2023 KSAAC Summer

Official Solutions

by

Automata

2023년 8월 18일

문제		의도한 난이도	출제자
A	소수가 아닌 수 2	**어려워요**	flakepowders
B	아침 태권도	**어려워요**	juneekim7
C	이번에는 C번이 문자열	**어려워요**	eric00513
D	활쏘기 대결	**어려워요**	flakepowders
E	문제 수 줄이기	**어려워요**	mickeyjung
F	K-정렬 게임	**어려워요**	mickeyjung
G	책가방	**어려워요**	eric00513
H	테마파크	**어려워요**	mickeyjung
I	KSA에서 숨바꼭질	**어려워요**	mickeyjung
J	마라톤	**어려워요**	mickeyjung

Thanks to

- 후원
  - BaaaaaaarkingDog
  - utilforever
  - spat
  - parkky
  - bennyk0406

- 운영진
  - eric00513
  - flakepowders
  - juneekim7
  - mickeyjung

- 검수진
  - ai4youej
  - cologne
  - dohoon
  - gs20036
  - hjroh0315
  - jjkmk1013
  - jk410
  - kiwiyou
  - leo020630
  - parkky
  - petamingks
  - runnie0427
  - shiftpsh
  - spat
  - utilforever
  - yongwhan

A. 소수가 아닌 수 2

`math`

출제진 의도 – **어려워요**

- 제출 1063번, 정답 341명 (정답률 33.114%)
- 처음 푼 사람: **nick832**, 1분
- 출제자: `flakepowders`

- k가 최대 소숫점 여덟 자리까지 주어지므로, $10^9 \times k$는 반드시 정수입니다.
- 따라서 $k = \dfrac{10^9 \times k}{10^9}$로 표현하면 됩니다.

B. 아침 태권도

math

출제진 의도 – **어려워요**

- 제출 1150번, 정답 309명 (정답률 27.652%)
- 처음 푼 사람: **nflight11**, 2분
- 출제자: `juneekim7`

- $(0,0)$과 (X_i, Y_i)를 잇는 선분을 만들었을 때, 해당 선분이 x축과 이루는 각의 서로 다른 개수를 세는 문제입니다.
- 각을 나타내는 방법은 기울기, 삼각함수 등 다양한 방법이 있습니다.
- 단, 소수점을 사용할 경우 오차 범위를 주의해야 합니다. 예를 들어, 기울기를 저장할 경우 C 언어의 float 대신 double을 이용해야 합니다.
- 또한, 각을 나타낼 때 모든 사분면과 축 위의 점 (X_i, Y_i)를 잘 나타낼 수 있는지 확인해야 합니다.
- 집합 혹은 맵을 이용하면 $\mathcal{O}(N \log N)$의 시간복잡도로 정답을 구할 수 있습니다.

C. 이번에는 C번이 문자열

`greedy, sorting`
출제진 의도 – **어려워요**

- 제출 674번, 정답 270명 (정답률 40.950%)
- 처음 푼 사람: **xiaowuc1**, 5분
- 출제자: `eric00513`

C. 이번에는 C번이 문자열

- **서브태스크 1**은 $M = 1$이므로 N개의 문자들을 정렬하고 앞에 있는 K개를 붙여서 만들면 됩니다.
- **서브태스크 2**는 모든 문자열이 동일하므로 어떤 K개를 고르더라도 답은 같습니다.
- **서브태스크 3**은 그리디한 접근으로 해결할 수 있습니다.
- 우선, 각 문자열의 문자들을 정렬합니다. 예를 들어 *ACBA*는 *AABC*가 됩니다.
- 그다음 $S_1, S_2, \cdots, S_N$을 사전순으로 정렬하여 앞 K개를 이어붙입니다.
- 이렇게 만든 문자열의 문자들을 다시 정렬해주면 문제에서 원하는 T가 됩니다.

