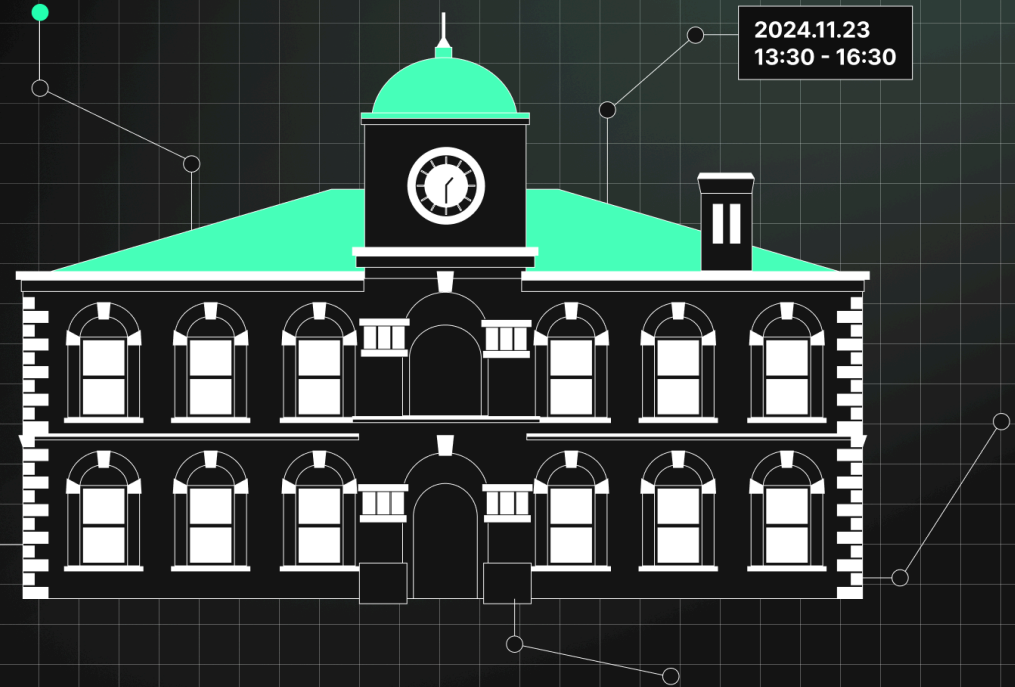


# 2024 건국대학교 프로그래밍 경진대회 Official Solution





문제

의도한 난이도

출제자

---

A	구구단표	Beginner	권순호
B	홀수로 나뉘라! 짝수로 나뉘라!	Easy	윤찬규
C	다이얼 룰렛	Normal	윤찬규
D	숫자 POP	Normal	윤찬규
E	방벽 게임	Normal	권순호
F	십자 찾기	Normal	윤찬규
G	방공호 만들기	Normal	윤찬규
H	격자 연결하기	Normal	윤찬규
I	우선순위 큐와 시뮬레이션	Normal	권순호
J	수열 만들기	Normal	권순호
K	순열과 증가수열	Hard	이승엽
L	완벽한 도시 설계	Challenging	이승엽


# A. 구구단표

bruteforce

출제진 의도 - **Beginner**

 제출 65회, 정답 22명 (정답률 33.85%)

 likescape, 1분

 권순호 yookwi

## A. 구구단표

- ✓ 모든 경우를 살펴봅시다.
- ✓  $2 \leq i \leq 9, 2 \leq j \leq 9$ 을 만족하는 모든  $(i, j)$ 에 대해  $i, j, i \times j$  중 주어진 수가 있는지 판단하면 됩니다.

## B. 홀수로 나눠라! 짝수로 나눠라!

greedy

출제진 의도 - **Easy**

 제출 62회, 정답 14명 (정답률 22.58%)

 codingstarfish, 9분

 윤찬규 dldyou

## B. 홀수로 나눠라! 짝수로 나눠라!

- ✓ 홀수로 나누기와 짝수로 나누기를 사용할 수 있을 때는 각각 언제일까요?
- ✓ 수열을 **1개**의 집합으로 나누어 봅시다.
- ✓ 집합을 이루는 원소의 합이 홀수라면, 홀수로 나누기를 사용하면 됩니다!
- ✓ 집합을 이루는 원소의 합이 짝수라면, 그 어느 방법으로도 수열을 나눌 수 없습니다.
- ✓ 수열을 **2개**의 집합으로 나누어 봅시다.
- ✓ 수열의 앞에서부터 집합을 2개로 나눠보며, 조건을 만족하는지 체크합니다.
- ✓ 각각 원소의 합이 짝수인 집합으로 쪼개지면, 짝수로 나누기를 사용할 수 있습니다.


