

세그먼트 트리

최백준 choi@startlink.io

RMQ

구간의 최소값 구하기

Range Minimum Query

- 배열 $A[1], \dots, A[N]$ 가 있고, 다음과 같은 연산을 수행해야 한다.

- 최소값: $A[i] \dots A[j]$ 중에서 최소값을 찾아 출력한다.
- 이러한 연산이 총 Q 개 주어진다.

(i, j)

1개 $\rightarrow O(N)$
 Q 개 $\rightarrow O(N \cdot Q)$

$N, Q \leq 10^5$ 만

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
3	6	2	5	3	1	8	9	7	3	5

다 해보기

다 해보기

구간의 최소값 구하기 (RMQ)

- 최소값: $A[i], \dots, A[j]$ 중에서 최소값을 찾아 출력한다. $O(N)$

```
int m = a[i];  
for (int k=i; k<=j; k++) {  
    if (m > a[k]) {  
        m = a[k];  
    }  
}
```

다 해보기

구간의 최소값 구하기 (RMQ)

- 최소값을 하나 구하는데 $O(N)$ 시간이 걸린다.
- 쿼리의 개수는 총 Q 개이기 때문에, $O(NQ)$ 의 시간이 필요하다.

N 개의 수 \rightarrow

\sqrt{N} 개의 $\sqrt{2}$

7

등이 있는 수의 개수

\sqrt{N} 개

루트N으로 나누기

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- N개를 루트 N개의 그룹으로 나누면 그룹에 포함된 수의 개수는 루트 N개이다.
- 그룹의 개수와
- 그룹에 포함된 수의 개수가
- 같다는 점을 이용한다.

~~Sqrt~~ decomposition
↓
Square root
√

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 영어로는 sqrt decomposition 이라고 한다.
- $R = \sqrt{N}$ 이라고 했을 때
- A 를 R 개의 그룹으로 나눈 다음에, $Group[i]$ 에 i 번 그룹의 최소값을 저장하는 방식

$N = 11$ $\sqrt{11} = 3.$

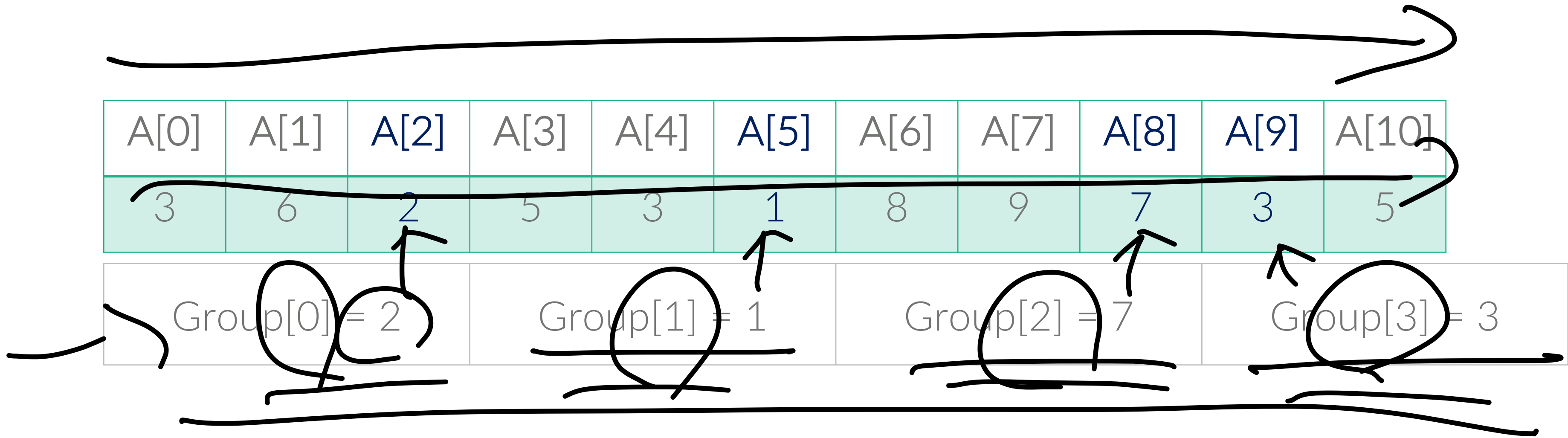
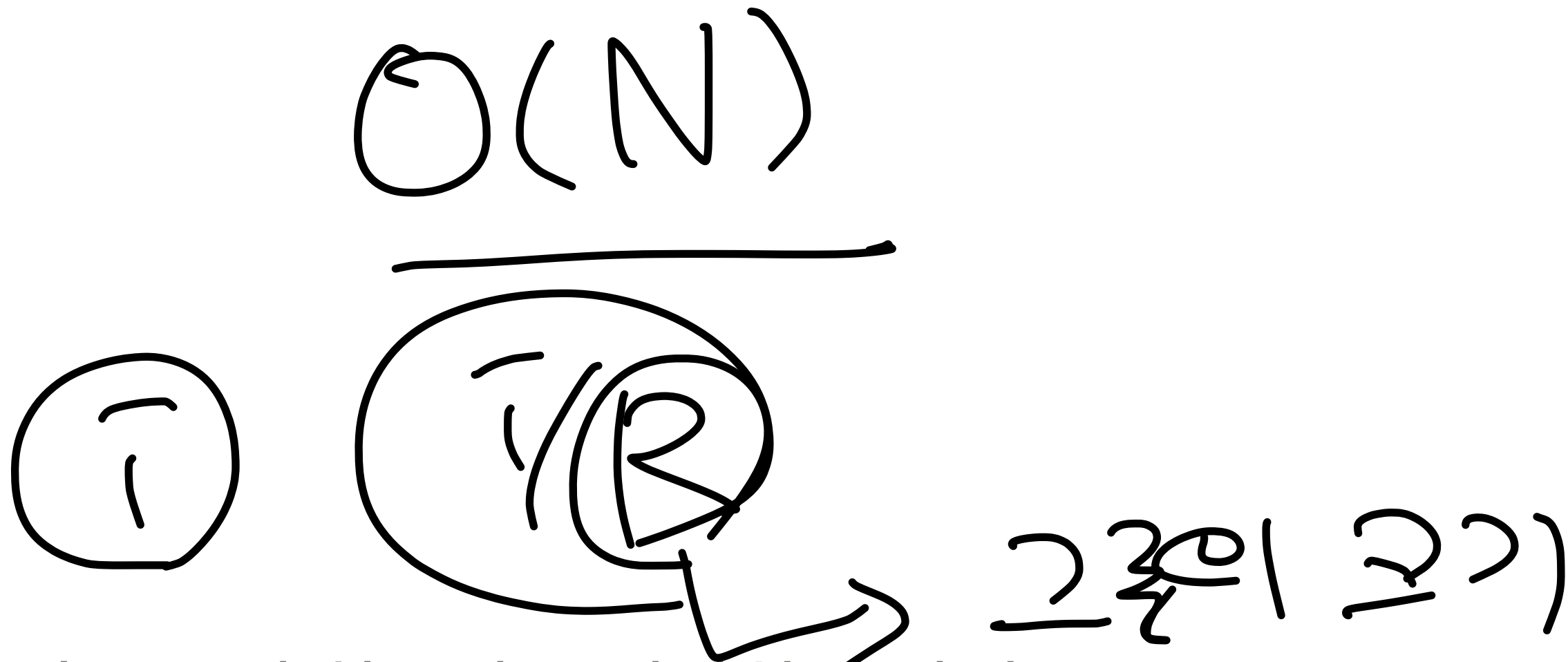
$R = 3$