D. 활쏘기 대결

`dp`, `game_theory`
출제진 의도 – **어려워요**

- 제출 359번, 정답 79명 (정답률 22.284%)
- 처음 푼 사람: **jeroenodb**, 12분
- 출제자: `flakepowders`

- S_i를 1번부터 i번 표적의 점수의 합으로 정의합니다.
- R_i를 1번부터 i번 표적까지만 고려했을 때 문제의 정답으로 정의합니다.
- $R_1 = \max(0, S_1)$입니다. $S_1 < 0$이라면 기권하는 것이 이득이고, 아니라면 1번 표적을 맞히는 것이 이득이기 때문입니다.
- $R_i = \max(R_{i-1}, S_i - R_{i-1})$입니다. i번 표적을 맞힌다면 그 점수를 얻는 대신 상대에게 R_{i-1}점을 줘야 하고, 그 위의 표적을 맞히거나 기권하면 그 점수를 상대에게 주는 대신 R_{i-1}점을 확보할 수 있기 때문입니다.
- 그러므로 $\mathcal{O}(N)$에 정답을 구할 수 있습니다.
- 별해로, $\mathcal{O}(N^2)$ DP를 세그먼트 트리 등을 이용해 최적화해서 $\mathcal{O}(N \log N)$에 정답을 구할 수도 있습니다.

E. 문제 수 줄이기

dp
출제진 의도 – **어려워요**

- 제출 345번, 정답 70명 (정답률 20.870%)
- 처음 푼 사람: **ainta**, 12분
- 출제자: `mickeyjung`

E. 문제 수 줄이기

- 모든 구간의 길이가 4 이하이면서 레벨 합을 최대로 만들 수 있습니다.
- 증명은 다음과 같습니다.
- 일단 레벨 합을 최대로 하면서 주어진 배열을 분할하는 방법들 중 아무 방법으로 분할합니다.
- 여기서 만들어진 길이가 5 이상인 구간들에 대해 길이 4 이하의 구간들로 쪼갤 수 있습니다.
  - 구간의 길이가 5 면 $(1,4)$, $(2,3)$, $(3,2)$ 중 하나로 반드시 쪼갤 수 있습니다.
  - 구간의 길이가 6 이면 $(2,4)$, $(1,2,3)$, $(3,2,1)$ 중 하나로 반드시 쪼갤 수 있습니다.
  - 구간의 길이가 7 이상이면 왼쪽에서부터 길이 1 또는 길이 2 의 구간을 떼어가면서 구간의 길이가 5 또는 6 이 되면 위와 같이 쪼개면 됩니다.
- 항상 $P + Q \geq P \oplus Q$ 이므로 위와 같은 방법으로 만들어진 분할도 문제의 정답이 됩니다.

- 이 사실을 파악했다면 다음과 같은 DP 정의로 문제를 해결할 수 있습니다.
- $dp[i][j]$ = i 번째 인덱스까지 구간을 나눴고 마지막 구간의 길이가 j 인 경우에 레벨 합의 최댓값
- 시간복잡도는 $\mathcal{O}\left(16N\right)$ 입니다.

F. K-정렬 게임

`game_theory, segment_tree`

출제진 의도 – **어려워요**

- 제출 229번, 정답 14명 (정답률 6.114%)
- 처음 푼 사람: **ormlis**, 78분
- 출제자: `mickeyjung`

- 먼저 임의의 순열에서 어떤 두 위치를 swap 하더라도 inversion의 기우성이 변한다는 사실을 알고 있어야 합니다.
- $A_k, A_{k-1}, \cdots, A_1$을 L, $A_{k+1}, A_{k+2}, \cdots, A_N$을 R이라고 정의하겠습니다.
- 일반성을 잃지 않고 1이 L에 존재한다고 가정합니다.
- 그리고 전체 inversion을 L의 inversion $+$ R의 inversion이라고 정의하겠습니다.

- $A_k = 1$인 경우
  - L과 R이 독립적입니다. 즉, L에 있는 수가 R로 넘어갈 수 없고, 마찬가지로 R에 있는 수도 L로 넘어갈 수 없습니다.
  - 따라서 게임 조건에 따라 어떻게 행동해도 전체 inversion이 홀수면 민찬이가 이기고 짝수면 준이가 이깁니다.
- $A_k \neq 1$이고 전체 inversion이 홀수인 경우
  - 민찬이가 1이 있는 위치와 k번 위치를 swap해서 L과 R이 독립적이게 만들어 승리할 수 있습니다.