## B. 홀수로 나눠라! 짝수로 나눠라!

- ✓ 모든 경우에서 위의 2가지 방법으로 구할 수 있나요?
- ✓ 홀수로 나누기를 사용하여 홀수 개의 집합으로 나눴다고 해봅시다.
- ✓ 집합의 개수와 각각의 원소의 합이 홀수이므로, 전체 원소의 합 또한 홀수가 됩니다.
- ✓ 결국 나눈 것들을 합쳐 1개의 집합으로 나눈 것으로 만들 수 있습니다.
- ✓ 짝수로 나누기를 사용하는 경우 또한 2개의 집합으로 나눈 것으로 합칠 수 있습니다.

## C. 다이얼 룰렛

bruteforce, greedy

출제진 의도 - **Normal**

 제출 23회, 정답 7명 (정답률 30.43%)

 codingstarfish, 23분

 윤찬규 dldyou



## C. 다이얼 룰렛

- ✓ 어떤 큰 원소를 잡고 시계와 반시계 방향으로 회전을 반복하는 것이 이득입니다.
- ✓  $A_t$ 에서 이를 반복할 때, 얻을 수 있는 점수는  $\sum_{i=1}^t A_i + (k - t) \times A_t$ 입니다.
- ✓ 반대로 돌려  $A_t$ 까지 갈 수도 있습니다.
- ✓ 이때 얻을 수 있는 점수는  $\sum_{i=t}^N A_i + (k - (N - t + 1)) \times A_i$ 입니다.
- ✓ 모든  $t$ 에 대해 최댓값을 구하면 됩니다.

## D. 숫자 POP

bruteforce

출제진 의도 - **Normal**

 제출 34회, 정답 12명 (정답률 35.29%)

 codingstarfish, 31분

 윤찬규 dldyou


## D. 숫자 POP

- ✓ 일일이 지워가며 확인하는 간단한 방법부터 봅시다.
- ✓  $A_i$ 에서 시작하여  $A_{i+t}$ 가  $A_i$ 와 다르다면  $A_{i+t}$ 를 지워가며 이어갑니다.
- ✓  $A_i$ 로 시작하는 연속 길이  $L$ 은  $\mathcal{O}(N)$ 에 구할 수 있습니다.
- ✓ 이를 모든 원소에 대해 진행하여 최댓값을 구하는 데  $\mathcal{O}(N^2)$ 이 걸립니다.
- ✓  $N$ 의 제한이 5,000 이하이므로 위의 방법으로 문제를 해결할 수 있습니다.
  
- ✓ 각 원소별 인덱스를 저장하여 이분탐색을 진행하여 구해줄 수도 있습니다.
- ✓ 이 방법의 경우  $\mathcal{O}(N \log N)$ 의 시간복잡도를 가집니다.

# E. 방벽 게임

ad\_hoc

출제진 의도 - **Normal**

 제출 18회, 정답 8명 (정답률 44.44%)

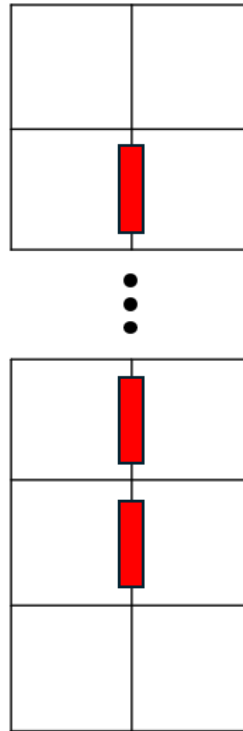
 wariatros, 31분

 권순호 yookwi

## E. 방벽 게임

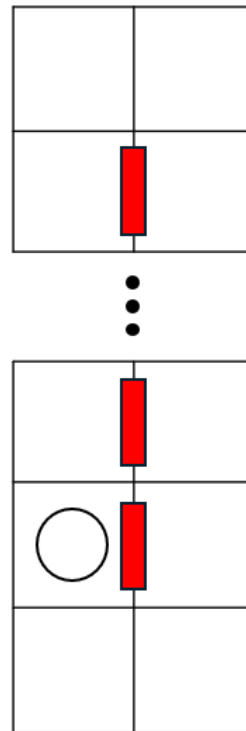
- ✓ 방벽을 설치하는 것부터 생각해봅시다.
- ✓ 최대한 말이 멀리 돌아가도록 만들어야 합니다.

