가희와 함께 하는 5회 코딩테스트

문제		의도한 난이도	출제자
A	Gahui and Soongsil University station	**Easy**	chogahui05
B	가희와 열리지 않는 건널목	**Easy**	chogahui05
C	가희와 노선 건설 놀이	**Medium**	chogahui05
D	가희와 서울 지하철 3호선	**Medium**	chogahui05
E	Gahui and ILGAM lake	**Medium**	chogahui05
F	가희와 지하철역 저장 시스템 1	**Hard**	chogahui05
G	가희와 지하철역 저장 시스템 2	**Challenging**	chogahui05
H	가희와 코드	**Challenging**	chogahui05
I	가희와 서울 지하철 1호선	**Challenging**	chogahui05

A. Gahui and Soongsil University station

`implementation`
출제진 의도 – **Easy**

- 제출 -, 정답 - (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

A. Gahui and Soongsil University station

- 에스컬레이터 1칸의 높이는 21cm, 계단 1칸의 높이는 17cm입니다.
- 승강장까지 내려가기 위한 계단 수와 에스컬레이터 단 수를 세어서 사칙 연산을 해 주면 됩니다.
- 유독 고도가 같다는 제약 조건이 많이 나온 것은 기분 탓입니다.

B. 가희와 열리지 않는 건널목

`implementation`, `string`, `parsing`
출제진 의도 – **Easy**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

B. 가희와 열리지 않는 건널목

- ✓ 상행과 하행 열차별로 접근 시간이 있습니다.
- ✓ 접근 시간을 받아 아래와 같이 처리하면 됩니다.
  - s 초에 접근한 경우, $s + 39$ 초까지 못 지나간다는 $flag$ 를 설정합니다.
  - 모든 처리 후, 못 지나간다는 $flag$ 가 있으면 차단기가 닫혀있다고 하면 됩니다.

C. 가희와 노선 건설 놀이

ad hoc, math
출제진 의도 – **Medium**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

C. 가희와 노선 건설 놀이

- 아. 딱 봐도 ad_hoc 스럽습니다.
- 그리고 가능한 경우만 주어진다고 하였습니다.
- 이런 문제를 풀 때에는 $corner$ 케이스와 같이 극값을 생각하는 것이 좋습니다.

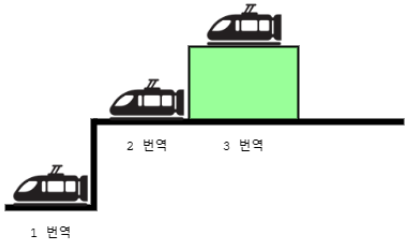
C. 가희와 노선 건설 놀이

- 먼저 가능한 경우만 주어집니다. 조건을 해석해 봅시다.
- 가장 거리가 먼 두 역은 1번 역과 n 번 역입니다. 그렇다면
  - 두 역 사이의 거리는 $n-1$ 이 되겠네요.
  - 1번 역의 고도만 -1로 낮춰 봅시다. 그러면 그림이 어떻게 그려질까요?

C. 가희와 노선 건설 놀이

- 1번째 역만 고도가 유독 낮습니다.
- 만약에 다른 역의 고도를 낮추지 않는다면 고도가 최소인 역은 **유일**합니다.

C. 가희와 노선 건설 놀이



- 만약에 출력 값이 d인 경우는 어떻게 하면 될까요?
- $d+1$번 역의 고도를 높이면 됩니다.

C. 가희와 노선 건설 놀이

- 즉 고도가 최소인 역을 극값으로 고정시켜 놓고
- 최대 고도를 가지는 역도 하나만 있게 하면 어렵지 않게 해결할 수 있습니다.

D. 가희와 서울 지하철 3호선

`brute force`, `bit operation`, `combination`
출제진 의도 – **Medium**

- 제출 -, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

D. 가희와 서울 지하철 3호선

- **롤러코스터 구간**의 길이를 구하는 것부터 해 봅시다.
  - 직전 역과 현재 역이 각각 지상역, 지하역이거나
  - 직전 역과 현재 역이 각각 지하역, 지상역이라면
- 이 두 역은 같은 롤러코스터 구간에 있습니다. 이건 그냥 $implementation$ 입니다.

