

알튜브

동적 계획법과 최단 거리 역추적

지금까지 배운 동적 계획법과 최단 거리는 단순히 최적해의 정답만 도출했습니다.
여기서 최적해를 만들기까지의 경로를 역으로 따라가며 구하는 문제들을 풀어봅시다.

보통 역추적은 동적 계획법, 최단 거리, 그리고 최단 경로 알고리즘과 함께 나와요.

현재 해를 구할 때 과거의 해를 활용하면 동적 계획법(DP)

두 정점 사이의 최단 거리를 구할 땐 BFS

가중치가 존재하는 그래프에서 정점 사이의 최단 거리를 구할 땐 최단 경로

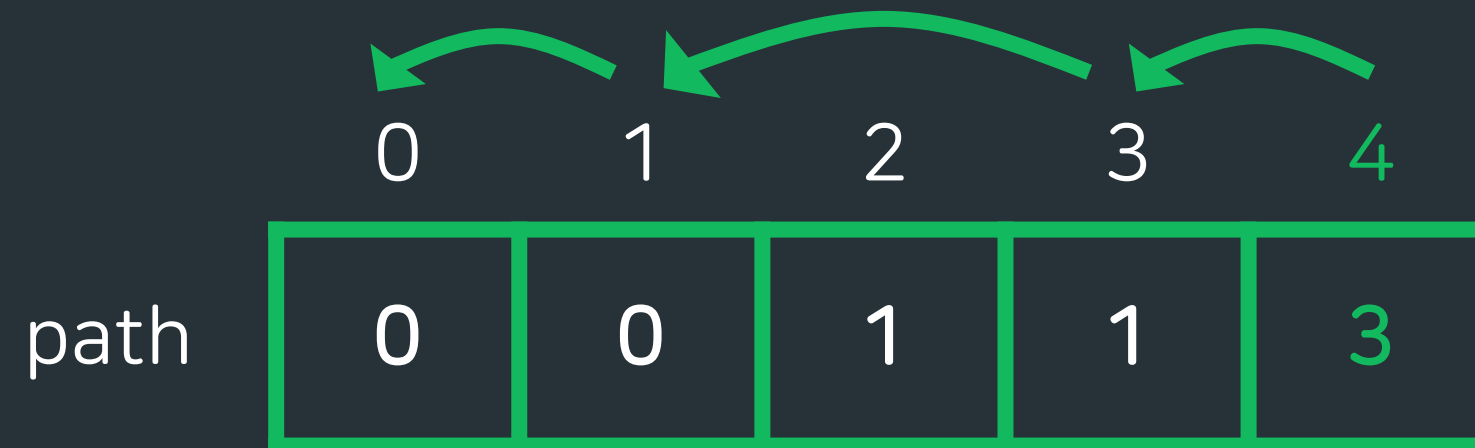
동적 계획법 역추적

- 현재 해(최종 해)를 구하기까지의 경로를 역으로 추적하는 것
- 기존 동적 계획법과 동일한 풀이에 경로를 저장하는 부분만 추가
- 최종 해가 존재하는 위치에서 시작해서 저장한 경로값을 따라 추적

경로는 어떻게?

배열 활용

- 경로 배열을 만들어서 DP배열의 값이 갱신될 때 사용한 과거의 해의 위치를 저장



백준 문제 번호 16563

에라토스테네스의 체 활용한 소인수분해

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-------|---|---|---|---|---|---|---|---|
| prime | 0 | 2 | 3 | 2 | 5 | 2 | 7 | 2 |

- 모두 완료하면 역으로 경로 조사 시작
- 8 -> $8/\text{prime}[8]$ -> 4 -> $4/\text{prime}[4]$ -> 2 -> $2 == \text{prime}[2]$ 종료

이때의 prime 값 출력

/<> 14002번 : 가장 긴 증가하는 부분 수열 4 - Gold 4

문제

- 수열 A가 주어졌을 때, 가장 긴 증가하는 부분 수열을 구하는 문제
- 만족하는 수열이 여러가지인 경우 아무거나 출력

제한 사항

- 수열 A의 크기 N은 $1 \leq N \leq 1,000$
- 수열 A를 이루고 있는 수는 $1 \leq A_i \leq 1,000$

예제 입력 1

```
6
10 20 10 30 20 50
```

예제 출력 1

```
4
10 20 30 50
```

/<> 11053번 : 가장 긴 증가하는 부분 수열 - Silver 2

문제

- 수열 A가 주어졌을 때, 가장 긴 증가하는 부분 수열의 길이를 구하는 문제

제한 사항

- 수열 A의 크기 N은 $1 \leq N \leq 1,000$
- 수열 A를 이루고 있는 수는 $1 \leq A_i \leq 1,000$

가장 긴 증가하는 부분 수열 (LIS)

DP

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감

| | 0 | 1 | 2 | 3 | 4 | 5 |
|----|------|-----|-----|-----|-----|-----|
| dp | 1 | 1 | 1 | 1 | 1 | 1 |
| | [10, | 20, | 10, | 30, | 20, | 50] |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | 0 | 1 | 2 | 3 | 4 | 5 |
|--------------------------|----|----|----|----|----|----|
| dp | 1 | 1 | 1 | 1 | 1 | 1 |
| [10, 20, 10, 30, 20, 50] | | | | | | |
| path | -1 | -1 | -1 | -1 | -1 | -1 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | | | | | | |
|------|-----------|-----|-----|-----|-----|-----|
| | ← 10 < 20 | | | | | |
| | 0 | 1 | 2 | 3 | 4 | 5 |
| dp | 1 | 2 | 1 | 1 | 1 | 1 |
| | [10, | 20, | 10, | 30, | 20, | 50] |
| path | -1 | 0 | -1 | -1 | -1 | -1 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | | | | | | | |
|------|---|------|-----|-----|-----|-----|-----|
| | ← | 0 | 1 | 2 | 3 | 4 | 5 |
| dp | | 1 | 2 | 1 | 1 | 1 | 1 |
| | | [10, | 20, | 10, | 30, | 20, | 50] |
| path | | -1 | 0 | -1 | -1 | -1 | -1 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | | | | | | | |
|------|---|------|-----|---------|-----|-----|-----|
| | ← | 0 | 1 | 2 | 3 | 4 | 5 |
| | | | | 20 < 30 | | | |
| dp | | 1 | 2 | 1 | 3 | 1 | 1 |
| | | [10, | 20, | 10, | 30, | 20, | 50] |
| path | | -1 | 0 | -1 | 1 | -1 | -1 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적


- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | | | | | | |
|------|--------------------------|---|----|---|---|----|
| | ← 10 < 20 | | | | | |
| | 0 | 1 | 2 | 3 | 4 | 5 |
| dp | 1 | 2 | 1 | 3 | 2 | 1 |
| | [10, 20, 10, 30, 20, 50] | | | | | |
| path | -1 | 0 | -1 | 1 | 2 | -1 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 수열을 0번 인덱스부터 탐색하며 해당 인덱스로 끝나는 “증가하는 부분수열”의 길이의 최댓값을 계산해나감
- 최댓값이 갱신될 때 어느 원소에서 온 건지 경로(인덱스) 저장

| | | | | | | |
|------|--|---|----|---|---|---------|
| |  | | | | | 30 < 50 |
| | 0 | 1 | 2 | 3 | 4 | 5 |
| dp | 1 | 2 | 1 | 3 | 2 | 4 |
| | [10, 20, 10, 30, 20, 50] | | | | | |
| path | -1 | 0 | -1 | 1 | 2 | 3 |

가장 긴 증가하는 부분 수열 (LIS)

DP + 역추적

- 가장 길이가 저장된 곳부터 역으로 **경로 추적**
- 경로를 추적하기 위해 원소값 자체가 아닌 **인덱스를 저장한 것**



- 5 -> path[5] -> 3 -> path[3] -> 1 -> path[1] -> 0 -> path[0] -> -1 종료



해당 인덱스의 수열 원소값 출력

최단 거리 역추적

- 최단 거리를 구하기까지의 경로를 역으로 추적하는 것
- 기존 BFS와 동일한 풀이에 경로를 저장하는 부분만 추가
- 도착 정점에서 시작해서 저장한 경로값을 따라 추적

배열 활용

- 경로 배열을 만들어서 탐색하는 자식노드에 부모노드의 위치를 저장
- 노드를 배열의 인덱스로 표현할 수 있는 경우 사용

큐 활용

- 탐색에 사용하는 큐를 pair로 만들어 누적 경로를 저장
- 메모리가 많이 소요될 수 있음

/<> 13913번 : 숨바꼭질 4 - Gold 4

문제

- 수빈이는 동생과 숨바꼭질을 하고 있음
- 수빈이는 $X-1$, $X+1$, $2*X$ 로 이동 가능 (모두 1초 소요)
- 수빈이와 동생의 위치가 주어질 때, 수빈이가 동생을 찾을 수 있는 가장 빠른 시간이 몇 초 후인지 구하는 문제
- 어떻게 이동해야 하는지도 출력