## E. 방벽 게임



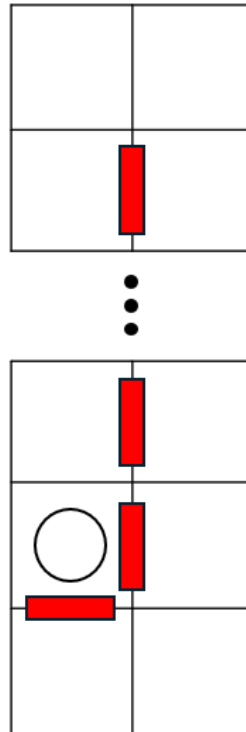
✓ 위와 같이 1행과  $N$ 행을 제외한 모든 행의 가운데에 방벽을 설치합니다.

## E. 방벽 게임



✓ 말이  $N - 1$ 행에 도달했을 때...

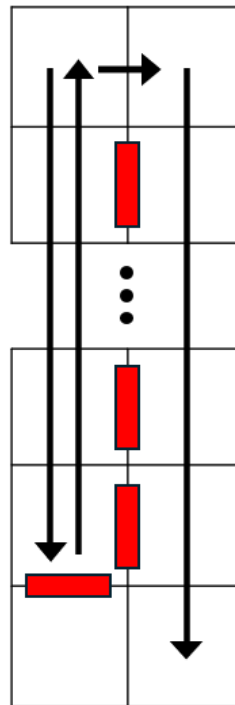
## E. 방벽 게임



✓ 말의 앞을 막아줍니다.



## E. 방벽 게임



- ✓ 말이 최적으로 이동하더라도 화살표 경로처럼 돌아가야 합니다.


## E. 방벽 게임

- ✓ 수식으로 정리하면  $3N - 4$ 번 이동해야합니다.
- ✓  $N = 2, 3$ 일 때는 예외적으로 구해줍니다.

# F. 십자 찾기

prefix\_sum

출제진 의도 - **Normal**

 제출 46회, 정답 7명 (정답률 15.22%)

 codingstarfish, 52분

 윤찬규 dldyou

## F. 십자 찾기

- ✓  $(r, c)$ 를 기준으로 크기가  $K$ 인 십자를 일일이 확인하는 방법은  $\mathcal{O}(K)$ 입니다.
- ✓  $N \times M$ 에 대해 모두 확인해야 하니  $\mathcal{O}(NMK)$ 로 시간초과가 납니다.
- ✓ 크기가  $K$ 인 십자를 빨리 확인할 수 있는 방법이 없을까요?


## F. 십자 찾기

- ✓ 누적합을 이용해 봅시다.
- ✓ 크기가  $K$ 라는 뜻은  $(r, c)$ 를 기준으로  $(r - K, c)$ 부터  $(r + K, c)$ 까지의 세로 합과
- ✓  $(r, c - K)$ 부터  $(r, c + K)$ 까지의 가로 합이  $2K + 1$ 이라는 것과 같은 의미입니다.
- ✓ 누적합 전처리를 통해 크기가  $K$ 인 십자를  $\mathcal{O}(1)$ 에 판별할 수 있습니다.
- ✓ 이를 이용해 모든  $(r, c)$ 에 대해 크기가  $K$ 인 십자의 개수를 세어주면 됩니다.
- ✓ 이 방법의 시간복잡도는  $\mathcal{O}(NM)$ 입니다.