D. 가희와 서울 지하철 3호선

- 경우의 수는 어떻게 구할까요? 그냥 층수별로 완탐하면 시간초과가 날 거 같습니다.
- 대신에 지상역이면 1, 지하역이면 0이라는 상태로 처리합시다.
  - 역이 $B1$층에 있거나 $B2$층에 있거나 롤러코스터 구간에는 영향을 미치지 않아요.
  - 역이 $F1$층에 있거나 $F5$층에 있거나 롤러코스터 구간에는 영향을 미치지 않아요.

D. 가희와 서울 지하철 3호선

- 역의 상태가 주어졌어요. 이전 역이 지하역이고 현재 역이 지상역이라고 해 봅시다.
  - 이전 역이 $B5$ 층에 있었다면, 현재 역은 $F3$ 층에 있으면 안 되나요?
  - 이전 역이 $B4$ 층에 있었다면, 현재 역은 $F1$ 층에 있으면 안 되나요?
  - 둘 다 아님을 알 수 있어요.
- 이 말인 즉슨, 이전 역의 가능한 층수가, 현재 역의 가능한 층수 집합에 영향을 미치지 않습니다.

D. 가희와 서울 지하철 3호선

- 따라서 $\mathcal{O}(n * 2^n)$ 으로 처리 가능합니다.
- 여담으로 이 문제는 지상으로 올라갔다 지하로 내려갔다 하는 구간이 재밌어서 낸 문제입니다.

E. Gahui and ILGAM lake

`implementation, prefix_sum`
출제진 의도 – **Medium**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

E. Gahui and ILGAM lake

- 왜 $ILGAM$ 호수를 냈을까요?
- 건국대에서 제일 유명한 것이기 때문입니다. 랜드마크 정도로 생각하면 될까요?
- 일감호가 몇 대학교보다 크기 때문에 n 제한도 커지게 되었습니다.

E. Gahui and ILGAM lake

- ✓ n이 매우 커서 일일히 인접한 곳을 방문하면서 합산하는 건 비효율적입니다. 그러면 어떻게?
  - 1바퀴 전체 거리를 구합니다.
  - a에서 b로 이동할 때 아래 둘의 합은 1바퀴 전체 거리입니다.
    - 반시계 방향으로 이동한 거리
    - 시계 방향으로 이동한 거리
  - 이를 이용해서 케이스 처리를 하면 됩니다.

F. 가희와 지하철역 저장 시스템 1

`data structure`, `bit operation`, `implementation`
출제진 의도 – **Hard**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

F. 가희와 지하철역 저장 시스템 1

- 전체 U 업데이트에 나오는 특징의 종류는 많아야 9개입니다.
- 9개. 뭔가 bit 스러운 것을 사용하고 싶지 않으신가요?

F. 가희와 지하철역 저장 시스템 1

- 특징 $k_1, k_2, \ldots$ 등이 나온다고 해 보겠습니다.
- 각 역이 해당 특징을 가지고 있으면 1, 없으면 0으로 $toggle$ 할 수 있겠습니다.
- 예를 들어 10001 은 특징 k_1, k_5 를 가지고 있는 것입니다.
- 이제 각각 상태별로 어떤 역이 있는지 저장하면 됩니다.

F. 가희와 지하철역 저장 시스템 1

- 여기서 문제, k_1 특징을 가지고 있는 역의 개수를 구해야 합니다.
- k_1, k_5 를 가지고 있는 역에 대해서는 어떻게 처리해야 할까요?
- 각 상태별로 모두 돌면서, bit_and 연산으로 조건 검사를 해 주면 됩니다.

G. 가희와 지하철역 저장 시스템 2

data structure, graph theory
출제진 의도 – **Challenging**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

G. 가희와 지하철역 저장 시스템 2

- $kakao$ 기출 문제를 2개에서 3개쯤 섞어서 낸 문제입니다.
- 15940번 문제를 참고하여 서술한 문제입니다. 한 번 풀어보시는 것도?

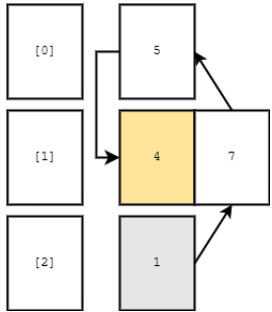
G. 가희와 지하철역 저장 시스템 2

- 최단 거리는 어렵지 않게 구하셨을 거 같습니다.
- 그런데 $cache$ 서버에 lru 까지 섞이니 이게 왜 끔찍한 혼종인가? 싶습니다.
- 분리해야 할 부분 lru 를 생각해 봅시다.
- $LinkedHash$ 계열이라던지 $OrderedDict$ 정도는 들어보셨을 텐데요.