제한 사항

- 수빈이의 위치 N은 $0 \leq N \leq 100,000$
- 동생의 위치 K는 $0 \leq K \leq 100,000$

예제 입력 1

5 17

예제 출력 1

4
5 10 9 18 17

예제 입력 2

5 17

예제 출력 2

4
5 4 8 16 17

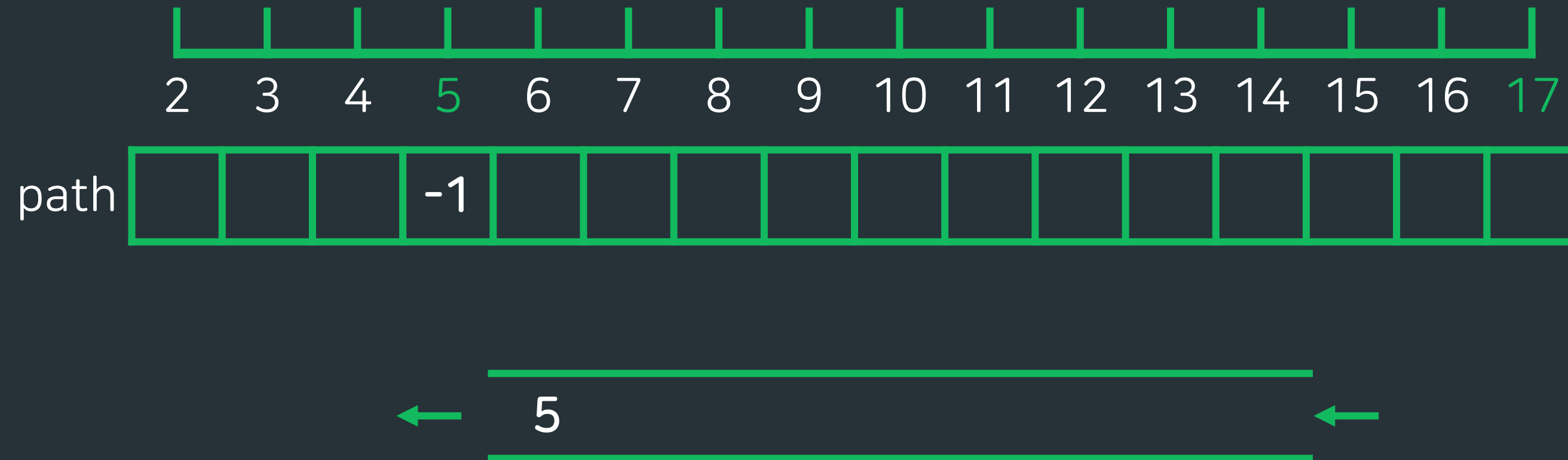
/<> 1697번 : 숨바꼭질 - Silver 1

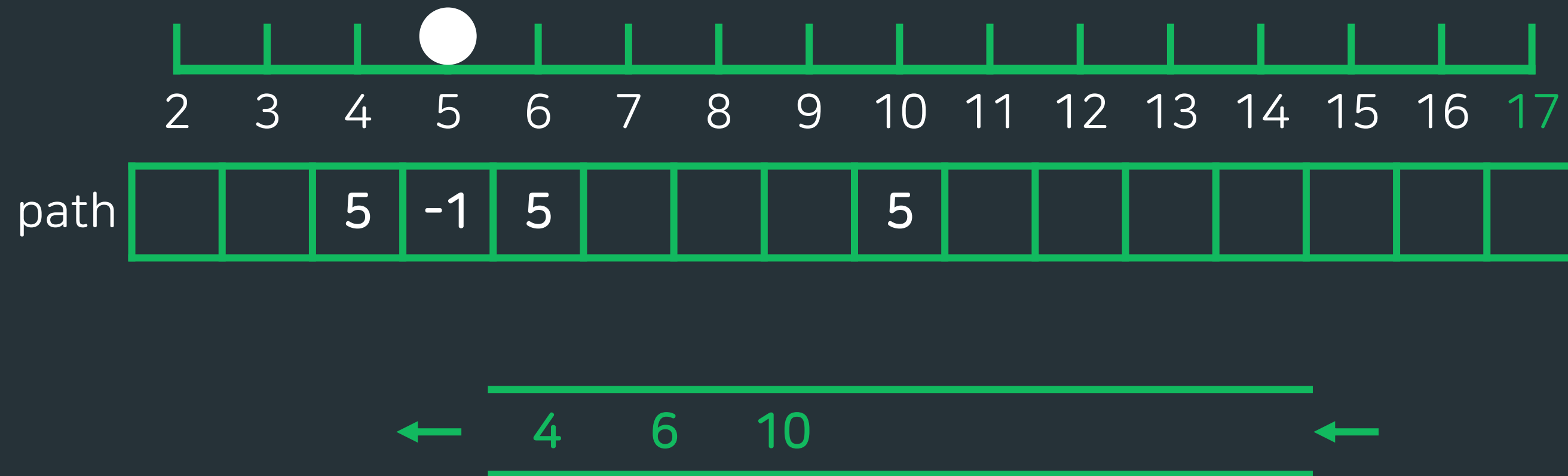
문제

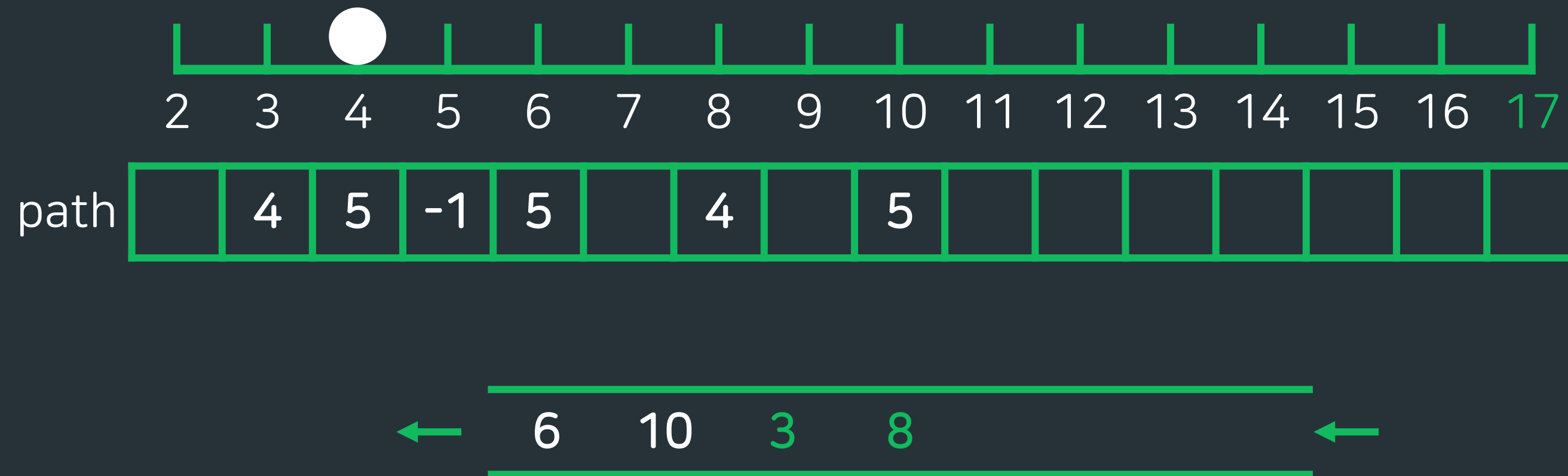
- 수빈이는 동생과 숨바꼭질을 하고 있음
- 수빈이는 $X-1$, $X+1$, $2*X$ 로 이동 가능 (모두 1초 소요)
- 수빈이와 동생의 위치가 주어질 때, 수빈이가 동생을 찾을 수 있는 가장 빠른 시간이 몇 초 후인지 구하는 문제

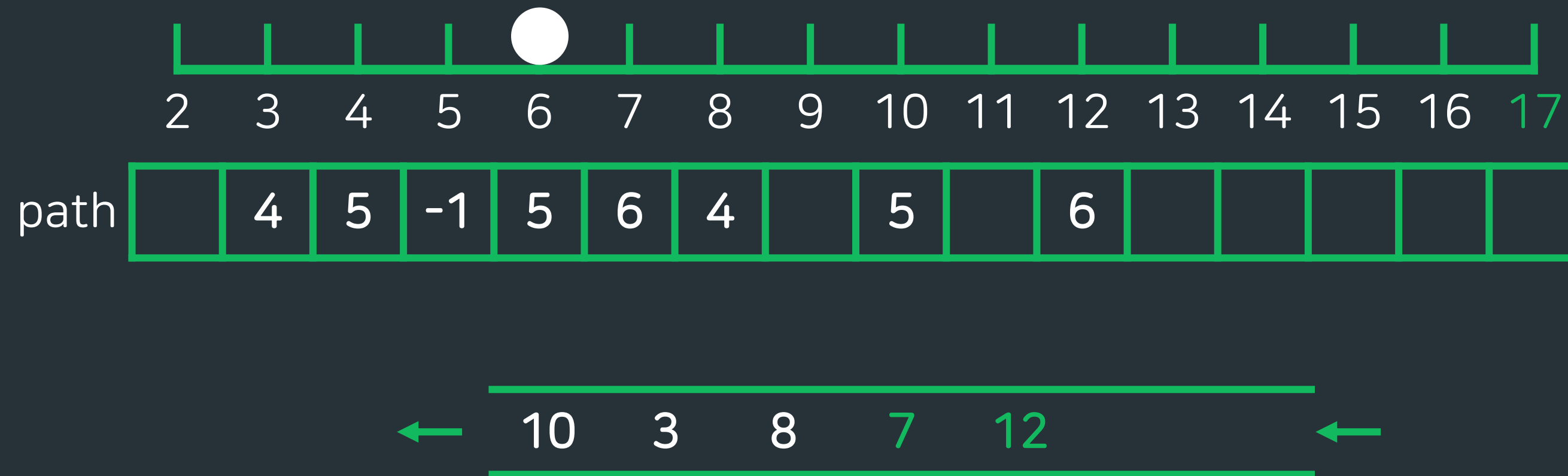
제한 사항

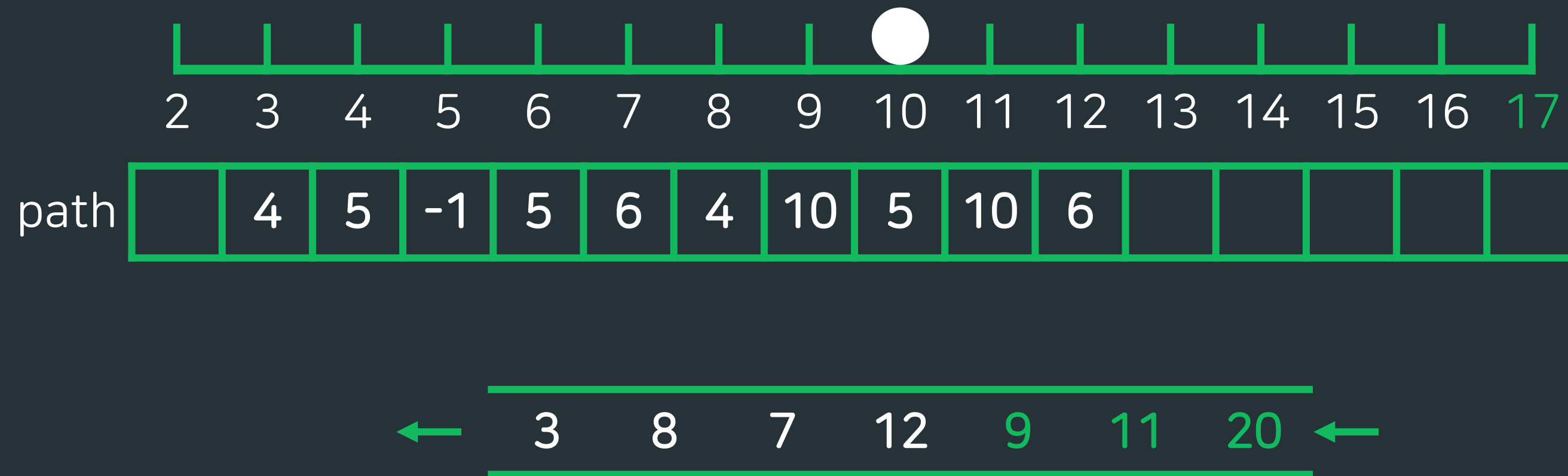
- 수빈이의 위치 N은 $0 \leq N \leq 100,000$
- 동생의 위치 K는 $0 \leq K \leq 100,000$