# G. 방공호 만들기

ad\_hoc, math

출제진 의도 - Normal

 제출 12회, 정답 1명 (정답률 8.33%)

 picaso70, 99분

 윤찬규 dldyou

## G. 방공호 만들기

- ✓ 45도 기울인 정사각형 꿀(마름모꿀)로 만드는 것이 최선입니다.
- ✓ 벽을 하나씩 늘려보면 4의 배수에 따라 규칙이 생기는 것을 확인할 수 있습니다.
- ✓ 벽의 개수  $N$ 에서  $\frac{N}{4} = K$ 라고 하면 방공호의 넓이는 아래의 규칙을 따릅니다.
- ✓  $N$ 을 4로 나눈 나머지가

0일 때,  $N - 1$ 개를 사용했을 때보다 넓이가  $K - 1$ 만큼 증가합니다.

1일 때,  $N - 1$ 개를 사용했을 때보다 넓이가  $K$ 만큼 증가합니다.

2일 때,  $N - 1$ 개를 사용했을 때보다 넓이가  $K$ 만큼 증가합니다.

3일 때,  $N - 1$ 개를 사용했을 때보다 넓이가  $K + 1$ 만큼 증가합니다.

- ✓ 이를 이용해  $\mathcal{O}(1)$ 에 답을 구할 수도 있습니다.

# H. 격자 연결하기

dynamic\_programming

출제진 의도 - Normal

 제출 4회, 정답 3명 (정답률 75%)

 wariatros, 115분

 윤찬규 dldyou



## H. 격자 연결하기

- ✓ 두 격자를 선택하여 최단 경로는 어떤 모양으로 생길까요?
- ✓ 좌상 → 우하, 우상 → 좌하 로 가는 두 가지 경로만 존재합니다.
- ✓  $(r, c)$ 까지 보았을 때, 얻을 수 있는 최대 점수를 각각의 상황에서 구해봅시다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ (1, 1)까지 보았을 때의 최대 점수는 -1점입니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ (1, 1)부터 거리가 1인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ 거리가 2인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	3	-3	6
6	5	7	-4
-4	-6	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ 거리가 3인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	3	0	6
6	11	7	-4
2	-6	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ 거리가 4인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	3	0	6
6	11	18	-4
2	5	-3	-3

## H. 격자 연결하기

- ✓ 좌상 → 우하 로 가는 경로부터 봅시다.
- ✓ 거리가 5인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	3	0	6
6	11	18	14
2	5	15	-3

## H. 격자 연결하기

- ✓ 거리가 6인 위치까지 보았을 때의 최대 점수는 다음과 같습니다.
- ✓ 좌상 → 우하 로 가는 경로에서의 최대 점수는 18점입니다.
- ✓ 우상 → 좌하 에서도 진행해보면 최대 점수가 19점인 것을 알 수 있습니다.

-1	-3	-2	-1
2	1	-3	6
4	5	7	-4
-4	-6	-3	-3

-1	-3	-2	-1
2	3	0	6
6	11	18	14
2	5	15	12




# I. 우선순위 큐와 시뮬레이션

binary\_search

출제진 의도 - Normal

 제출 11회, 정답 0명 (정답률 0%)

 —

 권순호 yookwi

## I. 우선순위 큐와 시뮬레이션

- ✓ 문제에서 주어진 큐의 성질을 파악해봅시다.
- ✓ 쿼리는 모든 원소에  $A_i$ 를 더하고  $K$ 로 나눈 나머지를 취하고 있습니다.
- ✓ 그 후 큐에 들어있는 원소 중에서 최댓값을 출력합니다.
- ✓ 현재의 top  $T$ 에서  $x$ 를 더했을 때의 값이  $K$ 보다 작다면 top은  $T + x$ 입니다.
- ✓  $K$ 보다 크다면 top의 가장 유력한 후보는  $T$  다음으로 컷던 원소에  $x$ 를 더한 값입니다.
- ✓ 만약 이 값도  $K$ 보다 크다면 그 다음으로 컷던 원소가 후보가 됩니다.

## I. 우선순위 큐와 시뮬레이션

- ✓ 우선순위 큐의 원소들을 모두 정렬한 상태로 시작해봅시다.
- ✓ 앞에서 했던 것처럼 쿼리를 진행하면, 문제가 원순열의 형태를 띄고 있는 것을 확인할 수 있습니다.
- ✓ 모든 수에  $A_i$ 를 더할 필요가 있을까요?