	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
	3	6	2	5	3	1	8	9	7	3	5
그룹	0			1			2		3		

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 영어로는 sqrt decomposition 이라고 한다.
- $R = \sqrt{N}$ 이라고 했을 때
- A 를 R 개의 그룹으로 나눈 다음에, $Group[i]$ 에 i 번 그룹의 최소값을 저장하는 방식



루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 영어로는 sqrt decomposition 이라고 한다.
- $R = \sqrt{N}$ 이라고 했을 때
- A 를 R 개의 그룹으로 나눈 다음에, $Group[i]$ 에 i 번 그룹의 최소값을 저장하는 방식

선처리
preprocessing, $O(N)$

```
for (int i=0; i<n; i++) {
    if (i%r == 0) {
        group[i/r] = a[i];
    } else {
        group[i/r] = min(group[i/r], a[i]);
    }
}
```

2개의
그룹으로 나누기

$[1, 3)$

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

그룹의 크기:



- 최소값을 구하는 쿼리 i, j 는 두 가지 경우가 있다. ($i \leq j$)

1. i 와 j 가 같은 그룹인 경우
2. i 와 j 의 그룹이 다른 경우

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
3	6	2	5	3	1	8	9	7	3	5
G[0] = 2			G[1] = 1			G[2] = 7			G[3] = 3	

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 최소값을 구하는 쿼리 i, j 는 두 가지 경우가 있다. ($i \leq j$)

1. i 와 j 가 같은 그룹인 경우

- 이 경우에는 그룹에 포함된 수를 모두 최소값 비교할 수 있다.
- 그룹에 들어있는 수의 개수는 루트 N 개이기 때문에, 총 걸리는 시간은 $O(\text{루트 } N)$ 이다.

```
for (int i=start; i<=end; i++) {  
    ans = min(ans, a[i]);  
}
```

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 최소값을 구하는 쿼리 i, j 는 두 가지 경우가 있다. ($i \leq j$)

2. i 와 j 가 다른 그룹인 경우

- 이 경우에는 3가지로 나눌 수 있다.

- i 가 들어있는 그룹
- j 가 들어있는 그룹
- i 와 j 사이에 들어있는 그룹

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

$\bar{r}=1, \bar{s}=9$

- A[1] ~ A[9]의 최소값을 구하는 경우
- N = 11
- R = 루트N = 3
- 1의 그룹: $1/R = 1/3 = 0$
- 9의 그룹: $9/R = 9/3 = 3$

- 시작 그룹에 들어있는 수의 개수는 R개
- 끝 그룹에 들어있는 수의 개수는 R개
- 시작과 끝 그룹 사이에 있는 그룹의 수는 R개
- 아래 표시된 값을 비교해야 한다.

\sqrt{N}
 \sqrt{N}
 \sqrt{N}
 $O(\sqrt{N})$

\sqrt{N} 개

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
3	6	2	5	3	1	8	9	7	3	5
G[0] = 2			G[1] = 1			G[2] = 7			G[3] = 3	

루트N으로 나누기

구간의 최소값 구하기 (RMQ)

```
while (true) {  
    ans = min(ans, a[start]);  
    start += 1;  
    if (start % r == 0) {  
        break;  
    }  
}
```

그런데...

$$(N + Q\sqrt{N})$$

선처리: $O(N)$
쿼리: $O(\sqrt{N})$

```
while (true) {  
    ans = min(ans, a[end]);  
    end -= 1;  
    if (end % r == r-1) {  
        break;  
    }  
}  
  
for (int i=start/r; i<=end/r; i++) {  
    ans = min(ans, group[i]);  
}
```


루트N으로 나누기

구간의 최소값 구하기 (RMQ)

- 총 $O(3\sqrt{N})$ 의 시간이 걸리게 된다.

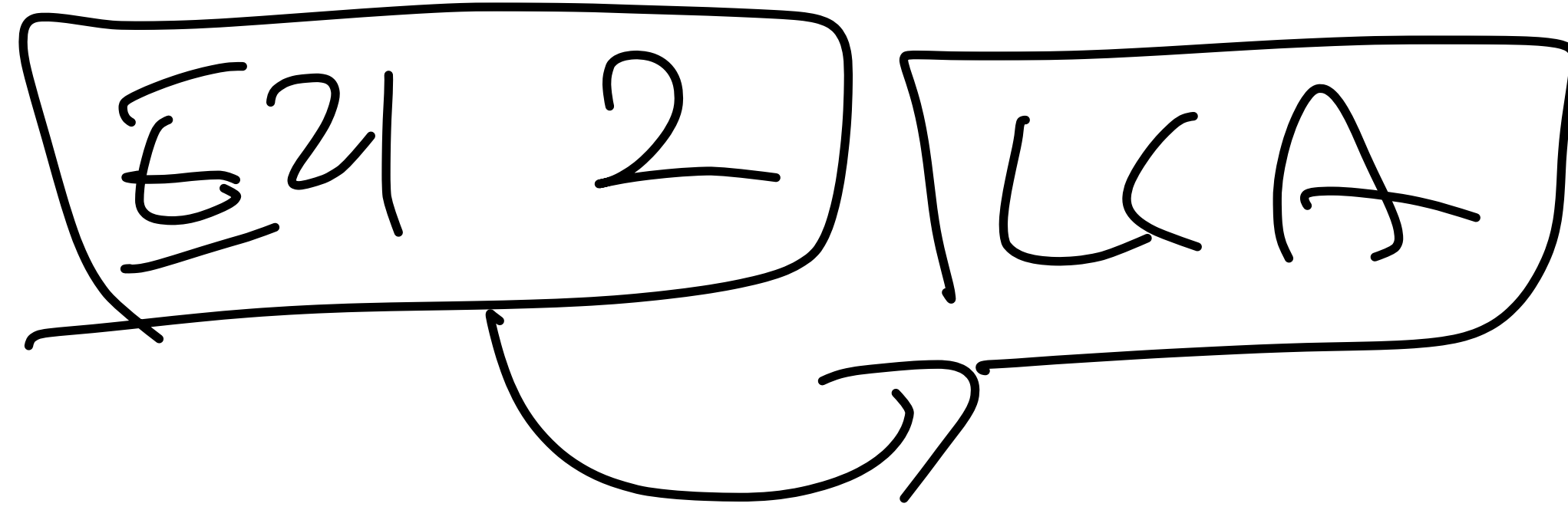
루트N으로 나누기

18

구간의 최소값 구하기 (RMQ)