G. 가희와 지하철역 저장 시스템 2

- 원래 $hash$ 계열은 들어온 순서가 보장 안 되긴 합니다. 최근 $python$ 은 보장되긴 하지만요.
- 그런데 $order$ 가 붙으면 네. 들어온 순서대로 순회 가능합니다. 어떻게?
- $java$ 의 $LinkedHash$ 에는 $Link$ 가 있습니다. 그리고 $Hash$ 가 있어요.

G. 가희와 지하철역 저장 시스템 2



- 실제로 내부를 보면 위 그림과 같은 구조로 이루어져 있습니다.
- 고로 가장 앞에 있는 것과 뒤에 있는 것도 무리 없이 뺄 거 같습니다.
- $head$와 $tail$만 안다면 말입니다.

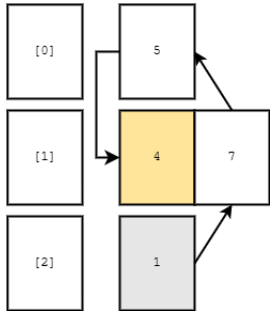
G. 가희와 지하철역 저장 시스템 2

- 중요한 것은 새롭게 정보에 접근했을 때, 앞으로던 뒤로 밀어야 합니다. 왜?
- 가장 최근에 접근한 정보이기 때문입니다.
- $head$와 $tail$만 안다면 말입니다.

G. 가희와 지하철역 저장 시스템 2

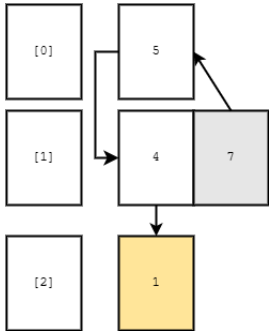
- java의 $LinkedHashMap$은 $removeEldestEntry$ 메소드가 있습니다.
- 무엇을 하는 것일까요? 가장 오래 전에 접근한 $item$을 제거합니다.
- 이는 어떻게 알 수 있을까요? $head$나 $tail$을 보면 됩니다.

G. 가희와 지하철역 저장 시스템 2



- 1, 7, 5, 4 순으로 추가되었다고 해 봅시다.
- 가장 오래 전에 접근한 것은 1 혹은 4 다라는 것을 알 수 있어요.
- 문제. 만약에 1에 접근하면 어떻게 되어야 할까요?

G. 가희와 지하철역 저장 시스템 2



- 당연하게도 위 그림과 같이 되어야 할 겁니다.
- $AfterNodeAccess$ 메서드를 보면, 최근에 접근한 것에 어떤 작업을 하고 있음을 알 수 있어요.

G. 가희와 지하철역 저장 시스템 2

- ✓ $removeEldestEntry$ 메서드를 $override$ 하면
  - sz 이상이면 삭제되게끔 트리거를 걸 수 있습니다.
  - $python$ 에서 $orderedDict$ 는 그런 게 있나 모르겠습니다.

G. 가희와 지하철역 저장 시스템 2

- ✓ $python$의 $orderedDict$에는 $move\_to\_end$ 함수가 있습니다.
- ✓ 이것은 $item$을 end로 이동시킵니다.
  - 그러면 용량이 꽉 차서 제거해야 할 때, $popitem$을 호출해야 하는데요.
  - 끝에서 제거하지 않고 처음에서 제거할 수 있게 $last$를 $false$로 설정하면 되겠네요.

G. 가희와 지하철역 저장 시스템 2

- 최단거리는 어떻게 구할까요?
- $many\_to\_many$ 이고 m 이 작으므로, $floyd\_algorithm$ 을 돌리면 됩니다.
- 그런데 여기서 걸리는 것은 거리가 여럿일 경우 id 가 작은 $cache$ 를 이용한다 입니다.
- 이것은 그냥 정렬하고 좌표압축 해 버리면 됩니다.

G. 가희와 지하철역 저장 시스템 2

- 정리하면
  - $Collection$ 별로 lru 같은 것을 쉽게 구현할 수 있게 하는 경우가 있습니다.
  - 이미 구현되어 있다면 **잘 쓰는 것**이 현명한 선택이 아닐까 싶습니다.
  - 사실 저도 이 편한 방법을 $setting$ 하면서 배웠습니다.