✓ $A_k \neq 1$이고 전체 inversion이 짝수인 경우

- 민찬이와 준이는 전체 inversion의 기우성이 바뀌지 않게 해야 합니다.
- 만약 기우성이 바뀌었다면 다음 턴에 상대방이 1을 k번째 인덱스에 위치시킴으로써 승리할 수 있기 때문입니다.
- 기우성을 유지시키려면 k번 위치와 R에 있는 어떤 위치를 swap해야 합니다.
- 이때, 이 어떤 위치를 x라고 하면 A_k와 A_x의 기우성은 달라야 합니다.
- $A_x < A_i < A_k$인 모든 위치 i가 각각 전체 inversion을 $+1$ 또는 -1 시키기 때문입니다.
- 따라서 한 번 swap 할 때마다 A_k의 기우성이 바뀌고 마지막에는 R에서 가장 작은 수와 swap 하게 되므로 처음 A_k와 R에서 가장 작은 수의 기우성이 다르면 민찬이가 이기고 같으면 준이가 이깁니다.
- 단, 예외적으로 R에서 가장 작은 수가 A_k보다 큰 경우에는 항상 준이가 이깁니다.

- inversion을 펜윅 트리나 세그먼트 트리 등을 이용해서 계산하면 시간복잡도 $\mathcal{O}(N \log N)$에 문제를 해결할 수 있습니다.

G. 책가방

greedy, sorting, priority_queue, tree_set
출제진 의도 – **어려워요**

- 제출 392번, 정답 27명 (정답률 6.888%)
- 처음 푼 사람: **um_nik**, 40분
- 출제자: eric00513

- **서브태스크 2**는 책을 가져가는 $\binom{N}{K}$ 가지의 경우들을 모두 시도해볼 수 있습니다.
- **서브태스크 3**은 $T_i = 0$ 이므로 W_i 와 V_i 만 고려하면 됩니다.
- 우선 책들을 V_i 를 기준으로 오름차순 정렬합니다.
- x 번 책은 반드시 가져가고 나머지 $K-1$ 권은 $1, 2, \cdots, x-1$ 번 책 중에 고른다고 가정하면, V_i 의 최댓값은 V_x 로 고정됩니다.
- $x = 1, 2, \cdots, N$ 순서로 진행하며 책들을 우선순위 큐에 하나씩 넣는 동안 우선순위 큐의 크기가 항상 $K-1$ 이 되도록 유지하면 W_i 들의 합의 최솟값을 구할 수 있습니다.
- 이때 우선순위 큐는 W_i 가 가장 큰 책이 Top으로 오게 해야 합니다.
- 시간 복잡도는 $\mathcal{O}(N \log N)$ 입니다.

- **서브태스크 4**는 책들을 T_i를 기준으로 내림차순 정렬한 후, **서브태스크 3**의 풀이를 똑같이 적용하면 됩니다.
- x번 책을 반드시 가져가고 나머지 $K-1$권을 $1, 2, \cdots, x-1$번 책 중에 고른다고 가정하면, T_i의 최솟값이 T_x로 고정되기 때문입니다.
- 시간 복잡도는 $\mathcal{O}\left(N^2 \log N\right)$입니다.

- **서브태스크 5**에서는 **서브태스크 3**과 마찬가지로 책들을 V_i를 기준으로 오름차순 정렬한 후 x를 1씩 증가시키면서 피로도의 최솟값을 계산합니다.
- x번 위치에 고정한 책 하나를 제외하고 남은 $K-1$권 중, $K-2$개는 W_i가 작은 순으로 선택합니다.
- 이때 선택된 $K-1$개의 책들의 T 값 중 최솟값을 T_{min}이라고 하겠습니다.
- 이제 나머지 하나의 책 y를 선택할 때, 두 가지 경우로 나눌 수 있습니다.
  - $T_y < T_{min}$인 경우 정답은 $W_y + T_y - T_{min}$ 만큼 더해지게 됩니다.
  - $T_y \geq T_{min}$인 경우 정답은 W_y 만큼 더해지게 됩니다.
- 따라서 x가 증가함에 따라 $W_y + T_y$의 최솟값과, W_y의 최솟값을 갱신해주고, T_{min}은 multiset이나 세그먼트 트리 등을 이용해서 관리해주면 $\mathcal{O}(N \log N)$에 풀 수 있습니다.