## I. 우선순위 큐와 시뮬레이션

- ✓ 여태까지 쿼리를 진행하며 총 더한 값  $S$ 만 가지고 진행해봅시다.
- ✓  $S$ 를  $K$ 로 나눈 나머지를  $B$ 라고 합시다.
- ✓  $K - B$ 보다 크거나 같은 값 중 가장 작은 것을  $Q$ 에서 찾아봅시다.
- ✓ 해당 값은 무엇을 의미할까요?


## I. 우선순위 큐와 시뮬레이션

- ✓  $Q$ 에서  $K - B$ 보다 크거나 같은 값 중 가장 작은 것을  $Q_i$ 라고 합시다.
- ✓  $Q_i + B$ 는  $K$ 보다 크거나 같은 값 중에서 가장 작은 값입니다.
- ✓ 즉,  $Q_{i-1} + B$ 가  $K$ 보다 작은 값 중에서 가장 큰 값이므로 top이 됩니다.
- ✓ 위의  $K - B$ 를  $Q$ 에서 찾는 과정에서 이분탐색을 활용할 수 있습니다.

# J. 수열 만들기

constructive, greedy

출제진 의도 - Normal

 제출 2회, 정답 0명 (정답률 0%)

 —

 권순호 yookwi

## J. 수열 만들기

✓ 1과  $N$ 을 한번 이상 사용해서 수열을 만들면 아래와 같은 특징이 생깁니다.

$N$ 을  $k$ 번 연속해서 배치했을 때,  $\frac{k \times (k-1)}{2}$ 개의  $N$ 의 배수인 구간이 생깁니다.

1을 포함하는  $N$ 의 배수인 구간이 생길 수 없기 때문에 수열을 나눠서 생각할 수 있게 됩니다.

✓ 따라서  $N$ 을 연속해서 배치하고 1로 적절히 나누면  $N$ 의 배수인 구간을 쉽게 생각할 수 있습니다.

## J. 수열 만들기

- ✓ 이제  $N$ 에서  $\frac{k \times (k-1)}{2}$ 를 적절히 빼서 0으로 만드는 문제가 되었습니다.
- ✓ 이때 사용한 모든  $k$ 의 합 + 사용한  $k$ 의 개수  $-1$ 이  $N$ 보다 작거나 같다면 항상 수열을 만들 수 있습니다.




## J. 수열 만들기

- ✓ 뺄 수 있는 가능한 큰  $k$ 를 잡아  $N$ 에서 뺍니다.
- ✓ 이 과정을 반복하면  $N = 2, N = 5$ 를 제외한 모든 경우가 가능합니다.
- ✓ 해당 경우는 예제를 보거나 브루트포스를 돌려 불가능하다는 것을 확인할 수 있습니다.

# K. 순열과 증가수열

ad\_hoc, math, combinatorics

출제진 의도 - Hard

 제출 0회, 정답 0명 (정답률 0%)

 —

 이승엽 delena0702

## K. 순열과 증가수열

- ✓  $B$ 가 증가수열이기 위해서는  $B_i < B_{i+1}$  이어야 합니다. ( $1 \leq i \leq N - K$ )
- ✓  $A_i + A_{i+1} + \dots + A_{i+K-1} < A_{i+1} + \dots + A_{i+K-1} + A_{i+K}$
- ✓  $A_i < A_{i+K}$
- ✓ 이를 정리하면 다음과 같이  $K$ 개의 그룹으로 나눌 수 있습니다.
- ✓ 
$$\begin{cases} A_1 < A_{1+K} < A_{1+2K} < \dots \\ A_2 < A_{2+K} < A_{2+2K} < \dots \\ \dots \\ A_K < A_{2K} < A_{3K} < \dots \end{cases}$$

## K. 순열과 증가수열

- ✓ 1부터  $N$ 까지의 수가 어느 그룹에 들어갈 지 결정한다면 경우의 수를 구할 수 있습니다.
- ✓ 각 그룹은 오름차순 정렬되므로, 그룹 내의 순서는 고려하지 않아도 됩니다.