H. 가희와 코드

`implementation`
출제진 의도 – **Challenging**

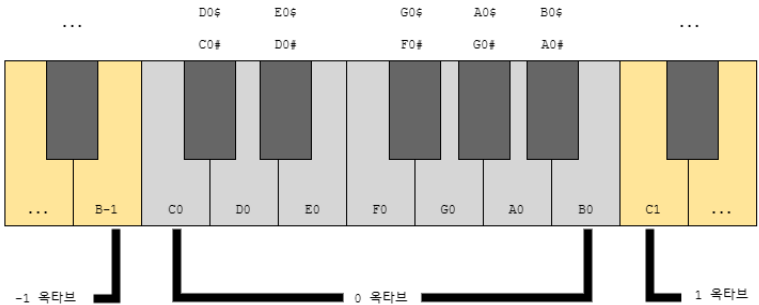
- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

H. 가희와 코드

- 엄청나게 긴 지문입니다.
- 정보량도 어마무시하게 많습니다. 여기에 나온 음악 용어만 몇 개인지.. 국어 영역인가요?
- 핵심 내용만 빠르게 추려 봅시다.

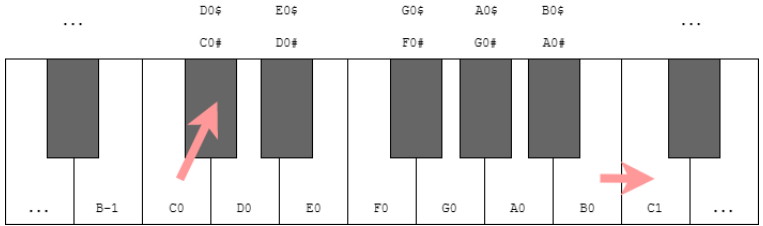
- 음은 옥타브 정보와 음 이름 정보가 들어 있습니다.
  - 음 이름은 도, 레, 미, 파, 솔, 라, 시와 같은 것들입니다.
  - 옥타브는 음의 높이를 나타냅니다.

H. 가희와 코드



- 뜬금없이 피아노 그림이 하나 주어졌어요.
  - -1, 0, 1 옥타브 순으로 음 높이가 높아진다고 했어요.
  - 그리고 이 그림은 다음에 나오는 정보를 파악하기 위한 중요한 정보입니다.

H. 가희와 코드



- 두 음이 같은 옥타브에 있으면
  - 음 이름과 대응되는 순번이 1 차이이면 인접합니다.
  - 예를 들어, $C0$ 과 $C0\#$ 은 인접합니다.
- 두 음이 다른 옥타브에 있으면
  - $octave$ 옥타브 B 와 $octave + 1$ 옥타브 C 가 인접합니다.

H. 가희와 코드

- 표 1과 매치해 보면 반음씩 올라가는 것이 어떤 의미인지 알 수 있지만
- 조금 더 직관적으로 그려보면 위와 같습니다.
- 이러면 인접 관계가 조금 더 명확하게 보입니다.

H. 가희와 코드

- 음 높이가 높아지면서 옥타브도 높아집니다. 음의 이름은 규칙이 없을까요?
- $C, C\#, \ldots, B$ 의 패턴이 계속 반복됨을 알 수 있습니다.
- 따라서 나중에 구현할 때 원형으로 처리하는 $trick$ 이 쓰일 수도 있겠네요.

H. 가희와 코드

- 다음에 장 3도와 단 3도가 주어집니다.
  - 이들은 반음 4개, 3개 거리라고 되어 있어요.
  - 결국 음 이름이 u인 것에서 단 3도 높은 음 이름을 구해야 할 수도 있겠네요.
  - 이는 원형 배열에서 거리 처리로 퉁치면 될 거 같네요.

- 이제 **코드**를 봅시다.
- 근음을 기준으로 장 3도, 단 3도 위에는 $Major$ 코드이고 ... 라는 내용이 있습니다.
- $Major, minor, aug, dim$ 에 대해 처리해 주면 됩니다.
- 근음이 반음 높아지면, 3 음과 5 음도 반음 높아진다는 것을 파악하시면 쉽습니다.

- ✓ 이거 코드 종류별로 함수 따로 만들어야 할까요?
- ✓ 코드 종류가 4개밖에 없으니 $hard\_coding$ 합시다.
- ✓ 어떤 코드 c가 음의 이름 u_1, u_2, u_3을 가지고 있다면, 그것도 같이 저장합시다. 어떻게?