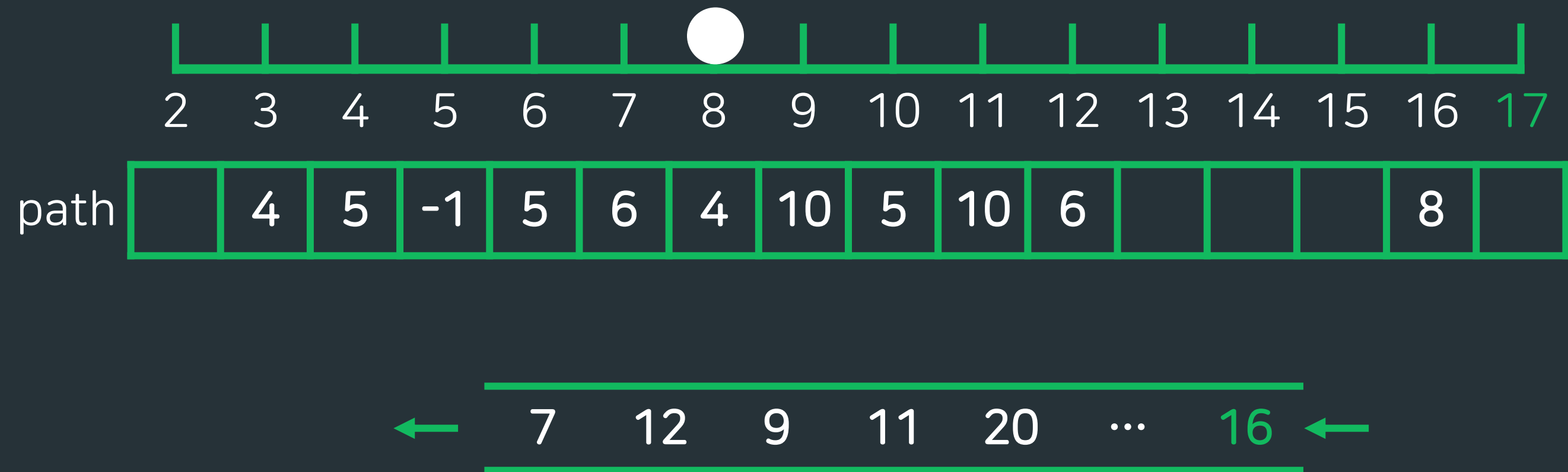


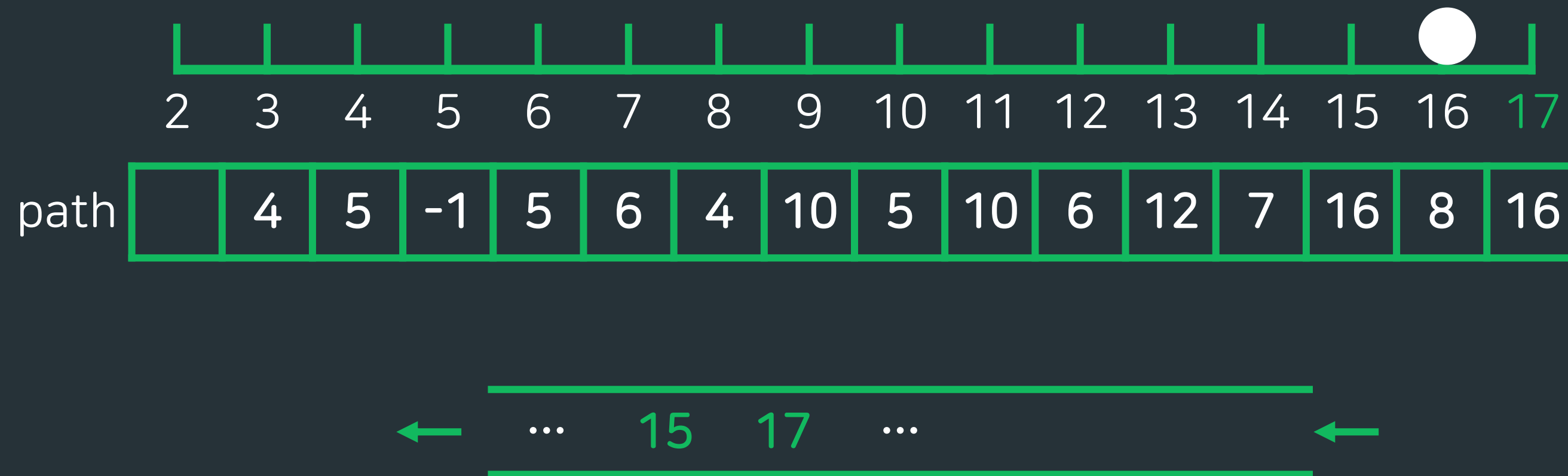


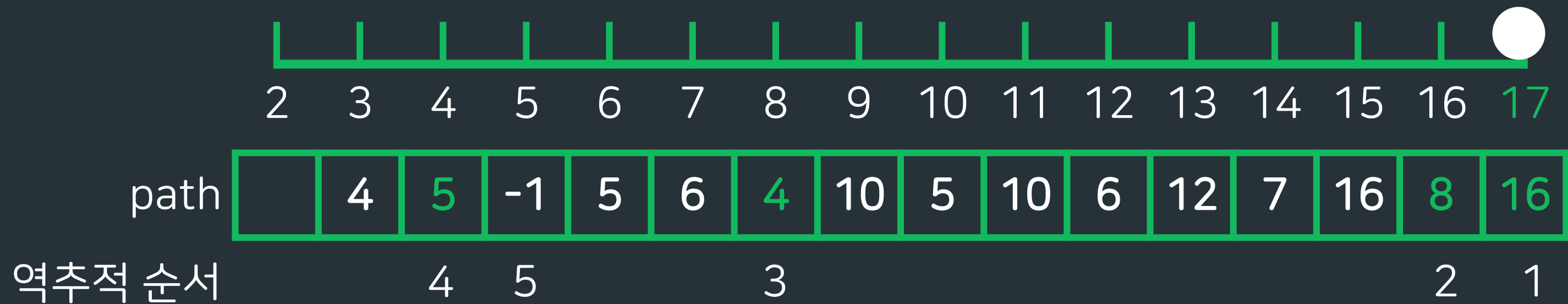












→ 5 → 4 → 8 → 16 → 17

/<> 11779번 : 최소비용 구하기 2 - Gold 3

문제

- 한 도시에서 출발하여 다른 도시에 도착하는 m 개의 버스가 있다
- A번째 도시에서 B번째 도시까지 가는데 드는 버스 비용을 최소화 하는 문제
- 최소 비용과 경로를 출력하라 (항상 경로 존재한다 가정)

