통해 구할 수 있습니다.
- 시간복잡도는 $O(N \log N)$ 입니다.



- ✓ 컨벡스 헐 알고리즘을 이용해서도 풀 수 있습니다.
- ✓ Voronoi diagram 상에서 cell이 unbounded 된다는 것은 문제에서 주어진 N 개의 점과 원점으로 이루어진 $N+1$ 개로부터 구해전 컨벡스 헐 위의 원점 O 가 존재한다는것과 동치입니다.
- ✓ 따라서 컨벡스 헐 알고리즘을 이용해서도 해결할 수 있습니다.
- ✓ 컨벡스 헐 계산 또한 $O(N \log N)$ 에 구현할 수 있습니다.

C. 이상한 배열

`tree_set`
출제진 의도 – **Medium**

- ✓ 제출 74번, 정답 18명 (정답률 25.676%)
- ✓ First Solve: **riroan**, 4분
- ✓ 출제자: 김범수`qjatn0120baek`

- 먼저 배열의 각 숫자가 어느 위치에서 나왔는 지를 체크해서 $A_i = A_j$ 조건을 확인할 수 있습니다.
- 배열의 특정 숫자 x에 대해 x가 배열에서 두 번 이상 등장한 경우에만 x를 고려해줘야 합니다.
- 이 경우, 배열에서 가장 빨리 등장하는 x와 배열에서 가장 마지막에 등장하는 x 사이에 있는 숫자들은 모두 x 이하가 되어야 합니다.

- 따라서 1이상 N 이하의 자연수 x에 대해 배열에서 x가 2번 이상 등장하는 지 확인하고, 2번 이상 등장한다면 가장 빨리 등장하는 x와 가장 늦게 등장하는 x 사이의 모든 숫자가 x 이하인지를 확인해야 합니다.
- 특정 범위 내의 숫자가 모두 x 이하인 지를 $O(N)$에 판단한다면 모든 자연수 x에 대해 판단하기 위해서는 $O(N^2)$의 사간복잡도가 필요하게 되어 시간초과로 틀리게 됩니다.

C. 이상한 배열



- set과 같은 자료구조를 이용하면 모든 과정을 $O(N \log N)$에 처리할 수 있습니다.
- 먼저 모든 x에 대해서 첫 번째로 등장하는 x와 마지막으로 등장하는 x를 기억해 둡니다.
- 이제 배열의 맨 처음부터 하나씩 보면서 아래 과정을 반복합니다.
  - 만약 A_i가 배열에서 맨 처음으로 등장한 값이라면 set에 A_i를 추가합니다.
  - 만약 A_i가 배열에서 가장 마지막으로 등장한 값이라면 set에 A_i를 제거합니다.
  - 만약 A_i가 set에 있는 가장 작은 값보다 크다면 정답은 No가 되고 과정을 종료합니다.
- 위 과정이 끝날 때까지 No가 출력되지 않았다면 Yes를 출력합니다.



- set에 있는 값들은 현재 배열의 값이 set에 있는 값보다는 작거나 같아야 합니다.
- 따라서 set에 있는 값들 중 가장 작은 값에 대해서만 확인해봐도 충분합니다.
- set에 값 추가, 삭제, 참조의 시간복잡도는 $O(\log N)$ 이므로 모든 과정을 $O(N \log N)$ 에 처리할 수 있어 정답을 받을 수 있습니다.
- 이 풀이 말고도 range minimum query를 수행하는 세그먼트 트리를 이용해서도 풀 수 있습니다.

D. 보물 사냥

`graph_traversal`, `scc`, `dynamic_programming`
출제진 의도 – **Challenging**

- 제출 29번, 정답 1명 (정답률 3.448%)
- First Solve: **aeren**, 109분
- 출제자: 오승혁menborong

D. 보물 사냥



- 먼저 레버가 없는 경우의 문제를 생각해봅시다.
- 같은 강한연결요소 위의 정점들 간에는 자유롭게 이동할 수 있기 때문에, 각 강한연결요소의 정점들은 하나의 정점으로, 그 정점의 보물의 가치는 강한 연결요소의 모든 보물의 가치의 합으로 생각할 수 있습니다.
- SCC 알고리즘을 통해 그래프를 DAG로 만들고, 그래프의 위상 순서대로 다이나믹 프로그래밍을 하는것으로 최대 비용을 얻을 수 있습니다.
- 이러한 과정들의 총 시간복잡도는 $O(N+M)$ 입니다.

