2025 SKKU 프로그래밍 대회 풀이

Official Solutions

by

NPC

2025년 5월 31일

문제		의도한 난이도	출제자
A	Bob부 멍충이	**Easy**	Coxie
B	e-코너 시스템 테스트 (Easy)	**Easy**	shjohw12
C	꽁꽁 얼어붙은 트리	**Medium**	itinbangk
D	DagDag구리	**Medium**	Coxie
E	e-코너 시스템 테스트 (Hard)	**Hard**	shjohw12
F	NPC 현수막 만들기	**Medium**	itinbangk
G	어려운 문자열 문제	**Challenging**	Coxie
H	대도시 구축	**Medium**	itinbangk
I	수열과 병규	**Hard**	itinbangk

A. Bob부 멍충이

ad-hoc

출제진 의도 – **Easy**

- 처음 푼 사람: **7loxo** (+2)
- 출제자: 이승재Coxie

A. Bob부 멍충이

- Alice에게 가장 유리하도록 하려면 배열을 내림차순으로 정렬하면 됩니다.
- A_i 는 서로 다르기 때문에, 가져갈 수 있는 점수는 계속 줄어듭니다.
- 따라서 N 이 짝수라면, 첫 턴에 Alice가 점수를 얻게 되며, Bob은 이 점수를 따라 잡을 수 없습니다. 따라서 Alice가 승리합니다.
- N 이 홀수라면, 첫 턴에 Bob이 점수를 얻게 되며, 자연스럽게 Bob이 승리합니다.

B. e-코너 시스템 테스트 (Easy)

adhoc

출제진 의도 – **Easy**

- 처음 푼 사람: **dbtodqorrn** (+6)
- 출제자: 신정환 `shjohw12`

B. e-코너 시스템 테스트 (Easy)

- 모든 도로의 길이가 같기 때문에 도착지와 가까워지면서 방향을 계속 바꾸는 식으로 이동하면 됩니다.
- $(1,1) \to (1,2) \to (2,2) \to \cdots \to (N-1,N-1) \to (N-1,N) \to (N,N)$ 와 같이 이동하면 됩니다.
- 주행 거리는 $2(N-1)$ 이고, 피봇턴 수행 횟수는 $2(N-1)-1$ 입니다.

C. 꽁꽁 얼어붙은 트리

`bfs/dfs, bruteforce`
출제진 의도 – **Normal**

- ✓ 처음 푼 사람: **seungwoo** (+23)
- ✓ 출제자: 박병규[itinbangk]

C. 꽁꽁 얼어붙은 트리

- $v(d) =$ 제동 지수가 d인 고양이가 루트 노드에서 시작해 멈춰있을 수 있는 노드의 개수
- $l(i) =$ level이 i인 노드의 개수
- 라고 정의합시다. 이 때 $v(d) = \sum_{i=0}^{N-1} l(i)\ (i\%d = 0)$ 입니다.
- N 미만인 $l(i)$의 값들은 한 번의 dfs 혹은 bfs로 $O(N)$만에 구할 수 있습니다.
- 각 $v(d)$는 $\frac{N}{d}$ 회의 연산으로 구할 수 있습니다.

C. 꽁꽁 얼어붙은 트리

- $d = 2$ 부터 $d = N$ 까지 $v(d)$ 를 구하는 총 시간복잡도는

$$\sum_{d=2}^{N} \frac{N}{d} = N \sum_{d=2}^{N} \frac{1}{d} = O(N \log N)$$

  입니다. 여기서 $\sum_{d=1}^{N} \frac{1}{d}$ 꼴의 합은 조화급수(Harmonic series)입니다.

- 총 $O(N \log N)$ 의 시간복잡도로 모든 $v(d)$ 값을 구할 수 있습니다.

D. DagDag구리

`graph`

출제진 의도 – **Medium**

- 처음 푼 사람: **bagger3025** (+106)
- 출제자: 이승재`Coxie`

D. DagDag구리

- 만약 어떤 노드 u의 out degree가 1이면 어떻게 될까요?
- 그럼, 해당 간선을 무조건 사용하게 됩니다.
- 따라서 out degree가 1인 노드에는 반드시 간선을 추가해야 합니다.
- 만약 모든 노드의 out degree가 2 이상이면 어떻게 될까요?
- 운이 나쁘게 두 경로가 한 점에 모이게 되더라도 타고 나갈 수 있는 간선이 두개 이상이기 때문에, 간선을 공유하지 않을 수 있습니다.
- 따라서 주어진 그래프에서 out degree가 1인 노드의 개수가 정답이 됩니다.