H. 테마파크

greedy, priority_queue, small_to_large
출제진 의도 – **어려워요**

- 제출 112번, 정답 12명 (정답률 10.714%)
- 처음 푼 사람: **ainta**, 79분
- 출제자: mickeyjung

H. 테마파크

- 테마파크를 1번 정점을 루트로 하는 트리라고 생각하겠습니다.
- i번 구역과 그 구역의 부모를 잇는 길에 행사를 여는 것을 i번 구역을 선택한다라고 정의하겠습니다.
- 깊이가 깊은 정점부터 차례로 볼 때, 다음과 같은 알고리즘을 생각해볼 수 있습니다.
- i번 구역이 유료로 운영된다면 i번 구역을 선택합니다.
- i번 구역이 무료로 운영된다면 i번 구역을 선택할지 말지 결정합니다.
- 만약 i번 구역을 선택한다면 이미 선택된 구역들 중 몇 개를 취소할 수 있습니다. 이때, 취소하는 구역들 중 자식-조상 관계인 쌍이 있다면 어떤 구역까지 가는 경로에서 티켓을 받는 횟수가 줄어드므로, 그런 쌍이 있으면 안됩니다.
- 따라서 자식-조상 관계인 쌍이 없는 집합들 중 비용 합이 가장 큰 집합을 알 수 있으면 됩니다.

- 위 조건을 만족하면서, 현재 선택된 구역들로 만들 수 있는 집합들 중 비용 합이 가장 큰 집합을 $S_{i,0}$ 이라고 하겠습니다.
- 그리고 $S_{i,1}$ 은 현재 선택된 구역들 중 $S_{i,0}$ 에 포함되지 않는 구역들로 만들 수 있는 집합들 중 비용 합이 가장 큰 집합으로 정의합니다.
- 같은 방법으로 $S_{i,j}$ 는 현재 선택된 구역들 중 $S_{i,0} \cup S_{i,1} \cup \cdots \cup S_{i,j-1}$ 에 포함되지 않는 정점들로 만들 수 있는 집합들 중 비용 합이 가장 큰 집합으로 정의합니다.
- 자식-조상 관계가 아닌 두 정점 u, v 에 대해서 S_u 와 S_v 를 합쳐 새로운 S_w 를 만든다고 하면 $S_{w,i} = S_{u,i} \cup S_{v,i}$ 임을 알 수 있습니다.

- 깊이가 깊은 구역부터 올라가면서 위와 같이 S를 나이브하게 합쳐주면 $\mathcal{O}\left(N^2\right)$에 **서브태스크 3**을 풀 수 있습니다.
- S를 집합이나 우선순위 큐로 관리하고 합치는 과정에서 작은 집합에서 큰 집합으로 합치는 테크닉을 사용하면 $\mathcal{O}\left(N\log^2 N\right)$에 **서브태스크 4**를 풀 수 있습니다.

I. KSA에서 숨바꼭질

`dp, trees`

출제진 의도 – **어려워요**

- 제출 115번, 정답 6명 (정답률 5.217%)
- 처음 푼 사람: **ormlis**, 142분
- 출제자: `mickeyjung`

- KSA를 트리로, 에릭이 숨어있는 건물을 정답 정점으로 정의하겠습니다.
- **서브태스크 1**은 모든 정점의 부모가 1이므로 답은 $N = 2$일 때만 예외적으로 1이고 나머지 경우에 항상 $N - 2$입니다.
- **서브태스크 2**부터는 DP를 사용해야 합니다.
- 가장 첫번째로 리모컨에 x번을 입력해서 d라는 답을 받았다고 가정합시다.
- 일단 트리의 루트를 x로 둡니다.
- 그러면 정답으로 가능한 모든 후보 정점의 깊이가 d로 같아지게 됩니다.
- 따라서 다음 질문을 할 때마다 질문한 정점과 정답 정점의 LCA를 알 수 있습니다.