D. 보물 사냥



- 이제 레버를 당기는 경우를 생각해봅시다.
- 정점 a에 도달하는 시점에서 간선이 추가되면서 그래프가 변하게 됩니다.
- 주어진 그래프 $G = (V, E)$를 두개의 그래프 $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$로 복사하고, 정점 $u \in V$가 그래프 G_1, G_2에 대응되는 정점들을 각각 u_1, u_2라고 합시다.
- 마찬가지로 B_1, B_2를 각 V_1, V_2의 정점들의 보물의 가치들을 저장한 배열이라고 합시다.
- 그래프의 상태가 변하는것을 표현하기 위해 간선 (a_1, a_2)을 추가해주고, 양방향 통로가 추가됐음을 표현하기 위해 간선 (x_2, y_2), (y_2, x_2)를 추가해줍시다.
- 하지만 이렇게 그래프를 구성하고 위와 같이 DP를 풀게 될 경우, 몇몇 정점들에서 중복적으로 보물을 얻게 됩니다.



- 어떤 정점 v에 대해서, 어떤 경로가 v_1도 방문했고, v_2도 방문했다고 합시다.
- 이 경우, v_1에서 a_1으로 가는 경로가 존재해야 하고, a_2에서 v_2로 가는 경로 역시 존재해야 합니다.
- 그런데 G_2는 G_1에서 간선들을 추가만 했기 때문에 v_2에서 a_2로 가는 경로 또한 존재해야합니다.
- 따라서 중복이 발생하는 모든 정점 v에 대해서, (v_2가 속한 SCC) = (a_2가 속한 SCC)입니다.



- 또한 레버를 당기는 케이스는 a_1 와 a_2 를 모두 방문하므로, 중복이 발생하는 모든 정점들을 모두 방문하고 a_2 로 돌아올 수 있습니다.
- 따라서 a_2 를 포함하는 SCC에 대응되는 모든 정점들이 모든 중복의 대상이 되는 정점들이고, G_1 상에서 대응되는 정점을 방문했다면 반드시 G_2 에서 중복으로 방문하고 a_2 로 돌아올 수 있습니다.
- 결론적으로 B_1 에서 이 정점들의 보물의 가치를 미리 0 으로 바꾸면 됩니다.
- 이러한 과정들의 총 시간복잡도는 마찬가지로 $O(N+M)$ 입니다.



- 최적의 답을 OPT 라고 합시다.
- 레버가 없는 케이스의 그래프의 DP 결과를 OPT_1, 레버가 당기는 경우의 케이스의 그래프의 DP 결과를 OPT_2 라고 합시다.
- 레버를 당기지 않는 케이스가 최적의 결과라면, $OPT = OPT_1$ 이고,
- 레버를 당기는 케이스가 최적의 결과라면, $OPT = OPT_2$ 입니다.
- 따라서 $OPT = \max(OPT_1, OPT_2)$ 입니다.
- 총 시간복잡도는 마찬가지로 $O(N + M)$ 입니다.



E. 지구평면설

`math`

출제진 의도 – **Medium**

- 제출 11번, 정답 2명 (정답률 18.182%)
- First Solve: **aeren**, 127분
- 출제자: 권찬우 `kwoncycle`



- ✓ 먼저 모든 h_{ij} 가 똑같아질 수 있는 조건을 찾아봅시다.
- ✓ 찬우가 하는 모든 시행에 대해서 아래의 불변값이 있는 것을 관찰할 수 있습니다.
  - i, j 에 대해, $h_{11}, h_{i1}, h_{1j}, h_{ij}$ 의 관계를 관찰해보면, 항상 $(h_{11} \times h_{ij})/(h_{i1} \times h_{1j})$ 가 일정하게 유지됨을 확인할 수 있습니다. 따라서 모든 h_{ij} 가 똑같아지기 위해서는 모든 i, j 에 대해, $(h_{11} \times h_{ij})/(h_{i1} \times h_{1j})$ 을 만족해야 합니다.
  - 역으로 항상 $(h_{11} \times h_{ij})/(h_{i1} \times h_{1j}) = 1$ 을 만족할 경우, 시행을 통해 모든 h_{ij} 를 동일하게 만들 수 있습니다.