제한 사항

- 도시의 개수 n 은 $1 \leq n \leq 1,000$
- 버스의 개수 m 은 $1 \leq m \leq 100,000$
- 버스 비용 c 는 $0 \leq C \leq 100,000$

예제 입력 1

```
5 8
1 2 2
1 3 3
1 4 1
1 5 10
2 4 2
3 4 1
3 5 1
4 5 3
1 5
```

예제 출력 1

```
4
3
1 3 5
```

Hint

1. 우선 **최소 비용**만 구하는 건 이미 어떻게 해야 할지 배웠어요
2. **경로**는 어떻게 저장할 수 있을까요? 지금까지 저장한 방식과 거의 같아요!

/<> 1916번 : 최소비용 구하기 - Gold 5

문제

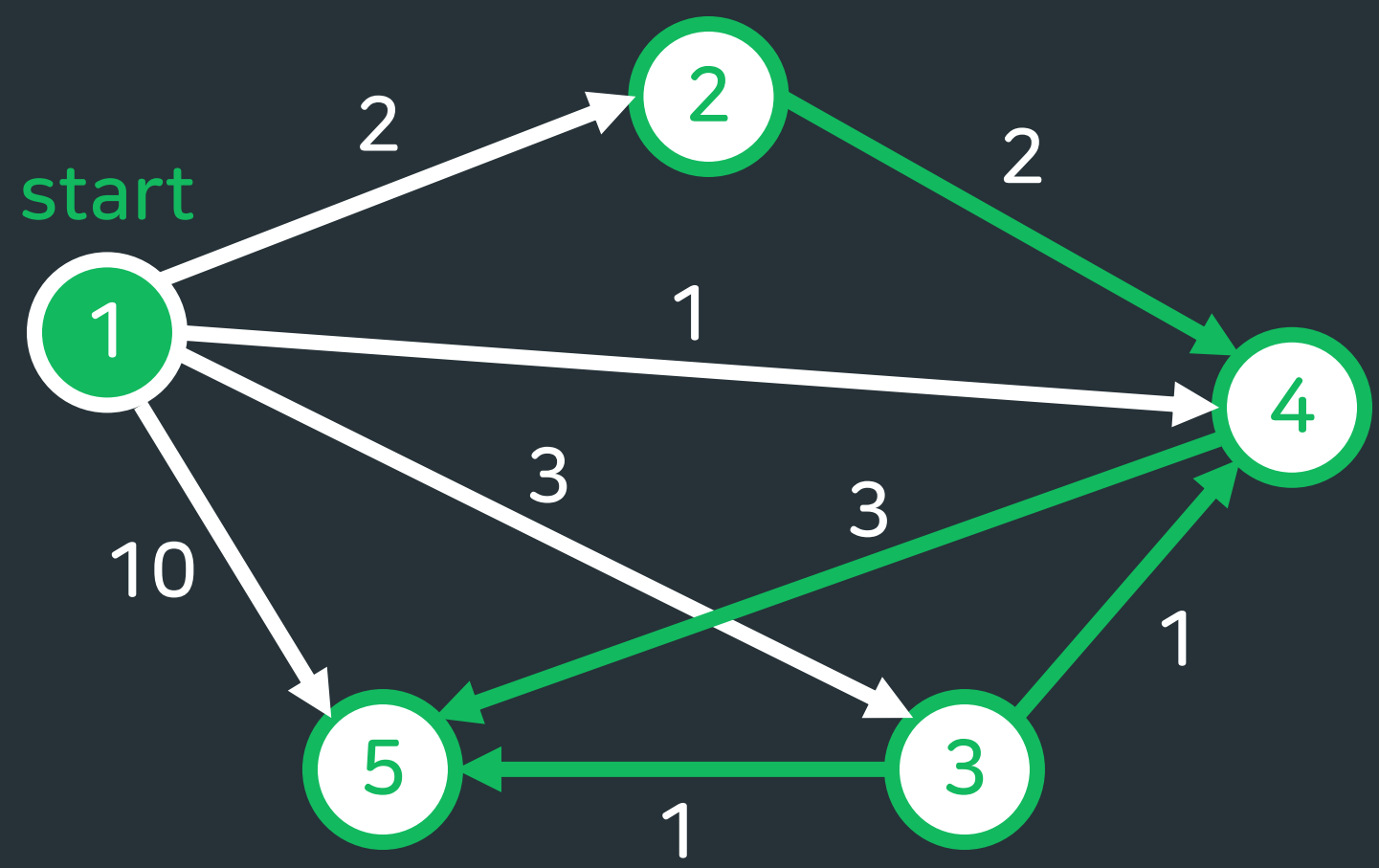
- 한 도시에서 출발하여 다른 도시에 도착하는 m 개의 버스가 있다
- A번째 도시에서 B번째 도시까지 가는데 드는 버스 비용을 최소화 하는 문제
- 최소 비용을 출력하라 (항상 도착점까지 갈 수 있다 가정)