- ✓ 
$$\frac{N!}{\prod_{i=1}^K [i \text{ 그룹의 항목 수}]!} = \frac{N!}{\prod_{i=1}^N \left\lfloor \frac{i+K-1}{K} \right\rfloor}$$

# L. 완벽한 도시 설계

graph\_theory

출제진 의도 - Challenging



제출 2회, 정답 0명 (정답률 0%)

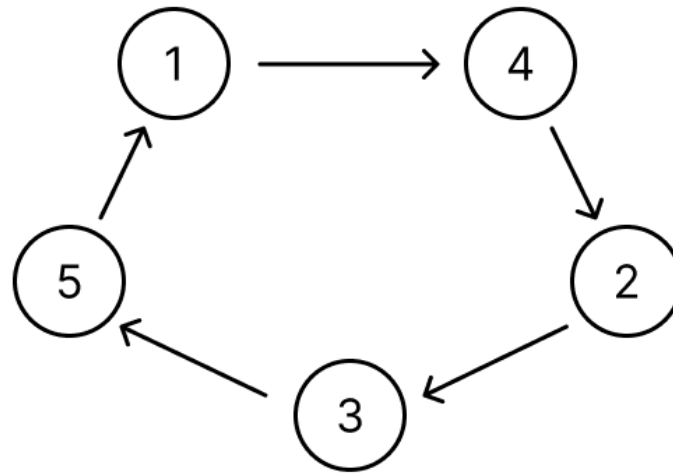


—



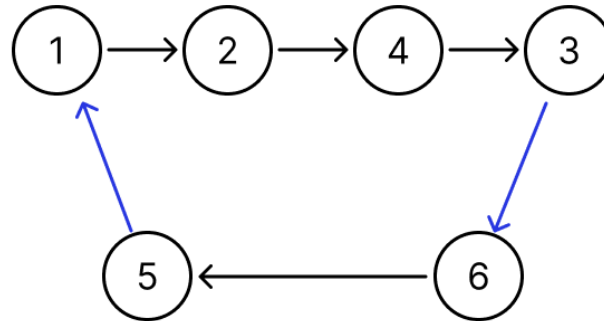
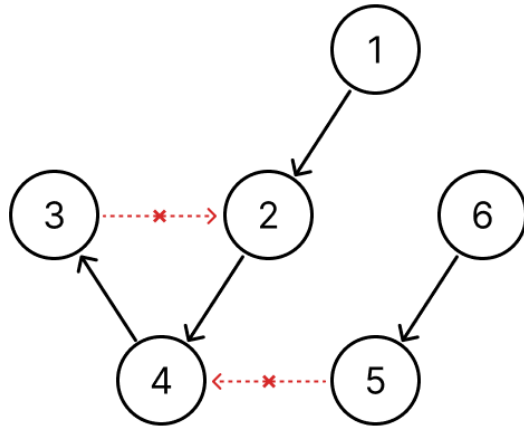
이승엽 delena0702

## L. 완벽한 도시 설계



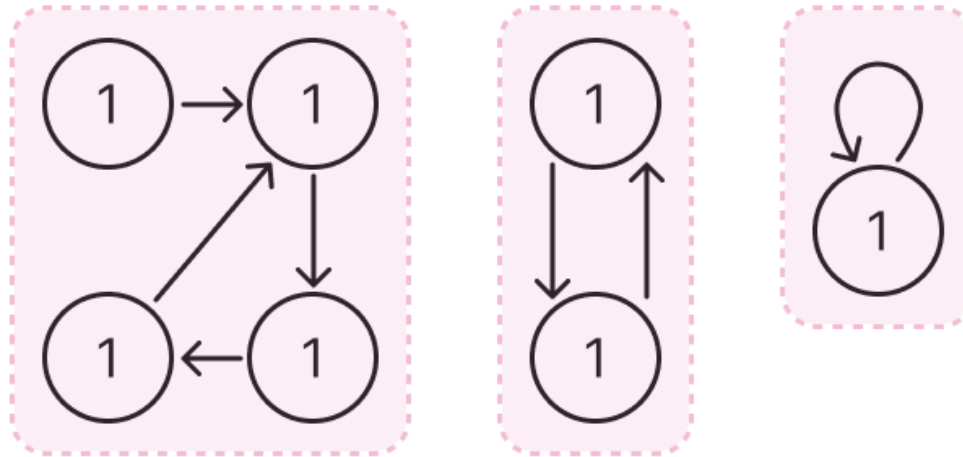
- ✓ 문제의 요구사항을 만족시키고, 어떤 도시에서도 모든 도시로 갈 수 있도록 도시를 설계하기 위해서는 원형으로 구성해야 합니다.
- ✓ 모든 도시에 들어오는 도로가 단 한 개가 되고, 모든 노드가 연결되어 있으면 됩니다.
- ✓ 도로의 목적지를 변경하는 것이 아니라, 도로를 끊고 다시 연결하는 방법으로 생각해 봅시다!