E. 지구평면설



- 이제 $a_1, a_2, \ldots, a_{2n}$ 에 등장하는 수의 종류를 최소화해봅시다.
- 관찰로부터 $a_{2i-1} = a_1 h_{11}/h_{1i}$ 이고, 마찬가지로 $a_{2i} = a_2 h_{11}/h_{i1}$ 임을 알 수 있습니다. 즉, a_1, a_2 만 정해지면 다른 a_i 의 값들이 정해집니다.
- 이제 a_1/a_2 의 값을 적절히 정해주면서 $a_1, a_2, \ldots, a_{2n}$ 에 등장하는 수의 종류를 최소화해야 합니다.
- $a_{2i-1} = a_{2j}$ 라는 것은 $a_1/a_2 = h_{1i}/h_{i1}$ 와 동치이므로, 모든 i, j 에 대해 가장 많이 등장하는 h_{1i}/h_{i1} 를 찾아서 a_1/a_2 로 설정해주면 등장하는 수의 종류가 최소가 됩니다.
- 그렇게 구한 a_1/a_2 에서 서로 다른 수의 개수를 카운팅해주면 됩니다.

F. 단순한 그래프와 이상한 쿼리

`graphs`, `bipartite_graph`, `ad_hoc`
출제진 의도 – **Hard**

- ✓ 제출 11번, 정답 6명 (정답률 54.545%)
- ✓ First Solve: **hijkl2e**, 54분
- ✓ 출제자: 장래오 `leo020630`



- ✓ 우선 a와 b는 같은 연결 요소에 속해야 합니다.
- ✓ 일단 같은 연결 요소에 속한다면, $k = 1$에 대한 답은 항상 Yes입니다.
- ✓ 잘 생각해보면, $a \to b \to a \cdots$ 와 같은 식으로 k가 홀수일 때도 항상 경로를 만들 수 있다는 것을 알 수 있습니다.



- k가 짝수라고 가정합시다. 만약 2의 배수인 경로를 만들 수있다면, 어떤 간선을 왕복하는 것으로 경로의 길이를 2씩 늘릴 수 있으므로 $k = 2$로 가정하고 문제를 해결해도 무방합니다.
- $k = 2$인 경우에 대한 문제를 해결해 봅시다.
- 만약 길이가 홀수인 사이클이 있다면, 홀수 사이클을 한 번 도는 것으로 경로의 길이의 홀짝성을 바꿀 수 있습니다.
- 따라서 길이가 홀수인 사이클이 있다면 $k = 2$에 대한 답은 Yes입니다.



- ✓ 홀수 사이클이 없다면, 이 그래프는 이분 그래프가 됩니다.
- ✓ 이분 그래프에서 같은 집합에 속하는 두 정점 사이의 거리는 항상 짝수이고, 다른 집합이라면 항상 홀수입니다.
- ✓ 따라서 이분 그래프에서 a와 b가 같은 집합에 속한다면 Yes, 아니라면 No입니다.

F2. 평범한 그래프와 이상한 쿼리

`graphs`, `bipartite_graph`, `ad_hoc`
출제진 의도 – **Challenging**

- ✓ 제출 2번, 정답 1명 (정답률 50.000%)
- ✓ First Solve: **aeren**, 141분
- ✓ 출제자: 장래오 `leo020630`



- F에서 달라지는 부분은 다음과 같습니다.
- $k = 2$이고 홀수 사이클이 없는 상황을 생각합시다.
- 이 때, F에서는 두 정점이 이분 그래프에서 같은 집합에 속한다면 모든 짝수인 k에 대해 경로를 만들 수 있었습니다.
- 하지만 이 문제에서는 그렇지 않습니다.