- $dp[i]$ = i번 정점의 서브트리에 정답 정점이 속한다는 것을 알 때, 정답 정점을 찾기 위한 질문의 최소 횟수 라고 정의합니다.
- $depth[i] = d$일 때, $dp[i] = 0$입니다.
- $depth[i] < d$일 때는 i번 정점의 자식들 중 DP 값이 큰 것부터 차례로 시도해보는 것이 최적입니다.
- 따라서 X를 i번 정점의 자식들의 DP 값을 내림차순 정렬한 배열이라고 한다면 $dp[i] = \min_{1 \le i \le |X|} (X[i] + i - 1)$입니다.
- $ans[i][j]$를 $x = i, d = j$로 놓고 위 DP를 풀었을 때 $dp[i]$값이라고 합시다.
- 정답은 $\min_{1 \le i \le N} \left(\max_{0 \le j < N} ans[i][j] \right) + 1$가 되고, 시간복잡도는 $\mathcal{O}\left(N^3 \log N\right)$입니다.

- **서브태스크 3**은 여기서 조금 더 최적화를 해야합니다.
- $ans[i][j]$ 들을 구할 때 DP 과정에서 사용되는 서브트리들이 겹치게 됩니다.
- 각 서브트리는 하나의 방향이 있는 간선에 대응되므로 서브트리의 개수는 총 $2N-2$개 입니다.
- 각 서브트리에 1 번부터 $2N-2$ 번까지 번호를 매긴 후,
- DP를 다음과 같이 정의합시다.
- $dp[i][j] =$ 정답 정점이 i 번 서브트리에 속하고, 정답 정점과 i 번 서브트리의 루트 사이의 거리가 j 임을 알 때, 정답 정점을 찾기 위한 질문의 최소 횟수

- i 번 서브트리의 높이를 $H[i]$ 라고 하면 $dp[i][j]$ 에서 j 의 범위는 0 이상 $H[i]$ 이하입니다.
- $dp[i][0]$ 부터 $dp[i][H[i]]$ 까지를 묶어서 계산해봅시다.
- 여기서, 받아와야하는 값은 i 번 서브트리의 루트의 자식들의 DP값입니다.
- 이때, 자식들의 높이의 합은 N 이하이므로 받아오는 값이 N 개를 넘지 않는다는 것을 알 수 있습니다.
- 따라서 $dp[i][0], dp[i][1], \cdots, dp[i][H[i]]$ 를 묶어서 계산하는 시간복잡도는 $\mathcal{O}(N \log N)$ 입니다.
- 이를 모든 i 에 대해 해주면 $\mathcal{O}(N^2 \log N)$ 에 문제를 풀 수 있습니다.

J. 마라톤

graphs, dijkstra, dp, cht
출제진 의도 – **어려워요**

- 제출 83번, 정답 6명 (정답률 7.229%)
- 처음 푼 사람: **um_nik**, 86분
- 출제자: mickeyjung

- 마라톤 경기장을 그래프로 보겠습니다. j 번 학생은 간선을 $j-1$ 번 지나야 합니다.
- i 번 정점에 연결된 가장 짧은 간선의 가중치를 S_i 라고 정의하겠습니다.
- 모든 i 에 대해 다음과 같은 경로를 고려해봅시다.
- 1 번 정점에서 출발해서 i 번 정점에 도달 $\rightarrow$ i 번 정점에서 가장 짧은 간선 왕복 여러번 $\rightarrow$ i 번 정점에서 출발해서 N 번 정점에 도달
- 편의상 1 번 정점에서 출발해서 i 번 정점에 도달하는 경로를 A 경로,
- i 번 정점에서 가장 가중치가 작은 간선 왕복 여러번하는 경로를 B 경로,
- i 번 정점에서 출발해서 N 번 정점에 도달하는 경로를 C 경로라고 하겠습니다.
- 이때 A 경로 또는 C 경로의 간선 수가 하나 증가할 때마다 B 경로의 간선 수는 하나 감소하게 됩니다.