## L. 완벽한 도시 설계



- ✓ 그래프를 모두 직선 형태로 분리한다면 다시 연결했을 때, 원형으로 구성할 수 있습니다.
- ✓ 따라서 그래프를 직선으로 만들 수 있는 최소 도로 제거 횟수를 구하면 정답을 구할 수 있습니다.
- ✓ 예외적으로, 주어진 그래프가 이미 원형 그래프라면 도로를 분리하지 않는 것이 더 적은 횟수로 만들 수 있습니다. (0회)

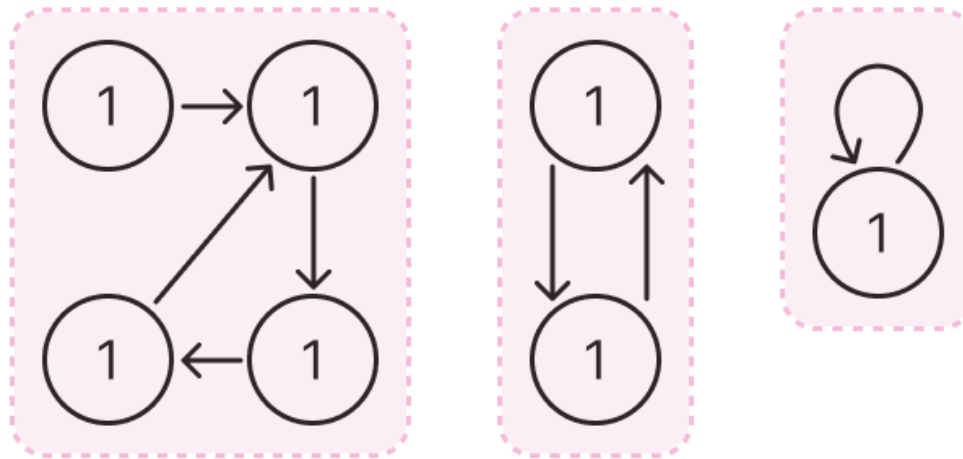
## L. 완벽한 도시 설계



- ✓ 그래프의 연결된 컴포넌트들에 대해 생각해 봅시다.
- ✓ 각 컴포넌트들은 노드의 개수와 간선의 개수가 같습니다.



## L. 완벽한 도시 설계



✓ 즉, 사이클이 반드시 한 개 이상 있습니다.