- ✓ 예를 들어, 정점이 두 개이고 두 정점을 가중치가 2인 간선으로 잇는 그래프에서는 경로의 길이가 4의 배수인 경로가 존재하지 않습니다.
- ✓ 이를 잘 살펴보면, F와 같은 이유로 연결 요소에 가중치가 홀수인 간선이 하나라도 있다면 짝수 간선 $\rightarrow$ 홀수 간선 $\rightarrow$ 홀수 간선 $\rightarrow$ 짝수 간선과 같은 방법으로 4로 나눈 나머지를 2씩 증가시킬 수 있다는 것을 알 수 있습니다.
- ✓ 따라서 홀수 사이클이 없고, 두 정점이 이분 그래프에서 같은 집합이며 연결 요소에 가중치가 홀수인 간선이 존재한다면 모든 짝수인 k에 대해 경로를 만들 수 있습니다.



- ✓ 그렇다면 모든 간선의 가중치가 짝수인 경우에 $k = 4$는 어떻게 판별할까요?
- ✓ 모든 간선의 가중치가 짝수이므로 모든 간선의 가중치를 2로 나눈 후, $k = 2$인 문제를 푸는 것과 동치입니다.
- ✓ $k = 2^x$ 인 경우에도 이와 같은 과정을 재귀적으로 처리해주면 됩니다.

F2. 평범한 그래프와 이상한 쿼리



- ✓ 사전 지식이 필요한 대신 구현이 더 쉬운 별해가 존재합니다.
- ✓ 그래프 위에서 무한히 움직인다는 것은 경로의 길이가 사이클의 길이의 선형결합과 임의의 경로 하나의 합으로 표현된다는 뜻입니다.
- ✓ 따라서, a에서 시작해 b에서 끝나는 임의의 경로의 길이를 l이라 하면 $l + \sum x_i c_i = yk$가 되어야 합니다. x_i, y는 임의의 계수, c_i는 각 사이클의 길이입니다. 사이클에는 길이가 2인 사이클, 즉 $a \to b \to a$와 같은 형태도 포함됩니다.
- ✓ 결과적으로 부정방정식의 조건에 따라 $gcd(k, c_i) | l$이어야 합니다. $gcd(k, c_i)$는 모든 사이클의 길이와 k에 대한 최대공약수입니다.
- ✓ 그렇다면 모든 사이클의 길이는 어떻게 구할까요? 잘 생각해보면, $2w_i$와 같은 꼴이 이미 사이클에 포함되어 있으므로 fundamental cycle의 선형결합으로 모든 사이클을 표현할 수 있습니다. 따라서 fundamental cycle의 길이만 고려해주면 됩니다.

G. 대회 상품 정하기

`greedy`, `math`
출제진 의도 – **Easy**

- ✓ 제출 66번, 정답 19명 (정답률 28.788%)
- ✓ First Solve: **hijkl2e**, 66분
- ✓ 출제자: 오승혁 `menborong`



- 각 참가자에 대해서 하나하나 다음 상품을 할당할 수 있는지 확인하면 $O(NM)$ 또는 $O(N+M)$에 문제를 해결할 수 있지만, N이 최대 10^{12}이므로 제한시간 내에 해결하기 힘듭니다.
- 관점을 각 참가자가 아닌 각 상품으로 바꿔봅시다.

G. 대회 상품 정하기



- $a_1, a_2, \ldots, a_M$ 은 이미 단조 감조 순서로 정렬되어 있고, 1 번부터 순서대로 구매를 시도해봅시다.
- i 번째 상품을 구매하기 시도할 때의 남은 비용을 X_i, 남은 참가자의 수는 N_i 라고 합시다.
- 그러면 남은 참가자들이 구매할 수 있는 한도에서 최대한 구매 가능한 수는 $\left\lfloor \frac{X_i - N_i \cdot a_M}{a_i - a_M} \right\rfloor$ 가 됩니다.
- 하지만 이 경우 남은 인원 N_i 보다 커질 수 있으므로, 실제 가능 구매한 수는 $\text{ans}_i = \min\left(N_i, \left\lfloor \frac{X_i - N_i \cdot a_M}{a_i - a_M} \right\rfloor\right)$ 가 됩니다.
- 또한 $X_{i+1} = X_i - \text{ans}_i \cdot a_i$, $N_{i+1} = N_i - \text{ans}_i$ 로 계산할 수 있습니다.
- 이 과정은 $O(M)$ 에 가능하며, 제한시간 내에 충분히 계산 가능합니다.