H. 가희와 코드

- 마디별로 음이 나옵니다. 음은 음 이름과 옥타브 정보가 있으니
- 음을 읽을 때 마다 아래 알고리즘을 생각해 볼 수 있어요.
  - 음 n_1 의 음 이름이 u_1 이라 할 때
  - 음 이름이 u_1 인 것을 가지는 코드들에 대해서
  - 구성하는 음 이름이 나온 총 횟수를 모두 더합니다.
  - 그렇다면 con[음 이름] = **음 이름을 가진 코드들**로 정보를 저장하면 되겠네요.

H. 가희와 코드

- ✓ 각 코드별로 코드를 이루는 음 이름이 몇 번 나왔는지 카운트 하는 것은 해결했습니다.
- ✓ 이전 p개의 마디 중 k회 이상 쓰인 코드인 것은 어떻게 검사할까요?
  - 코드 c_1이 어느 마디들에 나왔는지 저장합니다.
  - $dynamic\_array$ 같은 $array$에 저장하면
    - 마지막에 나온 위치도
    - 최근 k번째에 몇 번 마디에서 나왔는지를
  - $\mathcal{O}(1)$에 찾을 수 있습니다.

H. 가희와 코드

- 등장하는 코드는 48개입니다.
- 코드를 구성하는 음 이름의 합산 횟수가 가장 많은 코드도 48개의 코드를 순회해 보면 됩니다.
- 이전 p개의 마디 중 k회 이상 나왔는지도 48번 다 돌리면 됩니다.

I. 가희와 서울 지하철 1호선

`data_structure`, `greedy`, `math`, `sliding_window`
출제진 의도 – **Challenging**

- 제출 -번, 정답 -명 (정답률 -%)
- 처음 푼 사람: -, -분
- 출제자: chogahui05

- 3 대장 중 제일 짧은 지문입니다.
- 사실 2호선을 타고 가면서 생각한 문제입니다.
- 그런데 머릿속에서 상상하는 것이랑 세팅하는 것이랑 차원이 다른 문제더라고요.

- 수 n 개의 최소 공배수가 소수라면 어떤 특성이 있을까? 라는 질문에서 출발하였습니다.
- 더 생각해 보니 $greedy$ 와도 엮을 수 있을 거 같아, 출제하게 되었습니다.
- 하나씩 차례대로 설명해 보겠습니다.

- 수 n 개의 최소 공배수가 소수 k 라고 해 봅시다. 이 경우 n 개의 수는
  - 1또는 k 가 되어야 합니다.
  - 왜? k 가 소수이기 때문에, **1과 k 로만** 나누어 떨어지기 때문입니다.

- 따라서 s 번 역을 시발역으로 하고, e 번 역을 종착역으로 하는 열차 번호가 소수 k 라면
  - s 번째 역부터 e 번째 역까지 적혀있는 수는 1 또는 k 입니다.
  - s 번 역과 e 번 역 사이에 있는 v 번 역에 써져 있는 수를 x 라 하면
    - $x = 1$ 이거나 $x = k$ 이다.
  - 각 역마다 이러한 절들이 들어갑니다.

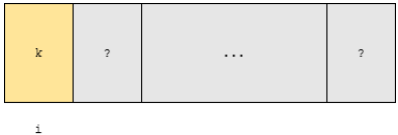
- 여기까지, 각 역에 들어갈 수 있는 수를 **제한**하였습니다.
- 이런 조건이 각 역마다 몇 개 들어있는지 봅시다.
  - $x = 1$ 이거나 $x = k$ 이다.
  - 단, k 가 같다면 중복 처리를 해 줍니다.

- 걸리는 조건의 개수
  - 0개인 경우 아무 수나 와도 됩니다.
  - 1개인 경우 k가 와야 합니다.
  - 2개 이상인 경우 무조건 1이 와야 입니다.

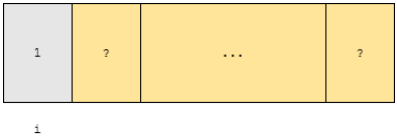
- 1번째 것은 당연한 것이니 생략하고 3번째 것부터 봅시다.
- 조건의 개수가 2개 이상이라는 건 무엇을 의미할까요?
  - $x = 1$ 이거나 $x = k_1$ 입니다.
  - $x = 1$ 이거나 $x = k_2$ 입니다.
- 여기서 k_1 과 k_2 는 다릅니다. 1이 아닌 다른 수가 이 두 조건을 모두 만족할 수 있을까요?