- 공간: $O(N)$
- 선처리: $O(N)$
- 최소값 구하는 시간: $O(\sqrt{N})$

$$\hat{z} = \hat{z}^{-1} + \hat{z}^{-1}$$



다이나믹 프로그래밍

PC [S]

$$7 = 1012 \frac{2}{10}$$

11. 1012512401

$$2\frac{5}{6} \quad 3\frac{1}{6}$$

1) PTPCPC-PPCP

미
AC[4]부터 2개의 최소

-

The diagram shows a 2D array D with dimensions 9x4. The elements are arranged in a grid:

$D[0][0]$	$D[3][0]$		
$D[0][1]$	$D[4][1]$		$D[8][1]$
$D[0][2]$			$D[7][2]$
$D[0][3]$			

Handwritten annotations include:

- A circle around $D[4][1]$.
- An arrow pointing to $D[4][1]$ from above.
- The number "4" written next to the circle.
- The number "1" written below the circle.
- The number "2" written to the right of the circle.

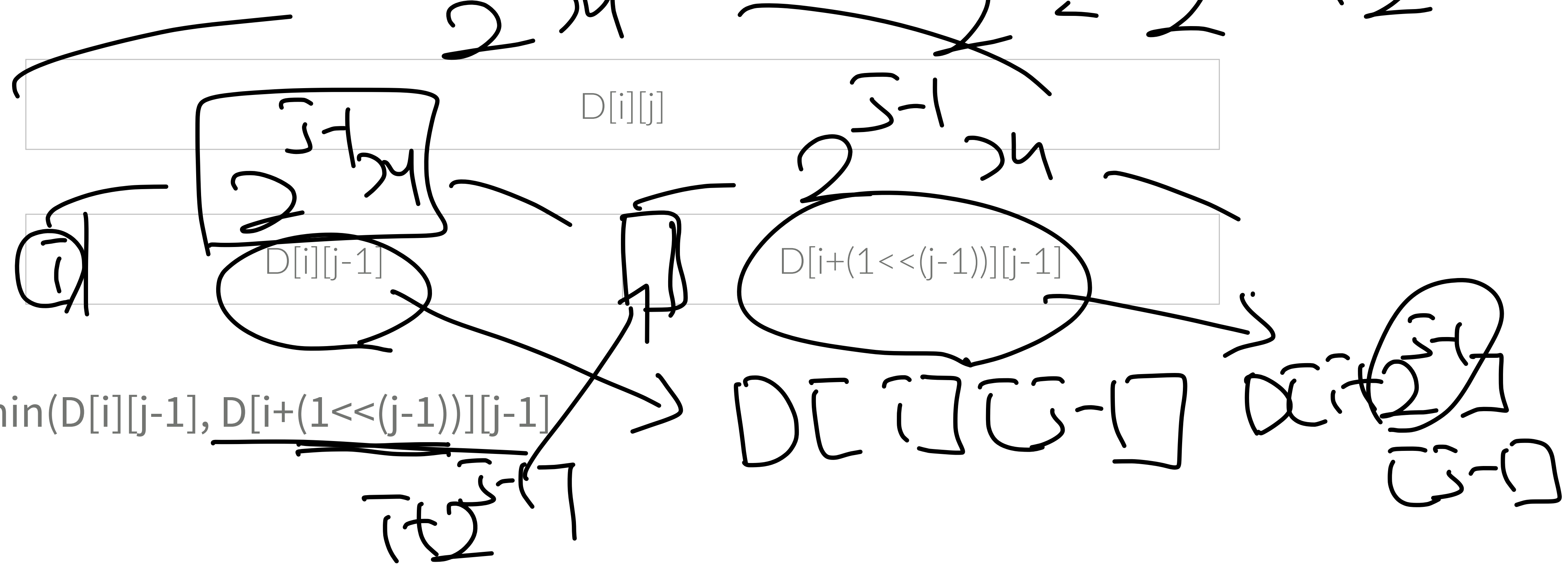
다이나믹 프로그래밍 $[N \times N]$ 공간, 1인차리

구간의 최소값 구하기 (RMQ)

- $D[i][j] = A[i]$ 부터 2^j 개의 최소값
- $A[i]$ 부터 2^j 개의 최소값은
- $A[i]$ 부터 2^{j-1} 개의 최소값과 $A[i+2^{j-1}]$ 부터 2^{j-1} 개의 최소값과 같다.