H. 삼각형 모험

`graph_traversal`
출제진 의도 – **Hard**

- ✓ 제출 2번, 정답 2명 (정답률 100.000%)
- ✓ First Solve: **aeren**, 167분
- ✓ 출제자: 장래오 `leo020630`

H. 삼각형 모험



- 각 삼각형을 정점이라 생각하고 그래프를 구성해봅시다.
- 지도의 모양에 따라, 각 정점의 차수는 최대 2입니다.
- 정점의 차수가 최대 2라는 성질 때문에 각 연결 요소는 원 모양의 단순 사이클이거나, 직선 모양의 체인이 됩니다.
- 각 연결 요소에서의 방문 순서를 구해 놓는다면 최단 거리는 $O(1)$에 구할 수 있습니다.
- 체인에서는 단순 연산을 통해, 사이클에서는 원에서의 두 경로 중 작은 것을 택하면 됩니다.



- 따라서 그래프를 잘 구성한 후 탐색 결과를 잘 저장하면 됩니다.
- 저장해야 할 정보는 해당 연결 요소가 사이클인지, 그리고 각 정점의 탐색 순서입니다.
- 원 모양에서의 탐색 순서를 잘 저장하기 위해 BFS 대신 DFS를 사용해야 합니다.
- 선 모양에서는 차수가 1인 정점에서 탐색을 시작해야 탐색 순서를 잘 저장할 수 있습니다.
- 이를 유의하며 구현하면 문제를 해결할 수 있습니다.

I. 분탕

`combinatorics`
출제진 의도 – **Challenging**

- ✓ 제출 3번, 정답 1명 (정답률 33.333%)
- ✓ First Solve: **aeren**, 229분
- ✓ 출제자: 권찬우`kwoncycle`

- 먼저 LDS의 길이가 2일 조건을 찾아야 합니다. (a_1, b_1)과 (a_2, b_2)가 각각 하나의 pair로 묶인다고 합시다.($a_1 < b_1, a_2 < b_2$)
- 만약 $a_1 < a_2 < b_2 < b_1$라면 위치가 바뀐 후에 돌의 위치는 차례대로 b_1, b_2, a_2, a_1가 되고, 이 4개의 수는 감소 수열을 이루어서 LDS의 길이가 2를 넘게 됩니다.
- 따라서 임의의 두 pair가 있을 때, 하나의 pair는 다른 pair를 포함하지 않아야 합니다.
- 반대로 임의의 두 pair에 대해, 하나의 pair가 다른 pair를 포함하지 않는다면 LDS의 길이가 정확히 2가 됩니다.

- 이제 돌 i가 더 작은 번호의 돌과 pair가 맺어질 경우 오른쪽 돌, 더 큰 번호의 돌과 맺어질 경우 왼쪽 돌이라고 정의합시다.
- 이때 어느 pair도 다른 pair를 포함하지 않기 때문에, $2N$개의 돌들이 각각 왼쪽 돌인지 오른쪽 돌인지만 정해주면 전체 pairing이 어떻게 맺어지는지 구할 수 있습니다.
- $X+1$부터 $Y-1$번째까지의 돌에서 왼쪽 돌과 오른쪽 돌의 개수가 고정되면, Catalan's triangle을 통해 쉽게 총 경우의 수를 구할 수 있습니다.
- 구체적으로, $X+1$부터 $Y-1$ 사이의 왼쪽 돌의 개수가 a개라면, 1부터 $X-1$번까지의 돌 중 왼쪽 돌의 개수는 $l_1 = (X-1+a)/2$개, 오른쪽 돌의 개수는 $r_1 = (X-1-a)/2$개가 존재합니다.