E. e-코너 시스템 테스트 (Hard)

`dijkstra`

출제진 의도 – **Medium**

- 처음 푼 사람: **bagger3025** (+118)
- 출제자: 신정환 `shjohw12`

E. e-코너 시스템 테스트 (Hard)

- 최단 거리 문제이기 때문에 기본적으로 다익스트라 알고리즘을 적용하면 됩니다.
- 피봇턴의 경우, 각 격자에 대해 가로로 들어오는 정점/세로로 들어오는 정점을 정의하고 가로→세로 이거나 세로→가로일 때 가중치를 주면 됩니다.
- 최단 거리를 구해야 하므로 pair의 first 가중치는 거리로, second 가중치는 피봇턴으로 하여 피봇턴 수행 시 -1의 가중치로 설정하고 답을 출력할 때 $-$를 붙이는 등의 방법을 적용하면 됩니다.
- 근데 가중치가 음수인데 다익스트라를 적용할 수 있나요? 결론부터 말하면 가능합니다. pair의 first 요소는 양수이고 second 요소만 음수이기 때문에, 같은 정점을 두 번 방문하는 일이 없기 때문에 가중치가 음수일 때 발생하는 시간 복잡도 이슈가 발생하지 않습니다.
- https://www.acmicpc.net/board/view/19865 을 참고하면 공부가 될 것입니다.

F. NPC 현수막 만들기

`bruteforce`, `math`
출제진 의도 – **Medium**

- 처음 푼 사람: **seungwoo** (+76)
- 출제자: 박병규[itinbangk]

F. NPC 현수막 만들기

- ✓ NPC 현수막이 되기 위한 조건들을 정리해봅시다.

1. 한 쌍의 N과 C 사이에 N 또는 C가 들어있으면 안되고, 가운데에는 P가 있어야 합니다.
2. 한 쌍의 N과 C를 포함하는 길이가 d이면, N 앞의 길이와 C 뒤의 길이의 합은 $L-d$ 이상 $R-d$ 이하여야 합니다. 여기도 N이나 C는 들어가면 안됩니다.

- ✓ 조건을 만족하는 쌍을 찾기 위해 input string에서 유용한 정보들만을 추출합시다.

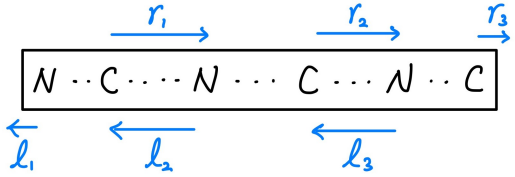
1. N또는 C 문자의 정보를 추출합니다. <char, index> 형태로 추출할 수 있습니다.
2. 이 정보를 순회하면서, N이 왼쪽, C가 오른쪽인 쌍을 찾습니다. N의 index가 l, C의 index가 r일 때 $(l+r)\%2=0$인지, 가운데 문자가 P인지 확인합니다.

F. NPC 현수막 만들기

- 추출한 정보를 통해 N 이전에 처음으로 나오는 N 또는 C의 index, C 이후에 처음으로 나오는 N 또는 C의 index도 쉽게 찾을 수 있습니다.
- $l_i =$ (N의 index - N 이전에 처음으로 나오는 N 또는 C의 index - 1)
- $r_i =$ (C 이후에 처음으로 나오는 N 또는 C의 index - C의 index - 1) 라고 할 때,
- naive하게 찾으면 $\sum l_i \times r_i$ 로, 최대 N^2 개의 쌍이 나와 시간초과가 발생합니다.

F. NPC 현수막 만들기

- 다만 우리는 아래 그림처럼, $\sum l_i$ 혹은 $\sum r_i$ 는 최대 N 임을 알 수 있습니다.
- 각 쌍마다 0 부터 $l_i - 1$ 까지의 l_p 에 대해
  $max(0, min(r_i, R - (l_p + len)) - max(0, L - (l_p + len)) + 1)$ 값을 구해주면 $O(N)$ 에 풀 수 있습니다.



G. 어려운 문자열 문제

`string`, `suffix array`, `segment tree`
출제진 의도 – **challenging**

- ✓ 처음 푼 사람: X
- ✓ 출제자: 이승재 `Coxie`

G. 어려운 문자열 문제

- S에서 위치 i를 중심으로 하는 팰린드롬의 최대 반지름을 r이라고 해봅시다.
- $[i-r, i+r]$ 이 팰린드롬 일텐데, 이 사이에서 문자를 제거하는것은 greedy하지 않습니다.
- 왜냐하면, 우리가 이미 만들 수 있는 팰린드롬을 굳이 제거하여, 이어붙이는 쪽에서 동일한 내용을 만들어야 하기 때문입니다.
- 따라서 $i+r+1$ 부터 제거하거나, $i-r-1$ 앞쪽을 제거해야 합니다.

