PNUPC 2025

by

PNU-PULSE

2024년 05월 10일

운영

- qvixnh22
- angel5337
- corntofu
- mbae059

출제

- qvixnh22
- corntofu
- mbae059

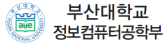
검수

- chika
- dongwook7
- jthis
- ksoosung77
- kwoncycle
- martin0327
- minjunnick
- tony9402
- utilforever
- wizardrabbit
- yyyy7089

주최

- PULSE

후원

- 부산대학교 정보컴퓨터공학부 

- 부산대학교 소프트웨어중심대학사업단

- 스타트링크(Startlink)

후원

- CERabi
- mbae059
- pjshwa
- rhdqor213
- toycartoon
- yyyy7089

문제	의도한 난이도	출제자
PNUPC에 한번도 빠지지 않고 출연한 산지니가 새삼 대단하다고 느껴지네	Easy	qvixnh22
삽입 정렬을 해볼까	Easy	mbae059
태종대 낚시 맛집	Easy	qvixnh22
퍼시스턴트 스택	Medium	qvixnh22
정말 간단한 문제	Hard	corntofu
Chill...은 내가 가장 좋아하는 소수	Hard	qvixnh22
dag lca	Medium	corntofu
pnupc 1K9	Hard	corntofu
행렬과 쿼리	Hard	qvixnh22

2A. PNUPC에 한 번도 빠지지 않고 출연한 산지니가 새삼 대단하다고 느껴지네

`string`, `implement`

출제진 의도 – **Easy**

- 제출 56번, 정답 33명 (정답률 58.929%)
- 처음 푼 사람: **stainless1028**, 1분
- 출제자: `qvixnh22`

2A. PNUPC에 한 번도 빠지지 않고 출연한 산지니가 새삼 대단하다고 느껴지네

- ✓ 두 문자열을 입력받아 앞 문자열에 있는 문자를 뒷 문자열에서 제거하면 됩니다.
- ✓ 구현하는 여러 방법이 있지만 세 가지 방법을 소개합니다.

2A. PNUPC에 한 번도 빠지지 않고 출연한 산지니가 새삼 대단하다고 느껴지네

- ✓ 앞 문자열을 하나씩 살펴보면서 뒷 문자열에 해당 문자가 있으면 $O(|T|)$ 에 삭제하면 $O(|S||T|)$ 에 해결할 수 있습니다.
- ✓ 뒷 문자열을 하나씩 살펴보면서 앞 문자열에 해당 문자가 등장하는지 $O(|S|)$ 에 살펴보면 $O(|S||T|)$ 에 해결할 수 있습니다.
- ✓ 앞 문자열에 나오는 문자와 나오지 않는 문자를 전처리하면 $O(|S|+|T|)$ 에 해결할 수 있습니다.

1A/2B. 삽입 정렬을 해볼까?

`sorting`
출제진 의도 – **Easy**

- 제출 172번, 정답 59명 (정답률 34.302%)
- 처음 푼 사람: **6729skl**, 2분
- 출제자: `mbae059`

1A/2B. 삽입 정렬을 해볼까?

- 삽입 정렬에 의해 이동되는 원소보다 작은 인덱스의 원소들은 항상 정렬되어있습니다.
- t 번째로 정렬되는 원소는 t 번째 원소이고, 그 앞의 원소들은 항상 정렬되어있습니다.
- 따라서 처음 t 개의 원소만 정렬하고 출력하면 됩니다.

2C. 태종대 낚시맛집

`brute_force, simulation`
출제진 의도 – **Normal**

- 제출 129번, 정답 10명 (정답률 7.752%)
- 처음 푼 사람: **cnvxlns**, 27분
- 출제자: `qvixnh22`

2C. 태종대 낚시맛집

- 사용할 떡밥의 개수가 정해져 있을 때 얻을 수 있는 이익을 구하는 것은 좀 더 쉽습니다.
- 모든 물고기를 크기별로 정렬한 뒤 큰 물고기부터 유인될 지 결정해주면 됩니다.
- 물고기의 수 $\times$ 최대 먹성 만큼의 떡밥을 사용하면 모든 물고기를 낚을 수 있고, 그 이상 사용하는 것은 항상 최적해가 아님을 알 수 있습니다.
- 따라서 1 부터 물고기의 수 $\times$ 최대 먹성까지 모든 경우에 대해 얻을 수 있는 이익을 계산한 뒤 그중 최대값을 출력하면 됩니다.

- 실제로 사용한 떡밥의 수에 대해 얻을 수 있는 이익의 그래프를 그려보면 monotone하지도, unimodal하지도 않습니다.
- 따라서 이분 탐색이나 삼분 탐색을 사용하면 틀린 풀이입니다.

1B. 퍼시스턴트 스택

`implements, simulation`
출제진 의도 – **Normal**

- ✓ 제출 105번, 정답 25명 (정답률 24.762%)
- ✓ 처음 푼 사람: **6729skl**, 12분
- ✓ 출제자: `qvixnh22`

- 1번 쿼리나 2번 쿼리가 주어질 때 스택을 복사해서 저장하기에는 복사 비용이 너무 크므로 스택을 복사하는 대신 쿼리의 내용을 순서대로 저장해야 합니다.
- 3번 쿼리가 주어지면 순서대로 저장한 1번 또는 2번 쿼리를 거꾸로 실행하면 됩니다.
- 모든 1번 쿼리와 2번 쿼리는 한 번 실행되고 많아야 한 번 3번 쿼리에 의해 취소되므로 $O(Q)$에 모든 쿼리를 처리할 수 있습니다.