제한 사항

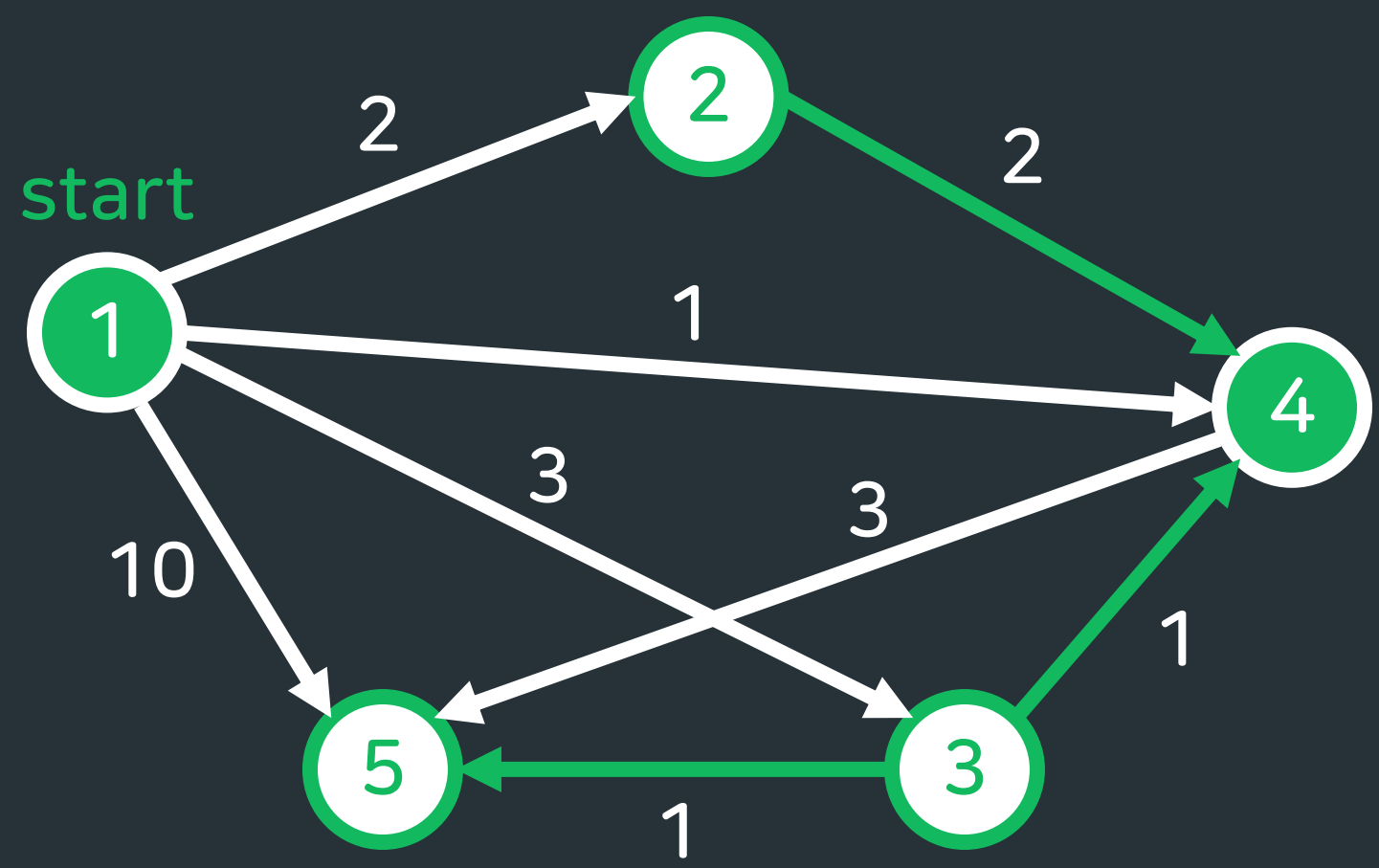
- 도시의 개수 n 은 $1 \leq n \leq 1,000$
- 버스의 개수 m 은 $1 \leq m \leq 100,000$
- 버스 비용 c 는 $0 \leq C \leq 100,000$

최단 경로 역추적

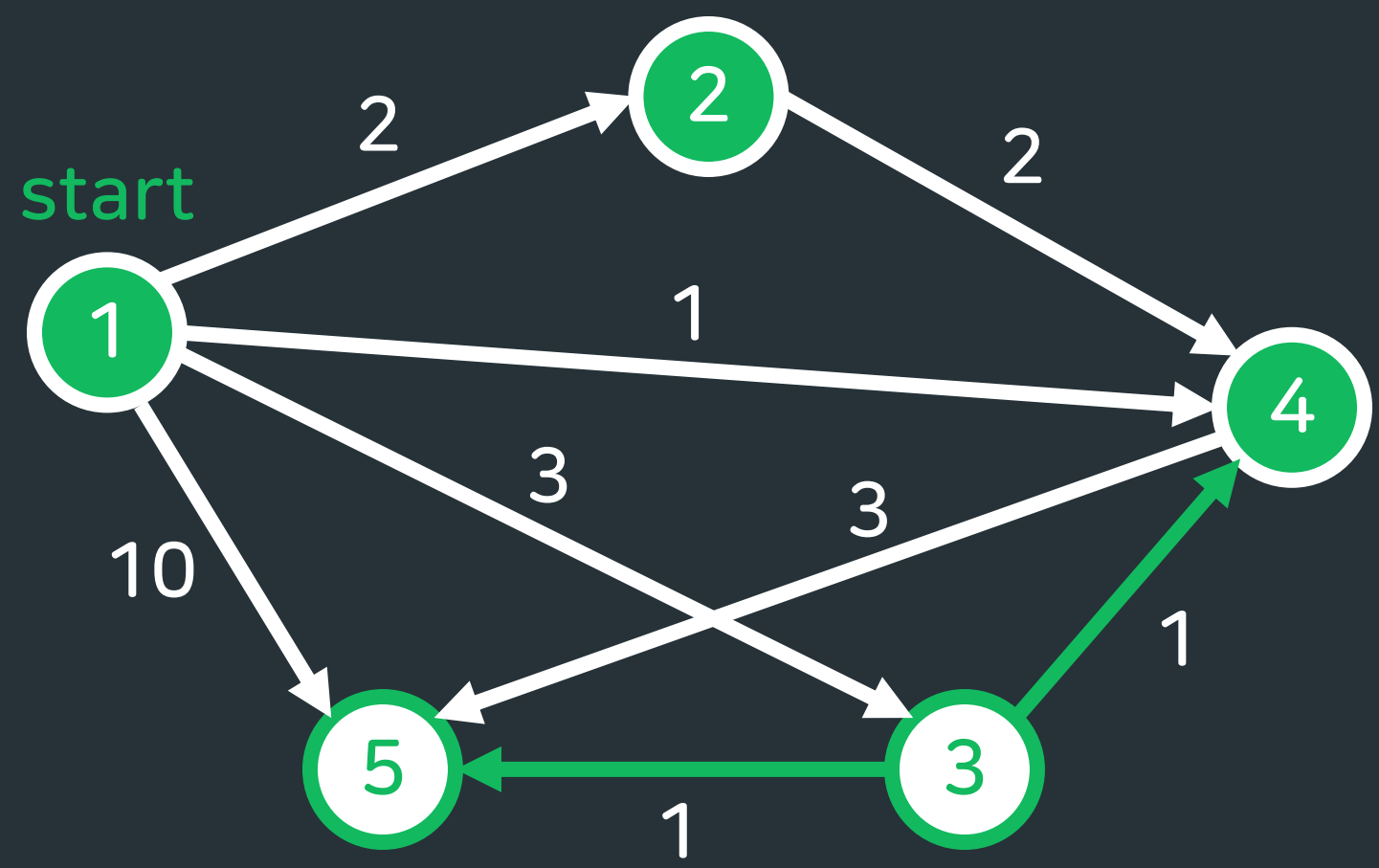
- 최단 경로의 값을 구하기까지의 경로를 역으로 추적하는 것
- 기존 최단 경로와 동일한 풀이에 경로를 저장하는 부분만 추가
- 도착 정점에서 시작해서 저장한 경로값을 따라 추적