G. 어려운 문자열 문제

- 이 두 case는 동일하기 때문에, $i + r + 1$을 제거하는 것을 생각해 봅시다.
- $i + r + 1$ 이상의 index에서 시작하는 부분 문자열 중 $i - r - 1$ 쪽과 가장 많이 일치하는 것을 찾으면 됩니다.
- 일단 T = reversed(S) + '$' + S를 만들어 줍니다.
- 이제 T의 $N - (i - r - 1) + 1$번째 위치에서 시작하는 suffix와 가장 길게 일치하는 $N + 1 + (i + r + 1)$ 이상에서 시작하는 suffix를 찾으면 됩니다.
- 이를 구하기 위해 T로 suffix array를 구합니다.
- suffix array상에서 $rank[N - (i - r - 1) + 1]$ 위치와 가장 가깝게 위치하는 $N + 1 + (i + r + 1)$ 이상의 값을 확인해주면 됩니다.
- 이러한 위치는 $rank[N - (i - r - 1) + 1]$ 보다 작은 위치일수도, 큰 위치 일수도 있으며 둘 다 확인해주면 됩니다.

H. 대도시 구축

`graph_theory`, `greedy`, `case_work`

출제진 의도 – **Medium**

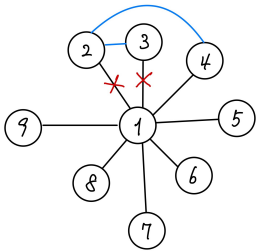
- 처음 푼 사람: **urin6695** (+157)
- 출제자: 박병규[itinbangk]

H. 대도시 구축

- $M = 0$일 때를 생각해볼까요? 이 때의 최소 비용은 1번 마을에 나머지 마을들을 모두 연결하면 최적임을 알 수 있습니다.
- 이 때 정답은 $\frac{N(N+1)}{2} + (N-2)$입니다.
- 여기서 추가로 고려해야할 점은, 강이 1번 마을와 다른 마을을 연결하는 경우의 대체 방안입니다.
- $M = 1$일 때, 강의 정보를 v(0)이라 합시다.
  1. v(0) = {1,2}이면 {2,3}으로 대체 (+2)
  2. v(0) = {1,x}이면 {2,x}으로 대체 (+1)

H. 대도시 구축

- ✓ $M = 2$일 때, pair 기준 정렬 해줍니다. 각 강의 정보를 v(0), v(1)이라 합시다.
- ✓ v(0) = {1,2} 일 경우 {2,3}으로 대체(+2)하지만, v(1)이 아래 경우라면 따로 핸들해줍니다.
  1. v(1) = {1,3}이면 {2,3}, {2,4}으로 대체 (+4)
  2. v(1) = {2,3}이면 {2,4}으로 대체 (+3)
  3. v(1) = {1,x} (x>3)이면 {2,3}, {2,x}으로 대체 (+3)



H. 대도시 구축

- ✓ v(0) = {1,3} 일 경우 {2,3}으로 대체(+1)하지만, v(1)이 아래 경우라면 따로 핸들해줍니다.
  1. v(1) = {2,3}이면 {3,4}으로 대체 (+3)
  2. v(1) = {1,x} (x>3)이면 {2,3}, {2,x}으로 대체 (+2)
- ✓ v(0) = {1,x} (x>3) 일 경우 {2,x}으로 대체(+1)하지만, v(1)이 아래 경우라면 따로 핸들해줍니다.
  1. v(1) = {2,x}이면 {3,x}으로 대체 (+2)
  2. v(1) = {1,y} (y>x)이면 {2,3}, {2,x}으로 대체 (+2)

I. 수열과 병규

`math, recursion`

출제진 의도 – **Hard**

- 처음 푼 사람: **dbtodqorrn** (+194)
- 출제자: 박병규[itinbangk]

- 우리의 목표는 S_n 을 빠르게 계산하는 것입니다. 먼저 3개의 식을 다 더해볼까요?
- 좌변과 우변을 모두 더해주면 $a_{3n} + a_{3n+1} + a_{3n+2} = 2a_n + 2$와 같이 깔끔하게 정리됩니다.
- $a_1 + a_2 = 3$이므로 $S_{3n+2} = 2S_n + 2n + 3$가 만족하고, 이는 $O(log_3 n)$에 구할 수 있습니다.
- 하지만 우리는 $n\%3 = 1, n\%3 = 0$인 경우에도 $O(log_3 n)$에 구할 수 있어야 합니다.

- $S_{3n+2} = 2S_n + 2n + 3$ 에 3번째 식인 $a_{3n+2} = -2a_{n+1} - 2a_{n+2} + 2$ 를 적용하면,
- $S_{3n+1} = 2S_{n+2} + 2n + 1$ 가 만족하고, 이는 $O(log_3 n)$ 에 구할 수 있습니다.
- $S_{3n+1} = 2S_{n+2} + 2n + 1$ 에 2번째 식인 $a_{3n+1} = 2a_{n+2} - 1$ 를 적용하면,
- $S_{3n} = 2S_{n+1} + 2n + 2$ 가 만족하고, 이는 $O(log_3 n)$ 에 구할 수 있습니다.
- 모든 자연수 n 에 대해 $O(log_3 n)$ 만에 S_n 을 구할 수 있으므로 총 시간복잡도 $O(Qlog_3 n)$ 에 문제를 풀 수 있습니다.