$D[i][j] = [부터 2^j개의 최소]$

$2^j = 2^{j-1} + 2^{j-1}$

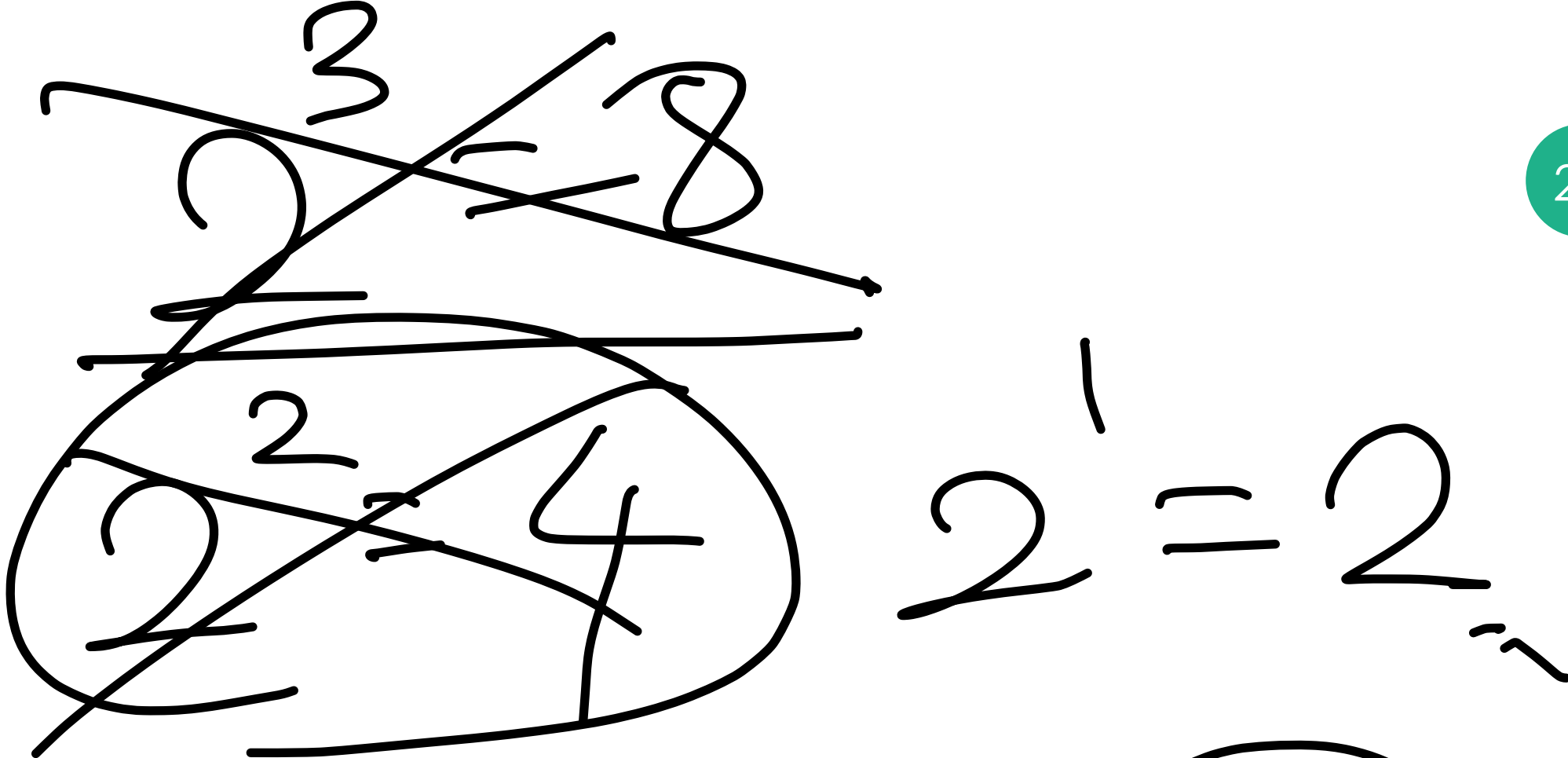


$D[i][j] = \min(D[i][j-1], D[i+(1 \ll (j-1))][j-1])$

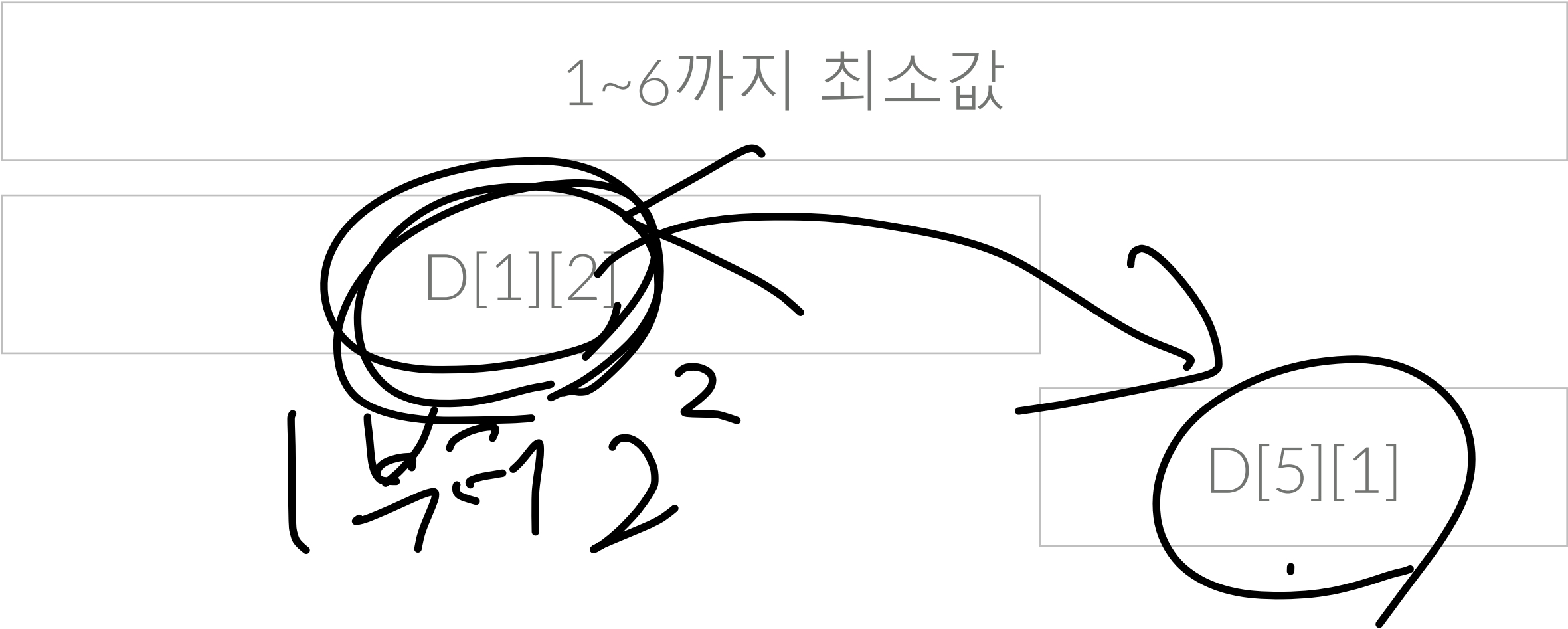
다이나믹 프로그래밍

구간의 최소값 구하기 (RMQ)

1부터 6까지 최소값 구하기



A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
3	6	2	5	3	1	8	9	7	3	5



다이나믹 프로그래밍

구간의 최소값 구하기 (RMQ)

- 2부터 8까지 최소값 구하기

2^2
 $O(\log N)$

23

A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]	A[10]
3	6	2	5	3	1	8	9	7	3	5

2~8까지 최소값

D[2][2]

D[6][1]

D[8][0]

다이나믹 프로그래밍

구간의 최소값 구하기 (RMQ)

- 공간: $O(N \lg N)$
- 선처리: $O(N \lg N)$
- 최소값 구하는 시간: $O(\lg N)$

$$N \lg N + Q \lg N$$

$$2 \sqrt{N}$$

다이나믹 프로그래밍

구간의 최소값 구하기 (RMQ)

```
for (int i=0; i<n; i++) {  
    d[i][0] = a[i];  
}  
for (int j=1; j<17; j++) {  
    for (int i=0; i<n; i++) {  
        if (i+(1<<j)-1 < n) {  
            d[i][j] = min(d[i][j-1], d[i+(1<<(j-1))][j-1]);  
        } else {  
            break;  
        }  
    }  
}
```