- 이 조건 하에 각 돌이 왼쪽 돌인지 오른쪽 돌인지 판단하는 경우의 수는, 괄호 $($ l_1 개와 $)$ r_1 개로 만드는 regular bracket sequence의 개수를 구하는 문제로 생각할 수 있습니다.
- 이때 카탈란 수를 구하는 과정에서 $y = x + 1$을 기준으로 대칭하는 테크닉을 사용하면, 돌 1에서 $X - 1$까지 왼쪽, 오른쪽 돌을 지정하는 경우의 수는 $\binom{l_1 + r_1}{r_1} - \binom{l_1 + r_1}{r_1 - 1}$가 됩니다.
- 마찬가지로 돌 $Y + 1$부터 $2N$까지 지정하는 경우의 수 또한 구할 수 있습니다. $X + 1$부터 $Y - 1$사이의 오른쪽 돌의 개수는 $Y - X - 1 - a$개이므로 돌 $Y + 1$부터 $2N$사이의 오른쪽 돌의 개수는 $r_2 = (2n - Y + (Y - X - 1 - a))/2$개, 왼쪽 돌의 개수는 $l_2 = (2n - Y - (Y - X - 1 - a))/2$개이고, 총 경우의 개수는 $\binom{r_2 + l_2}{l_2} - \binom{r_2 + l_2}{l_2 - 1}$가 됩니다.

- 최종적으로 $X+1$ 부터 $Y-1$ 사이의 왼쪽 돌의 개수가 a 개라면, 돌 $1 \sim X-1$ 까지 왼쪽, 오른쪽을 결정하는 경우의 수가 $\binom{l_1+r_1}{r_1} - \binom{l_1+r_1}{r_1-1}$, 돌 $X+1 \sim Y-1$ 까지 왼쪽, 오른쪽을 결정하는 경우의 수가 $\binom{Y-X-1}{a}$, 돌 $Y+1 \sim 2N$ 까지 왼쪽, 오른쪽을 결정하는 경우의 수가 $\binom{r_2+l_2}{l_2} - \binom{r_2+l_2}{l_2-1}$ 이므로, 이를 다 곱한 것이 $X+1$ 부터 $Y-1$ 사이의 왼쪽 돌의 개수가 a 인 경우의 수입니다.
- 따라서 $a = 0, 1, \ldots, Y-X+1$ 일 때 경우의 수를 모두 더한 것이 전체 경우의 수입니다.

J. 나무 타기

`dp_tree`, `graph_traversal`
출제진 의도 – **Medium**

- ✓ 제출 13번, 정답 8명 (정답률 61.538%)
- ✓ First Solve: **hijkl2e**, 66분
- ✓ 출제자: 정윤성`slah007`

J. 나무 타기



- $dp[i]$를 루트 정점에서 시작해 i번 정점을 도착점으로 하는 나무 타기 놀이의 수로 정의합시다.
- 각 정점과 루트와의 거리를 d_i라 하면 $dp[i]$는 i의 조상들 p에 대해 다음과 같이 정의할 수 있습니다.
- $dp[i] = \sum_{d_i - d_p \le a_p} dp[p]$
- 이를 적절한 방법으로 구현해주면 됩니다. DP식의 정의에 따라 여러 다른 풀이가 있을 수 있습니다.
- 시간 복잡도는 $O(N^2)$입니다.



J2. 나무 타기 (Hard)

`dp_tree`, `graph_traversal`, `prefix_sum`
출제진 의도 – **Hard**

- ✓ 제출 4번, 정답 3명 (정답률 75.000%)
- ✓ First Solve: **p_ce1052**, 148분
- ✓ 출제자: 정윤성[slah007]

J2. 나무 타기 (Hard)



- $dp[i]$ 를 루트 정점에서 시작해 i 번 정점을 도착점으로 하는 나무 타기 놀이의 수로 정의합시다.
- 각 정점과 루트와의 거리를 d_i 라 하면 $dp[i]$ 는 i 의 조상들 p 에 대해 다음과 같이 정의할 수 있습니다.
- $dp[i] = \sum_{d_i - d_p \leq a_p} dp[p]$
- 이를 빠르게 계산해야 합니다. 하지만 a_i 가 모두 달라 적절한 트리 알고리즘을 사용하기 어렵습니다.



- DFS 과정을 잘 응용해 봅시다. DFS로 트리를 탐색할 때, 현재 방문 중인 정점의 조상과 자신만 남긴다면 마치 선형 배열과 같이 생각할 수 있습니다.
- $dp[x]$를 계산했다면, 이 $dp[x]$가 더해져야 할 구간은 $d_i - d_x \leq a_x$인 구간입니다. 그리고 이 구간은 연속적입니다.
- 따라서, 이를 세그먼트 트리나 누적합 배열(imos법)을 이용해 관리해줄 수 있습니다.
- 이를 DFS 탐색을 시작할 때 더하고, 끝낼 때 빼 준다면 $dp[x]$ 값은 자신의 자손 노드에게만 영향을 주게 되므로 문제를 해결할 수 있습니다.