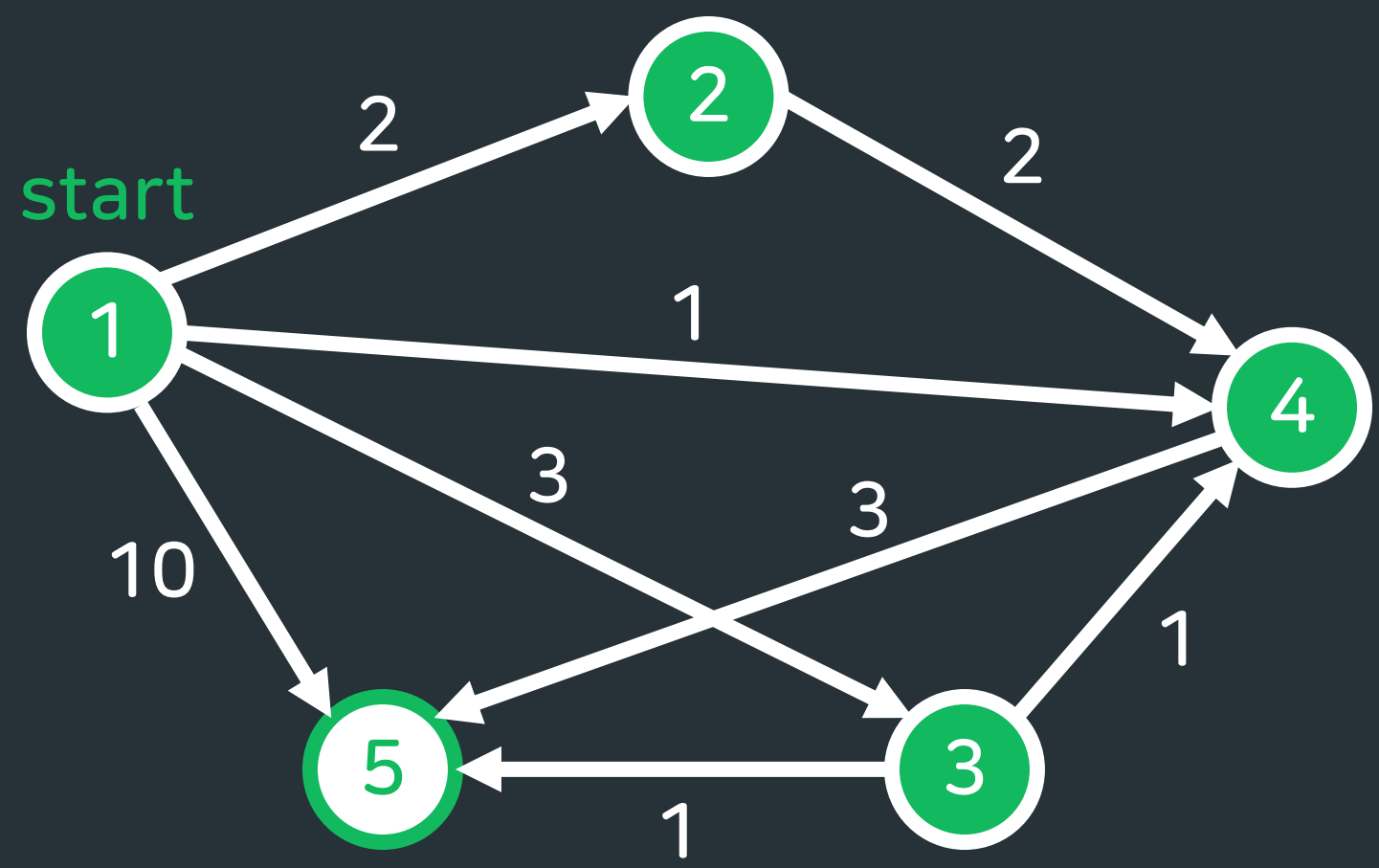
| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|----|
| dist | 0 | 2 | 3 | 1 | 10 |
| path | 0 | 1 | 1 | 1 | 1 |



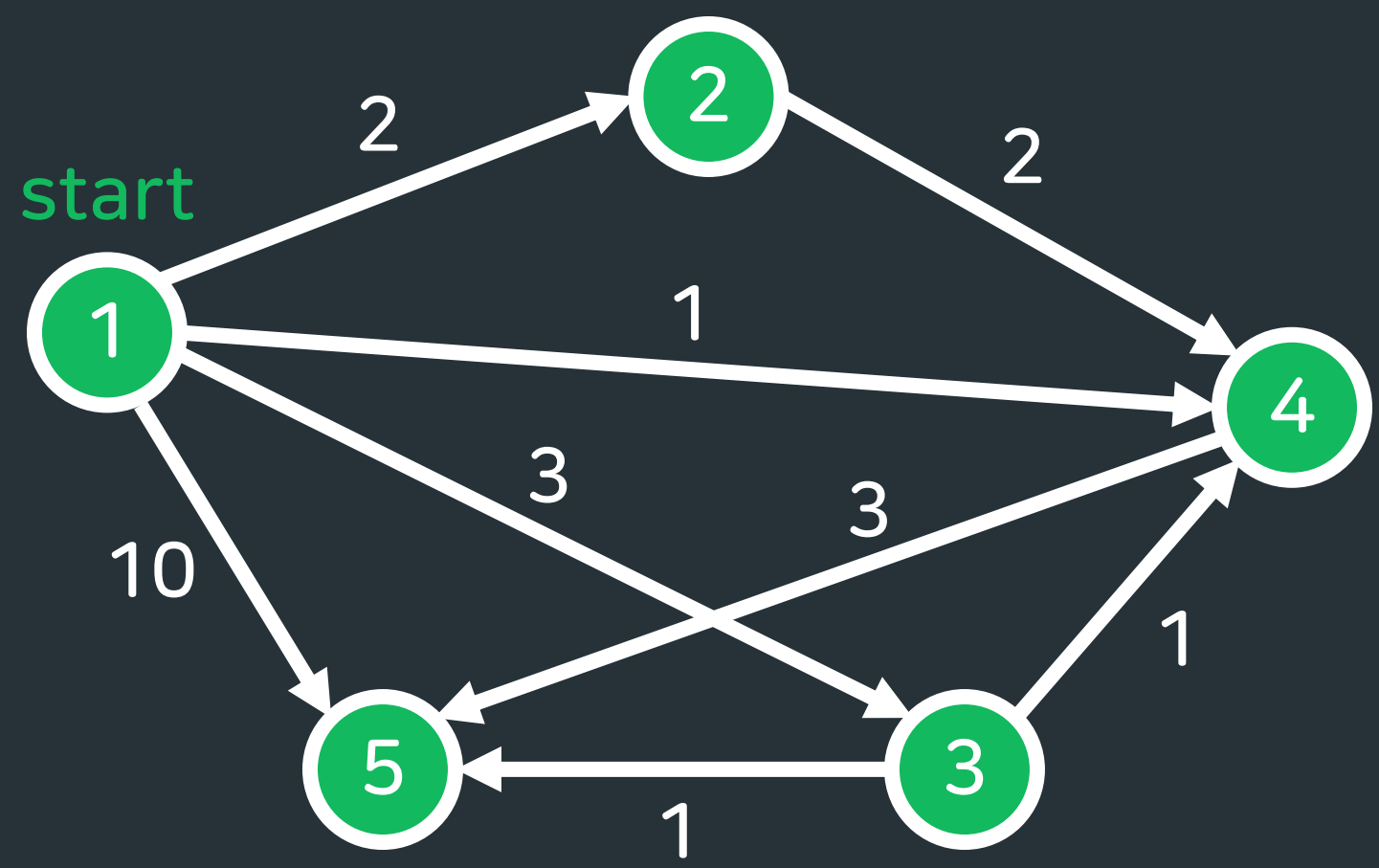
| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| dist | 0 | 2 | 3 | 1 | 4 |
| path | 0 | 1 | 1 | 1 | 4 |