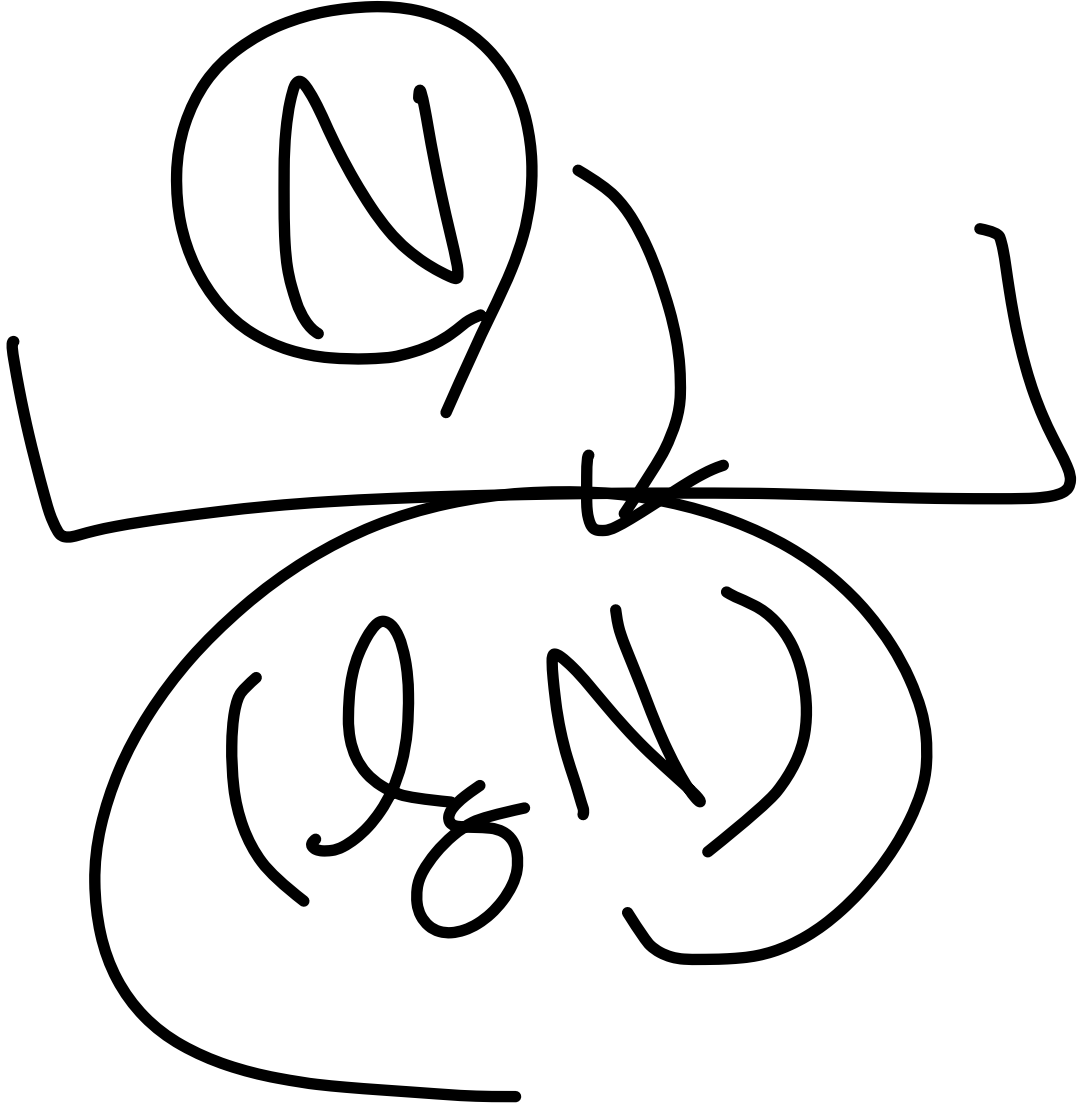
다이나믹 프로그래밍

구간의 최소값 구하기 (RMQ)

```
int ans = a[start];
int k = 16;
while (start <= end && k >= 0) {
    if (start + (1<<k) - 1 <= end) {
        ans = min(ans, d[start][k]);
        start += (1<<k);
    }
    k -= 1;
}
```

- 위의 소스에서 16은 문제의 N 제한이 10만이기 때문에, 임의로 정한 값이다.
- $2^{16} = 65536$ 이라, 2^{17} 크기를 가지는 경우는 없기 때문