K. 회전초밥

`data_structure`, `queue`
출제진 의도 – **Easy**

- ✓ 제출 46번, 정답 25명 (정답률 56.522%)
- ✓ First Solve: **infikei**, 23분
- ✓ 출제자: 김범수`qjatn0120baek`

K. 회전초밥



- 각 사람이 어떤 초밥을 먹는지가 아니라 각 초밥을 누가 먹는지에 집중하면 쉽게 풀 수 있는 문제입니다.
- 먼저 음식의 종류 수만큼의 큐를 정의합니다. 각 큐는 해당 초밥을 먹고 싶어하는 사람의 번호를 저장합니다.
- 이후 각 음식을 먹고싶어 하는 사람들을 먼저 먹는 순서로 큐에 저장합니다. 즉 번호가 빠른 사람을 먼저 저장합니다.
- 이제 특정 초밥이 만들어지면, 해당 초밥 큐의 맨 앞에 있는 사람을 가져옵니다. 이후 이 사람이 먹는 초밥의 개수를 1만큼 증가시킵니다.
- 이 과정을 모든 초밥에 대해 반복하면 각 사람이 몇 개의 초밥을 먹었는지를 확인할 수 있습니다.
- 위의 모든 과정의 시간복잡도는 $O(N+M)$으로 정답을 받을 수 있습니다.

L. 라이벌

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

- ✓ 제출 17번, 정답 5명 (정답률 29.412%)
- ✓ First Solve: **golazcc83**, 65분
- ✓ 출제자: 장래오 `leo020630`



- 각 학생에 대한 역라이벌의 수는 나보다 최소 하나의 능력치가 낮은 학생의 수입니다.
- 이는 전체 학생 수에서 나보다 모든 능력치가 높은 학생의 수를 뺀 것과 같습니다.
- 각 학생의 능력치를 모두 반전시켜 봅시다. 그러면 우리가 구해야 할 것은 각 학생에 대해 해당 학생보다 모든 능력치가 낮은 학생의 수입니다.



- 잘 생각해보면, 가질 수 있는 능력치의 가짓수는 $6^6 = 46656$ 가지입니다.
- 상태의 수가 적으므로 이를 배열로 잘 저장해봅시다.
- $dp[a_1][a_2][a_3][a_4][a_5][a_6]$ 을 i 번째 능력치가 a_i 이하인 학생의 수로 정의합시다.
- 이는 누적합 배열의 형태가 됩니다.



- ✓ 이제 해당 누적합 배열을 잘 채우면 됩니다. 이는 3가지 방법 정도로 해결할 수 있습니다.
- ✓ 1. 각 능력치에 대해 자신보다 모든 영역의 값이 작은 능력치만 방문하는 방법이 존재합니다. 시간 복잡도는 커 보이지만 탐색해야 할 공간의 수가 21^6 임을 보일 수 있습니다.
- ✓ 2. 2차원 누적합에서 주로 사용하는 포함배제의 원리를 사용할 수 있습니다. 이 경우의 시간 복잡도는 $O(12^6)$ 입니다.
- ✓ 3. 누적합을 다른 방식으로 계산할 수 있습니다. 각 차원에 대해 누적합을 한 번씩 계산하면 $O(6 \times 6^6)$ 에 문제를 해결할 수 있습니다.

M. 약속 장소 2

`string`, `greedy`
출제진 의도 – **Medium**

- ✓ 제출 40번, 정답 11명 (정답률 27.500%)
- ✓ First Solve: **hijkl2e**, 38분
- ✓ 출제자: 장래오 `leo020630`



- ✓ 주어진 문제 상황과 사전순 정렬에 대한 의미를 잘 파악해야 하는 문제입니다.
- ✓ 우선, 만들 수 있는 문자열의 개수는 $25N + 1$ 개입니다. K 가 이보다 크다면 -1 을 출력하면 됩니다.
- ✓ 이제 순서를 구해봅시다. 가능한 문자열을 모두 저장하는 방법으로는 시간과 메모리 모두 제한을 만족하며 문제를 풀 수 없습니다.
- ✓ 보다 효율적인 방법을 생각해봅시다. 크게 두 가지의 풀이가 있습니다.