- 따라서 어떤 j 번 학생에 대해 답을 구할 때, 모든 간선의 가중치를 (원래 가중치) - S_i 로 두고 A 경로와 C 경로의 최단 거리를 다익스트라 등으로 계산 후, $S_i \times (j-1)$ 을 더함으로써 정답을 구할 수 있습니다.
- 이때 다음과 같은 3 가지를 고려해주어야 합니다.
- 1.
  - 원래 가중치가 S_i 보다 작은 간선은 가중치가 음수가 되므로 지나면 안됩니다.
  - 가중치가 더 작은 간선을 지나는 경우에는 i 번 정점에서 왔다갔다 하는 것보다 그 간선을 왕복하는 것이 항상 이득이기 때문에 그 간선을 지나는 경우를 고려하지 않아도 됩니다.

2.

- B 경로의 간선 수는 항상 짝수이므로 (A 경로의 간선 수 + C 경로의 간선 수)와 $j-1$의 기우성이 서로 다를 때에는 j 번 학생에 대한 답을 구할 수 없습니다
- 따라서 A 경로와 C 경로의 간선 수가 홀수일 때의 최단거리, 짝수일 때의 최단거리를 각각 구해주어야 하는데 이는 한 정점을 홀수번째에 방문하는지 짝수번째에 방문하는지에 따라 2개의 정점으로 분할해서 다익스트라를 돌리면 됩니다.
- $A_{even}, A_{odd}, C_{even}, C_{odd}$를 각각 구했다면
  $j-1$이 홀수일 땐 $\min(A_{odd}+C_{even}, A_{even}+C_{odd})+S_i\times(j-1)$,
  짝수일 땐 $\min(A_{even}+C_{even}, A_{odd}+C_{odd})+S_i\times(j-1)$로 답을 구할 수 있습니다.

✓ 3.

- (A 경로의 간선 수 + C 경로의 간선 수)가 j를 초과하는 경우에는 B 경로에서의 왕복 횟수가 음수가 되어 잘못된 값을 얻을 수 있습니다.
- A 경로의 간선 수는 최대 $2N$이고 C 경로도 마찬가지이므로 (A 경로의 간선 수 + C 경로의 간선 수) $< 4N$입니다.
- 중요하진 않지만 조금 더 생각해보면 이 값이 $2N$까지도 bound 된다는 사실을 알 수 있습니다.
- 따라서 $j < 2N$인 j에 대해서만 따로 답을 구해주면 됩니다.
- 이는 DP나 벨만포드 등으로 구해줄 수 있습니다.

- $j-1$을 x라는 변수로 나타내고 y를 정답이라고 하면 각 i에 대해 다음과 같은 2개의 직선이 만들어집니다.
  - x가 홀수인 경우 : $y = S_i x + \min(A_{even} + C_{even}, A_{odd} + C_{odd})$
  - x가 짝수인 경우 : $y = S_i x + \min(A_{odd} + C_{even}, A_{even} + C_{odd})$
- 따라서 x의 기우성에 따라 Line Container를 2개 만들어주고 각 i에서 만들어진 직선을 넣어주면 x에 대해 CHT를 돌릴 수 있습니다.
- 결론적으로 $j < 2N$인 j들에 대해서는 전처리로 따로 계산해주고, $2N \leq j \leq K$인 j들의 답은 CHT로 구할 수 있습니다.

- 모든 j에 대해 CHT로 답을 구한다면 **서브태스크 2**를 맞힐 수 있습니다.
- **서브태스크 4**를 맞히기 위해서는 직선들의 교점들을 찾아 누적 합 공식을 사용하면 됩니다.
- 총 시간복잡도는 $\mathcal{O}(NM \log N)$입니다.
- 추가로, 다익스트라를 사용하지 않는 $\mathcal{O}(NM)$ 풀이도 존재합니다.