세그먼트 트리

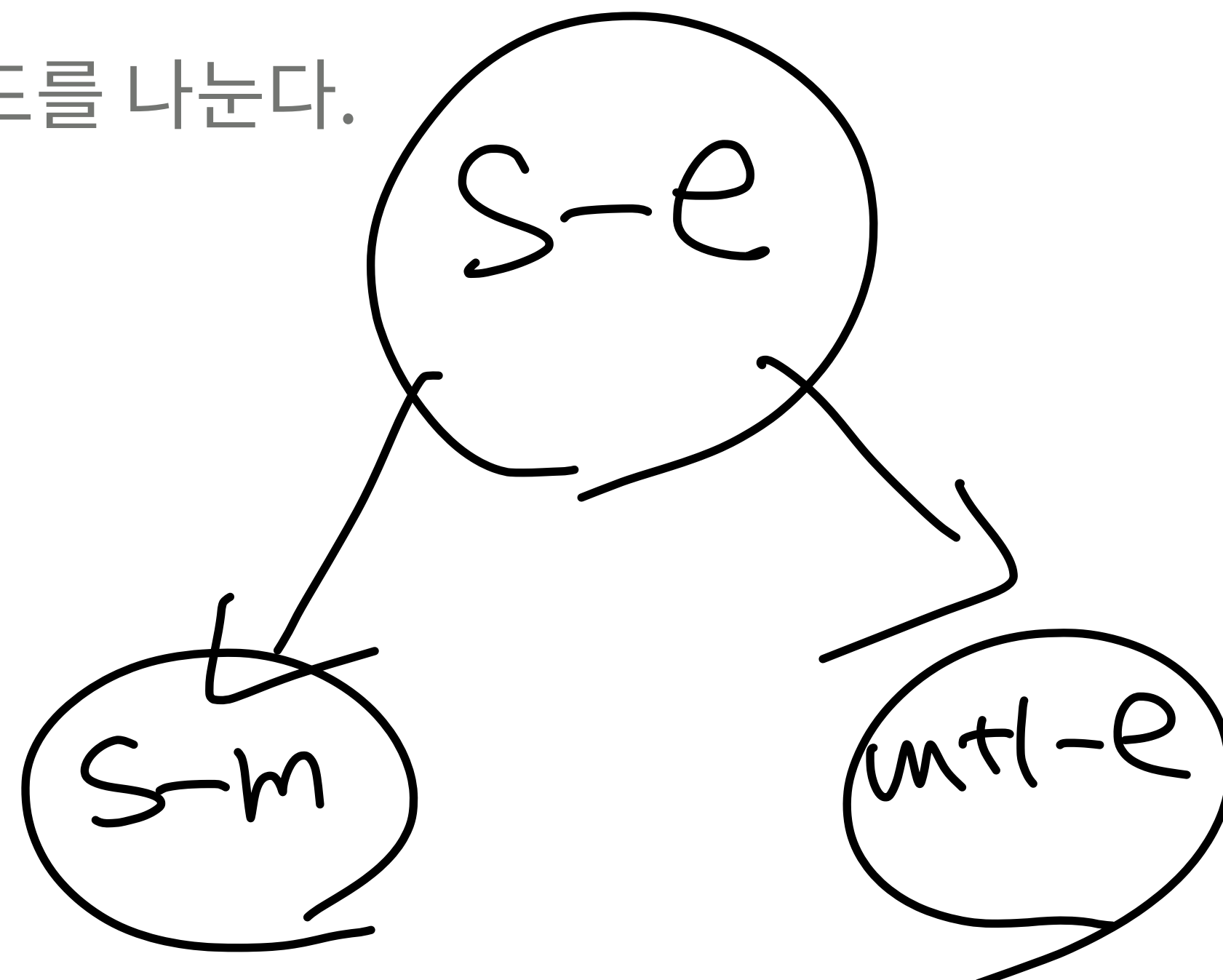


세그먼트 트리

구간의 최소값 구하기 (RMQ)

- 세그먼트 트리의 노드는 구간을 담당하고 있다.
- 노드가 담당하는 구간을 [start,end]라고 했을 때
- 노드는 [start,end]의 최소값을 가지고 있다.
- [start,end]의 최소값은 [start,mid]의 최소값과 [mid+1,end]의 최소값 중 최소값이다.
- 이 점을 이용해 왼쪽 자식과 오른쪽 자식으로 노드를 나눈다.

$$m = \frac{s + e}{2}$$



세그먼트 트리

구간의 최소값 구하기 (RMQ)

