단계별 모의고사 1회차 해설

문제	
1	전구
2	소수 최수 공배수
3	서로소 평균
4	블로그
5	학부 연구생 민상
6	곡예 비행
7	도시 건설
8	짝수 팰린드롬

1. 전구

1. 전구

- ✓ 주어지는 명령어에 따라서 “구현”하면 되는 문제입니다.
- ✓ 2번 명령어 같은 경우 `if`를 이용하여 코드를 작성해도 되지만 `bitwise xor`를 이용하면 `if` 문을 사용하지 않고도 간단하게 구현할 수 있습니다.
  - Java(27 line) : Java Solution
  - C++(20 line) : C++ Solution
  - Python(16 line) : Python Solution
  - Python(15 line) : Python Solution
- ✓ 시간복잡도 $O(NM)$

2. 소수 최소 공배수

2. 소수 최소 공배수

- ✓ 주어지는 N 개의 수들 중에서 소수인 것들을 골라내고 그 수들의 최소 공배수를 구해서 출력하고 만약 소수인 수가 없다면 -1 을 출력하면 되는 문제입니다.
- ✓ 소수인지 판별할 때 $O(\sqrt{N})$ 걸리는 알고리즘을 사용해도 시간안에 돌아갈 수 있습니다.
  - $O(\sqrt{N})$ 만큼 걸리는 알고리즘을 이용하여 이 문제를 해결할 때 시간복잡도는 $O(N\sqrt{A_i})$
  - 따라서, 해당 알고리즘을 이용하여 주어지는 모든 수가 소수인지 판별해도 시간복잡도상 문제 없는 것을 확인할 수 있습니다.
- ✓ A_i 가 최대 10^6 이라 **에라토스테네스의 체**를 이용하여 소수인지 판별 가능합니다.
  - 에라토스테네스의 체는 $O(NloglogN)$ 만에 N 이하의 수들 중 소수인 것을 찾을 수 있습니다.

2. 소수 최소 공배수

- 주어지는 수들 중 소수인 수를 다 찾았다면 아래 두 가지 방식 중 하나를 이용하여 최소 공배수를 구하면 됩니다.
- 첫 번째 방법으로는, LCM 알고리즘을 이용하여 최소 공배수를 구해도 됩니다.
- 두 번째 방법으로는, 중복된 수를 제거하여 그 수들을 모두 곱해도 됩니다. (모든 수는 소수이기 때문)

2. 소수 최소 공배수

- 따라서, 이 문제를 푸는 방법은 아래 두 단계를 걸쳐 답을 찾으시면 됩니다.
- 1. 주어지는 수들 중 소수를 찾는다. (만약, 소수인 수가 없다면 -1을 출력하고 프로그램을 종료한다.)
- 2. 찾은 소수들을 이용하여 최소 공배수를 계산한다.

3. 서로소 평균

3. 서로소 평균

- A와 B가 서로소이라는 것은 A와 B의 최대공약수의 값이 1을 의미합니다.
- 주어지는 수(A_i)와 X의 GCD 값이 1인 A_i 수를 골라낸 후 그 수들의 평균을 구하면 되는 문제입니다.
- 두 수의 최대공약수(GCD)를 구하는 시간복잡도는 $O(\log N)$으로 해당 문제를 풀기 위한 솔루션의 시간복잡도는 $O(N\log A_i + N)$ 입니다.

4. 블로그

4. 블로그

- 임의의 X 일 동안 연속된 날을 잡았을 때 가장 방문자의 수이어야 하고 그러한 구간이 여러 개가 존재한다면 이를 다 알아하는 문제입니다.
- 시간제한을 생각하지 않고 가장 쉬운 방법은 맨 앞에서부터 X 일을 보고 방문자 수를 계산하고 이전까지 계산한 방문자 수보다 크다면 갱신하는 방식으로 진행하면 됩니다.
- 하지만, 이 풀이에 대한 시간복잡도를 구해본다면 X 일을 보는 개수는 $N - X$ 개이고 X 일 안에 방문자 수의 합을 구해야하기 때문에 $O((N - X)X)$ 가 걸립니다.
- $1 \leq X \leq N \leq 250\,000$ 이기 때문에 위 시간복잡도에서 약 1.6×10^9 의 연산을 하게 되고 이는 1초 안에 모두 계산을 할 수 없는 양입니다.

- 이는 슬라이딩 윈도우라는 기법을 이용하면 시간복잡도 $O(N)$ 만에 계산을 할 수 있습니다.
- 맨 처음에 연속된 X 일 구간에서 방문한 수의 합을 계산하고 다음 구간을 계산할 때 새로 추가된 날짜의 방문자 수를 더하고 가장 오래된 날짜의 방문자 수를 빼면 $O(1)$ 만에 계산할 수 있습니다.
- 처음에 X 일만큼 먼저 계산을 하고 $N - X$ 개의 구간을 봐야하기 때문에 시간복잡도는 $O(X + N - X)$ 으로 $O(N)$ 가 됩니다.