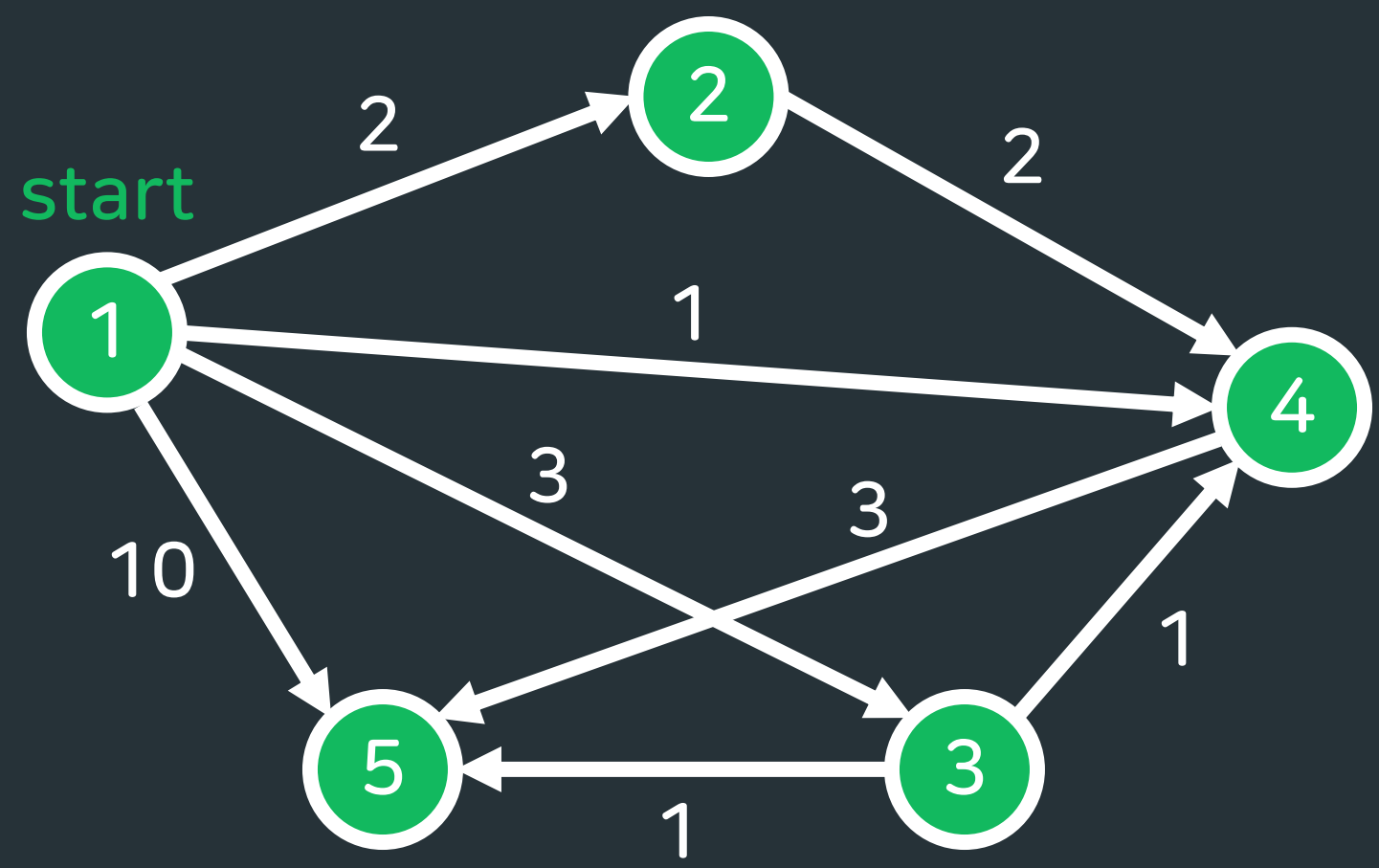
| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| dist | 0 | 2 | 3 | 1 | 4 |
| path | 0 | 1 | 1 | 1 | 4 |



| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| dist | 0 | 2 | 3 | 1 | 4 |
| path | 0 | 1 | 1 | 1 | 4 |



| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| dist | 0 | 2 | 3 | 1 | 4 |
| path | 0 | 1 | 1 | 1 | 4 |



| | 1 | 2 | 3 | 4 | 5 |
|------|---|---|---|---|---|
| dist | 0 | 2 | 3 | 1 | 4 |
| path | 0 | 1 | 1 | 1 | 4 |

정리

- 역추적은 기존 알고리즘에서 구한 해의 경로를 추적하는 것
- 기존 풀이에 경로를 저장하는 부분과 추적하는 부분만 추가하면 됨
- 주로 동적 계획법, 최단 거리(BFS), 최단 경로(다익스트라, 플로이드-워셜)와 같이 나옴
- 경로 저장은 배열로 구현
- 나중에 경로를 추적해야 하므로 위치(인덱스)를 저장

이것도 알아보세요

- 최단 거리 역추적 문제는 경로를 큐를 통해 저장할 수도 있어요. 배열로 경로를 저장하는 것과 다른 점이 무엇일까요?

필수

/<> 15683번 : 감시 - Gold 5

/<> 2615번 : 오목 - Silver 3

3문제 이상 선택

/<> 9252번 : LCS 2 - Gold 5

/<> 1719번 : 택배 - Gold 4

/<> 11780번 : 플로이드 2 - Gold 3

/<> 9019번 : DSLR - Gold 5

/<> 12852번 : 1로 만들기 2 - Silver 1

/<> 16964번 : DFS 스페셜 저지 - Gold 4