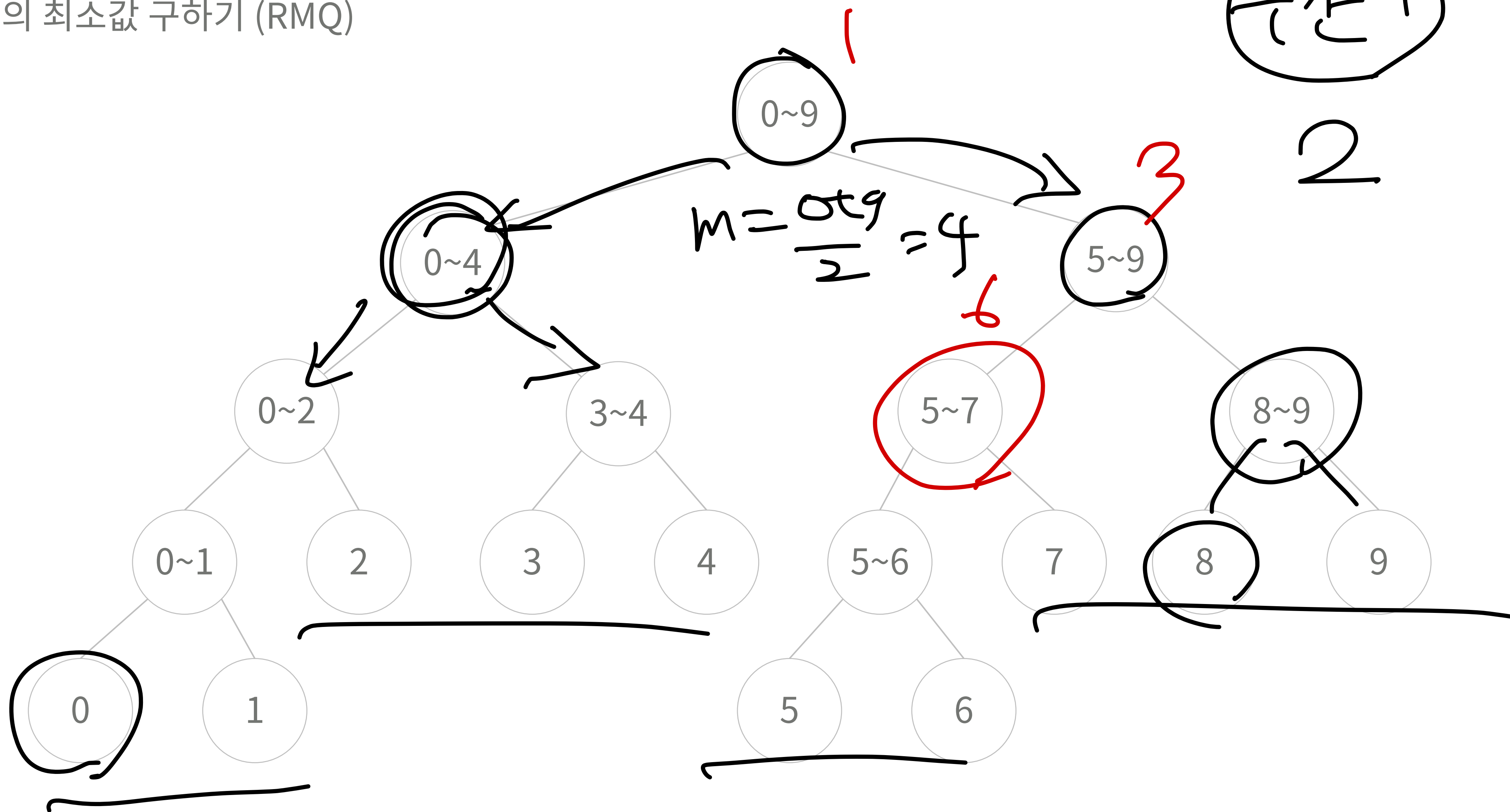
$N=10$

$0-9$

29

구간 1

2



세그먼트 트리

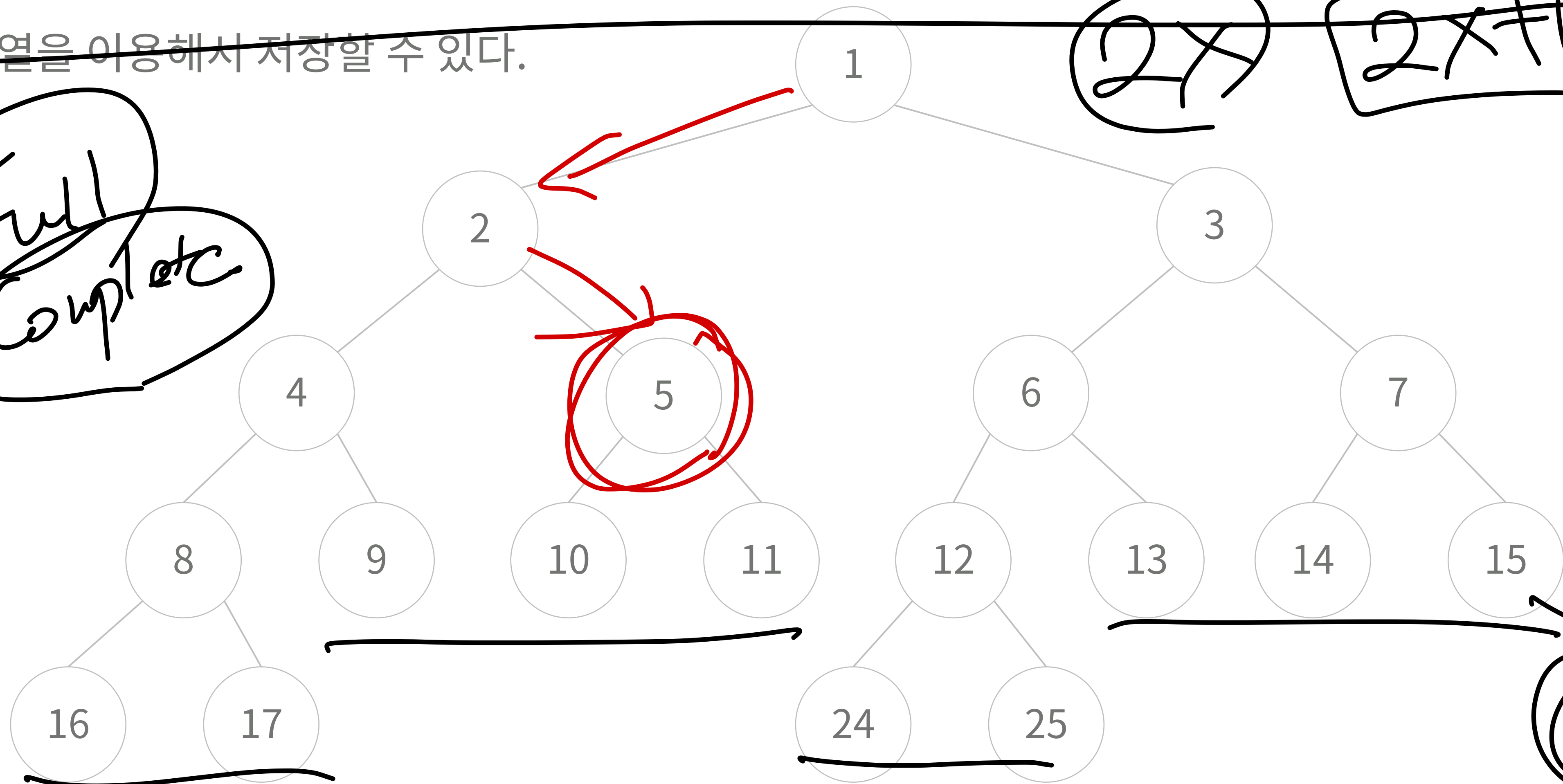
구간의 최소값 구하기 (RMQ)

Heap

30

- 배열을 이용해서 저장할 수 있다.

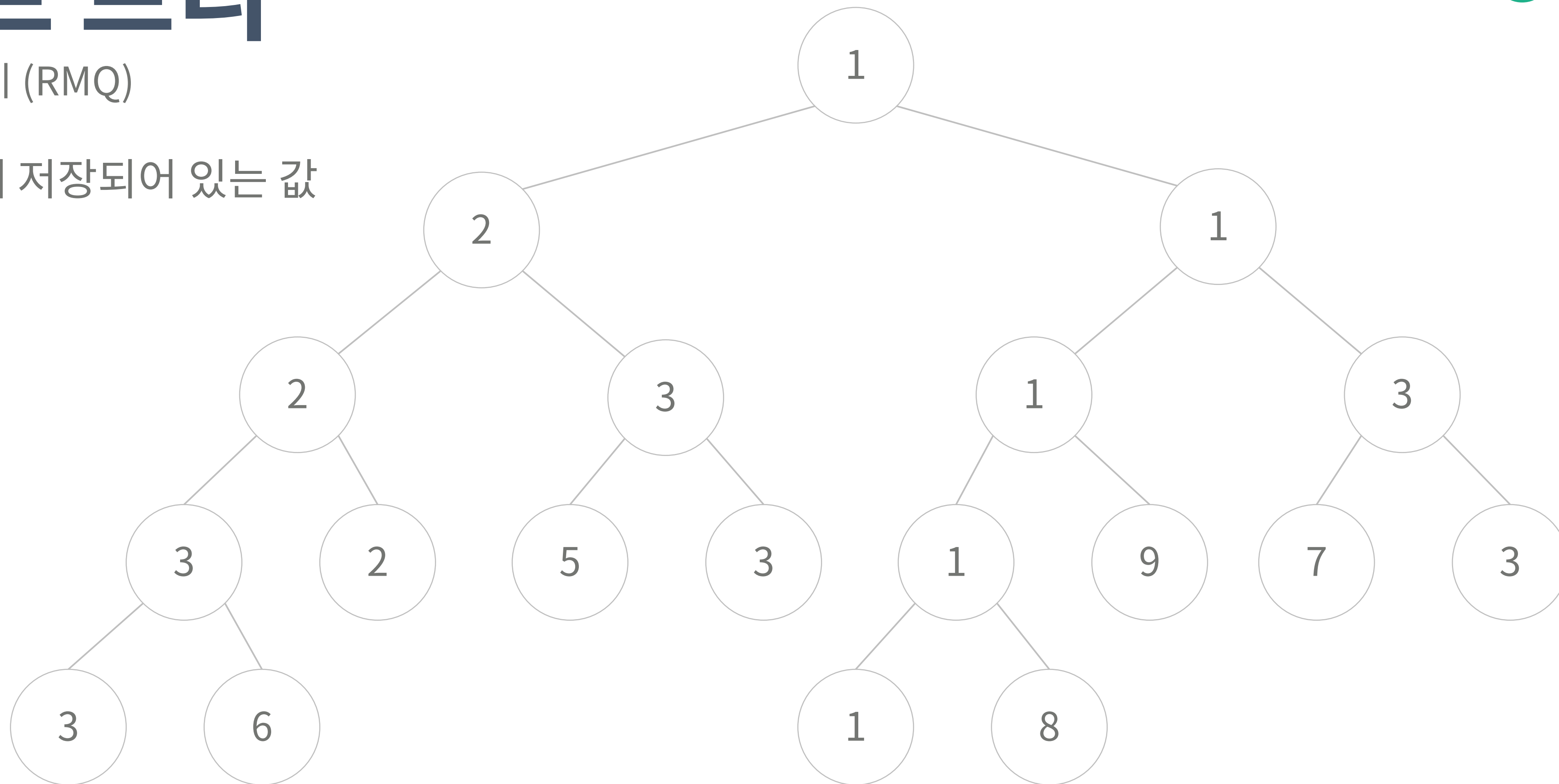
Full
Complete



세그먼트 트리

구간의 최소값 구하기 (RMQ)

- 세그먼트 트리에 저장되어 있는 값



A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]	A[9]
3	6	2	5	3	1	8	9	7	3

세그먼트 트리

구간의 최소값 구하기 (RMQ)

- 노드가 담당하는 구간의 크기가 $2K$ 라면
- 왼쪽 자식과 오른쪽 자식의 크기는 K 이다.
- 구간의 크기가 $2K+1$ 인 경우에는 왼쪽, 오른쪽 자식의 크기는 $K+1, K$ 가 된다.
- 따라서, 모든 리프 노드의 높이 차이는 0 또는 1이다.