5. 학부 연구생 민상

5. 학부 연구생 민상

- ✓ 에어컨에서 시원한 바람이 상하좌우 사방으로 나가면서 장애물(물건)에 의해 바람의 진행방향이 변경될 때, 바람이 도달할 수 있는 곳들을 체크하는 문제입니다. 이때, 에어컨이 있는 칸과 장애물(물건)이 있는 곳도 개수에 포함해야한다는 사실을 잊으면 안됩니다.
- ✓ 이 문제는 격자 모양의 그래프에서의 탐색을 하는 문제로 그래프 모델링을 잘 해야합니다.
- ✓ 격자 탐색할 때 아래와 같은 상태를 정의한다면 답은 맞게 구하지만 시간초과가 뜨거나 올바른 답을 계산할 수 없습니다.
  - $visited[x][y]$: 격자 (x, y)에 방문했는지 여부(0: 미방문, 1: 방문)를 체크하는 배열

5. 학부 연구생 민상

- ✓ 올바르게 계산을 하기 위해서는 아래와 같이 정의를 해야합니다. (2차원 배열로 두고 비트 연산을 이용해도 됩니다.)
  - $visited[x][y][dir]$: 격자 (x, y)에 dir 방향으로 방문했는지 여부(0: 미방문, 1: 방문)를 체크하는 배열
- ✓ 위와 같이 상태를 정의했다면 그 다음은 지문에서 주어진대로 구현을 진행하면 됩니다.
- ✓ 방향(dir)은 4가지고 격자는 $N * M$ 이기 때문에 모든 상태를 방문하기 위해서는 $O(4NM)$ 이 걸립니다.

6. 곡예 비행

6. 곡예 비행

- 해당 문제는 상승 비행의 경로와 하강 비행의 경로를 적절히 구해 최대 점수를 계산하는 문제입니다.
- 이 문제는 다이나믹 프로그래밍을 이용하여 풀 수 있습니다.
- 주어지는 모든 점수가 음수라면 점수는 음수일 수 있음을 조심해야 합니다.
- 또한, 받을 수 있는 점수들의 최소 점수, 최대 점수를 구해보면 방문하는 격자의 최대 개수는 약 $3\,000$ 개이기 때문에 최소 점수는 -3×10^7, 최대 점수는 3×10^7 로 32bit integer로 계산해도 문제 없다는 것을 알 수 있습니다.

6. 곡예 비행

- 점화식 정의를 해야하는데 두 가지 방법이 있습니다. 그 중 첫 번째 점화식을 세우면 아래와 같습니다.
- DP$[x][y][state]$: 모형 비행기가 격자 (x, y)에서 $state$(0: 상승, 1: 하강)
- DP$[x][y][0]$ = max(DP$[x-1][y][0]$, DP$[x][y+1][0]$) + score$[x][y]$
- DP$[x][y][1]$ = max(DP$[x][y][0]$, DP$[x-1][y][1]$, DP$[x][y-1][1]$) + score$[x][y]$
- 이런식으로 점화식을 세우면 문제를 풀 수 있지만, 조금만 생각해보면 더 간단하게 풀 수 있습니다.

6. 곡예 비행

- 구해야하는 값을 다시 정리해보면 모형 비행기가 출발하는 시작점에서 상승 비행와 상승 비행이 끝난 지점에서 도착점까지 하강 비행시 얻을 수 있는 점수를 구하는 문제입니다.
- 이는 “1. 시작점에서 상승 비행시 얻을 수 있는 점수”와 “2. 상승 비행이 끝난 지점에서 도착점까지 하강 비행시 얻을 수 있는 점수”로 나눌 수 있습니다.
- 두 가지 값을 구하는 것은 독립적이기 때문에 두 번째 경우를 뒤집어보면 “도착점에서 상승 비행시 얻을 수 있는 점수”로 변경할 수 있습니다.

6. 곡예 비행

- 상승 비행시 얻을 수 있는 점수를 구하는 점화식을 세우면 다음과 같습니다.
  - 시작점에서 상승비행: DP1$[x][y]$ = max(DP1$[x-1][y]$, DP1$[x][y+1]$) + score$[x][y]$
  - 도착점에서 상승비행: DP2$[x][y]$ = max(DP2$[x+1][y]$, DP2$[x][y+1]$) + score$[x][y]$
- DP1와 DP2 테이블을 채운 후 임의의 (x,y)에서 상승, 하강 비행시 얻을 수 있는 점수는 DP1$[x][y]$ + DP2$[x][y]$와 같습니다.
- 시간복잡도 : $O(NM)$

7. 도시 건설

7. 도시 건설

- 모든 건물들이 연결될 수 있도록 하면서 몇 개의 도로를 없애서 얼마나 예산을 아낄 수 있는지 물어보는 문제입니다.
- 이는 최소 스패닝 트리(MST)를 구하는 알고리즘을 이용하면 쉽게 풀 수 있습니다.
- 주어지는 도로를 이용하여 MST를 구할 때 선택된 도로의 개수가 건물의 개수 $-\ 1$가 아니라면 -1을 출력하면 됩니다.
- MST를 구하는 알고리즘은 흔히 사용하는 프림(Prim) 알고리즘, 크루스칼(Kruskal) 알고리즘 중에 사용하면 됩니다.

8. 짝수 팰린드롬

8. 짝수 팰린드롬

- ✓ 주어지는 수열을 길이가 짝수이고 팰린드롬인 연속된 부분 수열을 최대로 나눌 때 그 개수가 몇 개인지 구하는 문제입니다.
- ✓ 수열을 완전히 나눌 수 없을 경우는 -1을 출력하면 됩니다.
- ✓ 이 문제는 $O(N^2)$ 시간에 임의의 [l, r] 구간이 팰린드롬인지 계산하고 위 조건대로 나눌 수 있는 여부를 $O(N^2)$ 만에 계산할 수 있기도 합니다.
- ✓ 하지만, 조금만 관찰을 해보면 그리디하게 문제를 해결할 수 있습니다.
  1. 주어지는 수열을 완전히 나눠야합니다.
  2. 나뉜 연속된 부분 수열이 반드시 길이가 짝수인 팰린드롬이어야 합니다.
- ✓ 2번 조건에 의해 앞에서부터 최대한 짤라보면서 확인해볼 수 있습니다.
- ✓ 해당 방식으로 $O(N^2)$ 만에 구할 수 있습니다.