1. 주어진 문자열들의 구조를 이용한 풀이

- ✓ 우선, 사전순으로 가장 빠른 문자열은 첫 문자를 A로 바꾼 경우입니다.
- ✓ 그렇게 첫 문자를 A, B, C...로 변경한 문자열이 순서대로 위치하며, 이는 원래 문자열의 첫 문자에 도달할 때까지 계속됩니다.
- ✓ 첫 문자가 그대로라면, 두 번째 문자를 A로 바꾸는 것이 최적입니다. 이를 마지막 문자까지 반복하면 원래 문자열에 도달합시다.
- ✓ 같은 원리로, 이번에는 마지막 문자부터 서서히 증가시켜 나가면 됩니다.
- ✓ 예를 들어, 원래 문자열이 POS였다면 문자열들의 순서는 AOS → OOS → PAS → PNS → POA → POS → POZ → PPS → PZS → QOS → ZOS 와 같은 형태입니다.



2. 비교 연산을 잘 정의하는 정렬 풀이

- ✓ 문자열을 비교해 정렬하는 풀이를 사용할 수 없는 이유는 시간 복잡도가 $O(N^2 \log N)$, 공간 복잡도가 $O(N^2)$ 이기 때문입니다.
- ✓ 하지만 잘 생각해보면, 비교해야 하는 문자열들은 문자열의 대부분이 같은 구조를 띄고 있습니다.
- ✓ 각 문자열을 (바꾼 위치, 바꾼 문자)로 표현합시다.
- ✓ 만약 바꾼 위치가 같으면, 바꾼 문자끼리 비교하면 됩니다.
- ✓ 바꾼 위치가 다르다면, 두 위치 중 앞에 오는 위치에 해당하는 바뀐 문자와 원래 문자를 비교하면 됩니다.
- ✓ 이러면 $O(N \log N)$ 에 모든 문자열을 정렬할 수 있습니다.



N. 마지막 문제

`sorting`, `greedy`
출제진 의도 – **Easy**

- ✓ 제출 124번, 정답 48명 (정답률 38.710%)
- ✓ First Solve: **jh05013**, 8분
- ✓ 출제자: 장래오 `leo020630`



- 기존 문제들은 난이도 순서로 정렬되어 있다 봐도 무관합니다. 따라서 정렬을 하고 시작합시다.
- N 이 작으므로 적절한 $O(N^2)$ 이나 $O(N \log N)$ 정렬 알고리즘을 사용하면 됩니다.
- 문제를 추가할 수 없는 경우부터 판별해봅시다. 모든 $1 \leq i < N$ 에 대해 $A_i + 1 = A_{i+1}$ 이라면 문제를 추가할 수 없습니다.
- 그 외의 경우에는 문제를 최소 하나는 추가할 수 있습니다.

N. 마지막 문제



- 추가할 문제의 난이도를 X 라 합시다. 문제의 조건에 따라, 기존 N 개의 문제들 중 X 보다 쉬우면서 가장 어려운 문제와 X 보다 어려우면서 가장 쉬운 문제가 유일하게 존재하며 이 둘은 정렬된 배열에서 인접합니다.
- 이를 각각 A_i, A_{i+1} 이라 합시다.
- 우리의 목표는 $\min(X - A_i, A_{i+1} - X)$ 를 최대화하는 X 를 구하는 것입니다.



- 가능한 X 의 후보의 개수가 많기 때문에, 모든 X 에 대해 이를 모두 구하는 방법으로는 주어진 시간 내에 문제를 해결할 수 없습니다.
- 잘 생각해보면, 각각의 i 에 대해 최적의 X 를 구할 수 있습니다.
- $X = \lfloor \frac{A_i + A_{i+1}}{2} \rfloor$ 일 때가 각 i 에 대해 $\min(X - A_i, A_{i+1} - X)$ 를 최대화하는 X 입니다.
- 따라서, 모든 $X = \lfloor \frac{A_i + A_{i+1}}{2} \rfloor$ 에 대해 $\min(X - A_i, A_{i+1} - X)$ 의 값이 가장 큰 X 가 답이 됩니다.