- 1이 아닌 수는 1이 아니므로, 각 조건에 대해서 선행절은 무시됩니다. 따라서
  - $x = k_1$ 입니다.
  - $x = k_2$ 입니다.
- 조건을 동시에 만족해야 합니다. 그런데 k_1 과 k_2 는 다르다고 했습니다.
- 따라서, 둘 중 하나 이상의 수와 같다고 해도 다른 수와는 같지 않습니다.

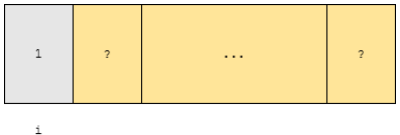
- 걸리는 조건의 개수가 $x = 1$ 또는 $x = k$ 1개인 경우 k가 와야 합니다. 이건 왜 그럴까요?
- **기회**라는 관점에서 접근해 봅시다.
  - 수 k를 넣을 수 있는 기회



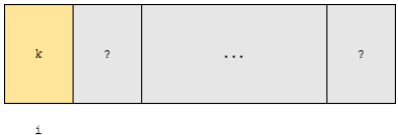
- i번째 위치에 k를 넣을 수 있다고 해 보겠습니다.
- 그리고 제가 표시한 직사각형 구간의 최소 공배수가 k라 하겠습니다.
- 그러면 노란색과 회색 영역 중 최소 하나 이상의 영역에 k가 들어가야 해요.
  - 수 1은 아무렇게나 들어가도 적당히 잘 맞추면 되겠지만
  - 수 k는 여러 절이 들어오면 들어가지 못합니다.



- i번째 위치에 1을 넣었다고 해 보겠습니다.
- 그러면 노라낵 영역 중 하나에 k가 들어가야 해요.
- 문제는, 회색 영역에는 k만 있는데 노란색 영역에는 그렇지 않을 수도 있어요.
- 기회가 있음에도 발로 차 버린 셈입니다. 이런.



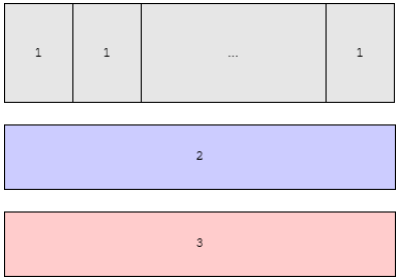
- 다른 관점에서 접근해 봅시다.
- 위 그림은 노란색 영역 중 최소 하나 이상에 k가 들어가야 하는 상황입니다. 이 상황 말이죠.



- 이 상태에서는 못 만들까요?
- 회색으로 칠해진 영역에 1이 들어가도 되지만 k가 들어가도 됩니다. 어?

- 상황을 다시 요약해 봅시다.
  - 상태 s_1 에서는 k 라는 상황만 쓸 수 있는데
  - 상태 s_2 에서도 k, k' 이라는 상황을 쓸 수 있는 셈입니다.
  - 그런데 s_2 가 s_1 보다 더 이득이에요.
- 문제를 잘 분석해 보니 구조가 그러하네요.

✓ 이제 진짜로 잘 채워졌는지 확인해 보아야 합니다. 왜?



- 진짜 구간의 lcm 이 k 가 맞나 확인해 보아야 합니다.
- $segment\_tree$ 나 lca 를 구축할 때 사용하는 dp 를 이용하면 쉽게 할 수 있습니다.
- 그런데 이 방법 말고 다른 방법이 없나요? 물론 있습니다.

- ✓ $sliding_window$를 아래 로직으로 돌리면 어떨까요? 1 또는 소수만 나오니
  - s로부터 보았을 때 나오는 수의 종류가 하나인 가장 우측 위치 계산
  - s로부터 보았을 때 나오는 수의 종류가 두 개인 가장 우측 위치 계산
- ✓ 다만 현실적으로 $segment_tree$나 lca가 훨씬 쉽고 구현이 간결할 뿐.

- 시간 복잡도는 $\mathcal{O}\left(Qlognlog(2000000)\right)$ 이 됩니다.
- 등장하는 수가 소수여야 하는 것을 이용해서 gcd 를 구하는 복잡도를 뗄 수도 있습니다.