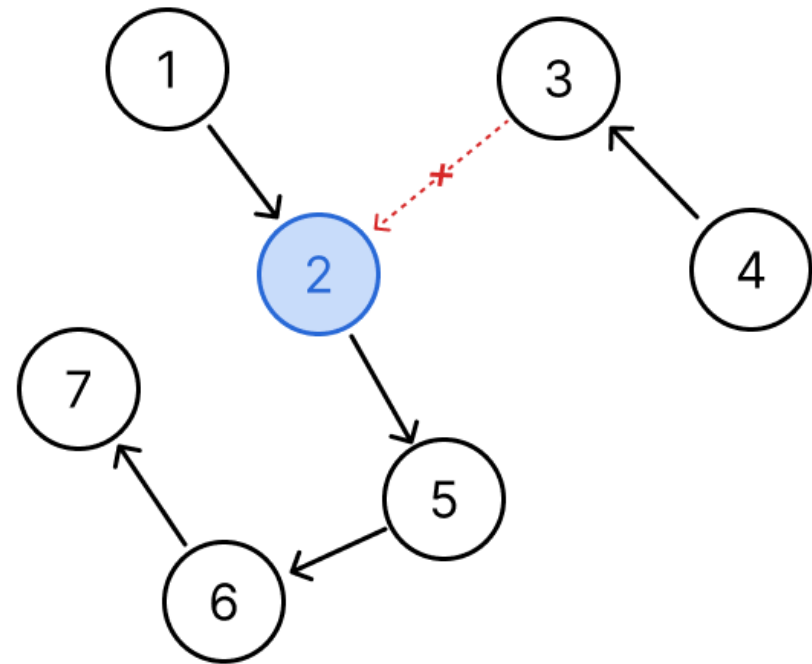
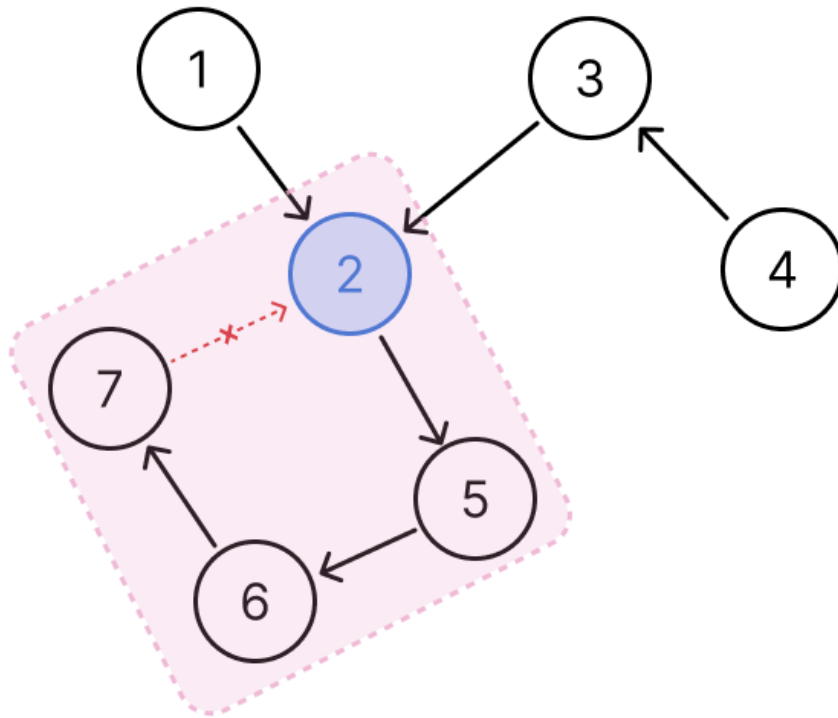
(트리에 한 개의 간선이 추가된다고 생각하면 편합니다.)

✓ 각 컴포넌트들은 다음의 방법으로 도로를 최소 횟수로 제거할 수 있습니다.

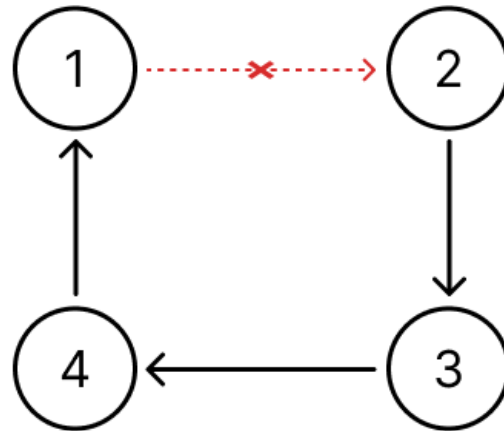
## L. 완벽한 도시 설계

1. 컴포넌트의 사이클에 indegree가 2 이상인 노드가 있는 경우
  - ✓ indegree가 2 이상인 노드 중 하나를 선택합니다.
  - ✓ 그 노드에 들어오는 간선 중, 사이클에 구성되는 간선을 제거합니다.
  - ✓ 남은 그래프는 트리가 되므로, indegree가 2 이상인 모든 노드에 대해서 아무 순서대로 들어오는 간선을 한 개가 남을 때까지 제거합니다.
  - ✓ 최소 간선 제거 횟수는  $\sum_{v \in V} \max([v \text{의 indegree}] - 1, 0)$  입니다.

## L. 완벽한 도시 설계



## L. 완벽한 도시 설계



2. 컴포넌트의 사이클에 indegree가 2 이상인 노드가 없는 경우

- ✓ 컴포넌트 전체가 원형 그래프일 수밖에 없습니다.
- ✓ 직선으로 수정하기 위해 임의의 간선 한 개를 자릅니다.
- ✓ 최소 간선 제거 횟수는 1 입니다.