1C/2D. 정말 간단한 문제

`math`
출제진 의도 – **Hard**

- 제출 182번, 정답 3명 (정답률 1.648%)
- 처음 푼 사람: **바람을 향해**, 28분
- 출제자: `corntofu`

- 두 가지 사실을 관찰해봅시다.
- 1. 분모와 분자가 모두 양의 정수인 유리수 $\frac{b}{a}, \frac{d}{c}$에 대해 $\min(\frac{b}{a}, \frac{d}{c}) \leq \frac{b+d}{a+c} \leq \max(\frac{b}{a}, \frac{d}{c})$ 입니다.
- 2. 위의 식에서 등호가 성립하는 조건은 $\frac{b}{a} = \frac{d}{c}$가 성립할 때이다.

1C/2D. 정말 간단한 문제

- $1.$에 의해 길이가 1인 최댓값은 항상 존재하므로 $O(N)$으로 길이가 1인 최댓값을 찾은 후, $2.$에 의해 최댓값으로만 이루어진 구간의 길이를 구해주면 됩니다.
- 시간복잡도는 $O(N)$입니다.

2E. Chill... 은 내가 가장 좋아하는 소수

`dp, sieve`
출제진 의도 – **Hard**

- 제출 60번, 정답 4명 (정답률 6.667%)
- 처음 푼 사람: **응애**, 106분
- 출제자: `qvixnh22`

2E. Chill...은 내가 가장 좋아하는 소수

- 모든 타일의 배치를 살펴볼 순 없습니다. 따라서 모든 타일의 배치를 살펴보지 않고 모든 경우의 수를 다 본 것과 같은 효과를 내야 합니다.
- 가장 오른쪽의 두 칸을 타일로 덮는 경우는 두 가지입니다.
- 나머지 칸을 어떻게 덮었는 지와 관계없이 두 칸을 타일로 덮는 점수가 결정됩니다.
- 이는 소수 타일링 문제의 부분문제이고 dp로 해결할 수 있습니다.

2E. Chill...은 내가 가장 좋아하는 소수

- 어떤 경우에 a 점을 받고 어떤 경우에 b 점을 받는지 결정해야 합니다.
- 에라토스테네스의 체를 이용해 모든 가능한 경우에 대해 소수인지 결정해두면 빠르게 판단할 수 있습니다.

1D. DAG LCA

`bfs`

출제진 의도 – **Medium**

- 제출 39번, 정답 8명 (정답률 20.513%)
- 처음 푼 사람: **pppqqq**, 69분
- 출제자: `corntofu`, `mbae059`

- 임의의 정점 u, v에 대해 u에서 v로 가는 최단 경로의 길이를 저장하는 $N \times N$ 배열을 설정한 후, 모든 정점을 시작점으로 하는 너비 우선 탐색을 시행해서 배열을 채웁니다.
- 배열을 채웠다면 각 질의에 대해서 모든 정점을 둘러보는 방법으로 $O(N)$의 시간복잡도에 답할 수 있습니다.
- 시간복잡도는 너비 우선 탐색을 N회 시행하고 질의를 Q번 답하므로 $O(N(N+M+Q))$입니다.

1E/2F. PNUPC 1K9

`brute_force`, `ad_hoc`
출제진 의도 – **Hard**

- 제출 27번, 정답 1명 (정답률 3.703%)
- 처음 푼 사람: **바람을 향해**, 96분
- 출제자: `corntofu`

- 주어진 수는 36진수이고 $p < 1000$ 이므로 s가 36보다 크거나 같은 경우엔 가장 아래의 두 자리만 적절하게 변경해도 조건을 만족시킬 수 있습니다.
- 답은 0, 1, 2 중 하나이므로, 브루트포스로 한자리씩 변경해주며 조건을 만족할 경우 1이 정답, 아니면 2가 정답이 됩니다.
- s가 한 자리일 때 -1이 정답이 되는 경우에 유의해야 합니다.
- 시간복잡도는 $O(|S|)$ 입니다.

1F. 행렬과 쿼리

`segtree, implement`
출제진 의도 – **Hard**

- 제출 0번, 정답 0명 (정답률 00.000%)
- 처음 푼 사람: , 0분
- 출제자: `qvixnh22`

- 행렬의 곱셈은 결합법칙이 성립합니다. 따라서 l 번째부터 m 번째 행렬의 곱과 $m+1$ 번째부터 r 번째 행렬의 곱을 알고 있으면 이 두 행렬을 곱하면 l 번째부터 r 번째 행렬의 곱을 쉽게 구할 수 있습니다.
- 이 행렬 곱 전처리를 구간 크기의 절반 단위로 미리 구해두면 총 $O(N)$ 개의 행렬 전처리를 하게 되고 한 번 업데이트될 때마다 $O(lgN)$ 개의 행렬만 업데이트하면 됩니다.
- 1번 쿼리마다 전처리된 구간 중 가장 긴 구간들의 총 길이는 $\frac{r-l}{2}$ 보다 큽니다. 따라서 $O(lgN)$ 만에 행렬곱을 구할 수 있습니다.
- 따라서 총 $O(QlgN)$ 만에 전체 질의를 수행할 수 있습니다. 이 방법을 세그먼트 트리라고 부릅니다.

- 행렬을 1번 쿼리가 주어질 때마다 순서대로 곱하면 시간초과를 받을 수 있습니다.
- 행렬 곱셈은 교환법칙이 성립하지 않습니다. 흔히 알려진 세그먼트 트리의 비재귀 구현 중 교환법칙이 가능한 경우에만 쓸 수 있는 구현을 사용하면 틀립니다.
- 다양한 크기의 행렬 곱셈에는 항등원인 행렬이 존재하지 않습니다. 따라서 이를 전처리하지 않고 사용한다면 틀릴 수 있습니다.