Start
end

~~max~~
node

노드가
담당하는 구간

저장되는 배열
인덱스

세그먼트 트리

구간의 최소값 구하기 (RMQ)

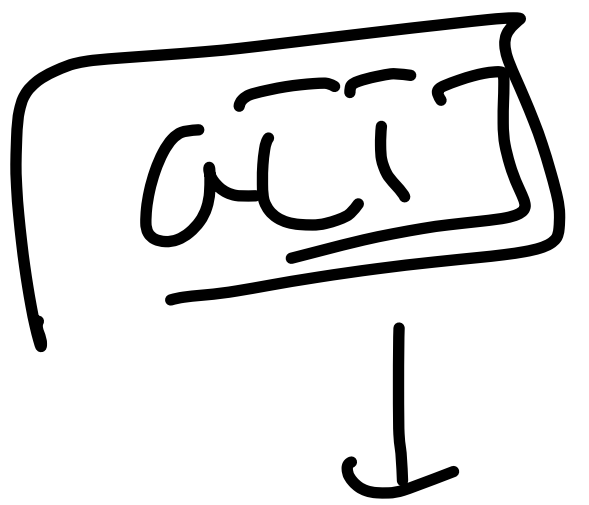
- N 이 2의 제곱꼴인 경우에는 Full Binary Tree
- 리프 노드가 N 개인 Full Binary Tree: 필요한 노드의 개수: $2N-1$
- 높이 $H = \lceil \lg N \rceil$ 이다.
- 필요한 배열의 크기: 2^{H+1}

세그먼트 트리

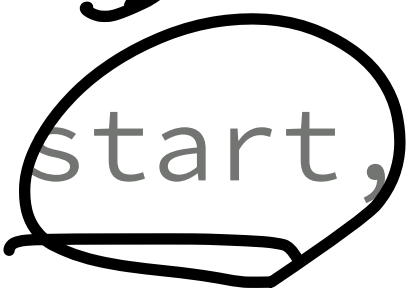
구간의 최소값 구하기 (RMQ)

```
void init(vector<int> &tree, vector<int> &a, int node, int start, int end) {  
    if (start == end) {  
        tree[node] = a[start];  
    } else {  
        init(tree, a, node*2, start, (start+end)/2);  
        init(tree, a, node*2+1, (start+end)/2+1, end);  
        tree[node] = min(tree[node*2], tree[node*2+1]);  
    }  
}
```

노드 저장
↓



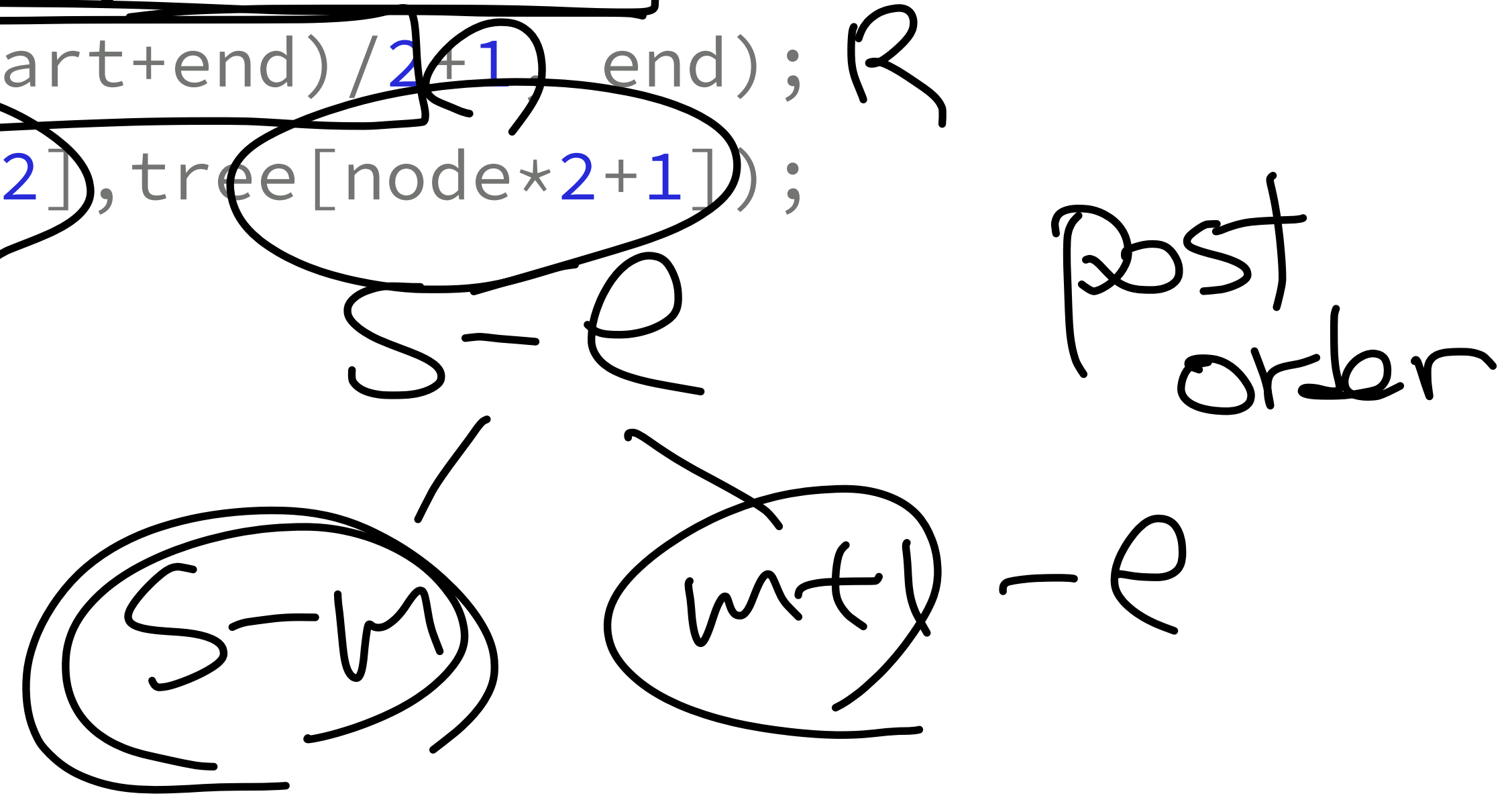
구간
↓



2개의 노드

↑
저장할 수 있는 위치!

node
2*node 2*node+1



세그먼트 트리

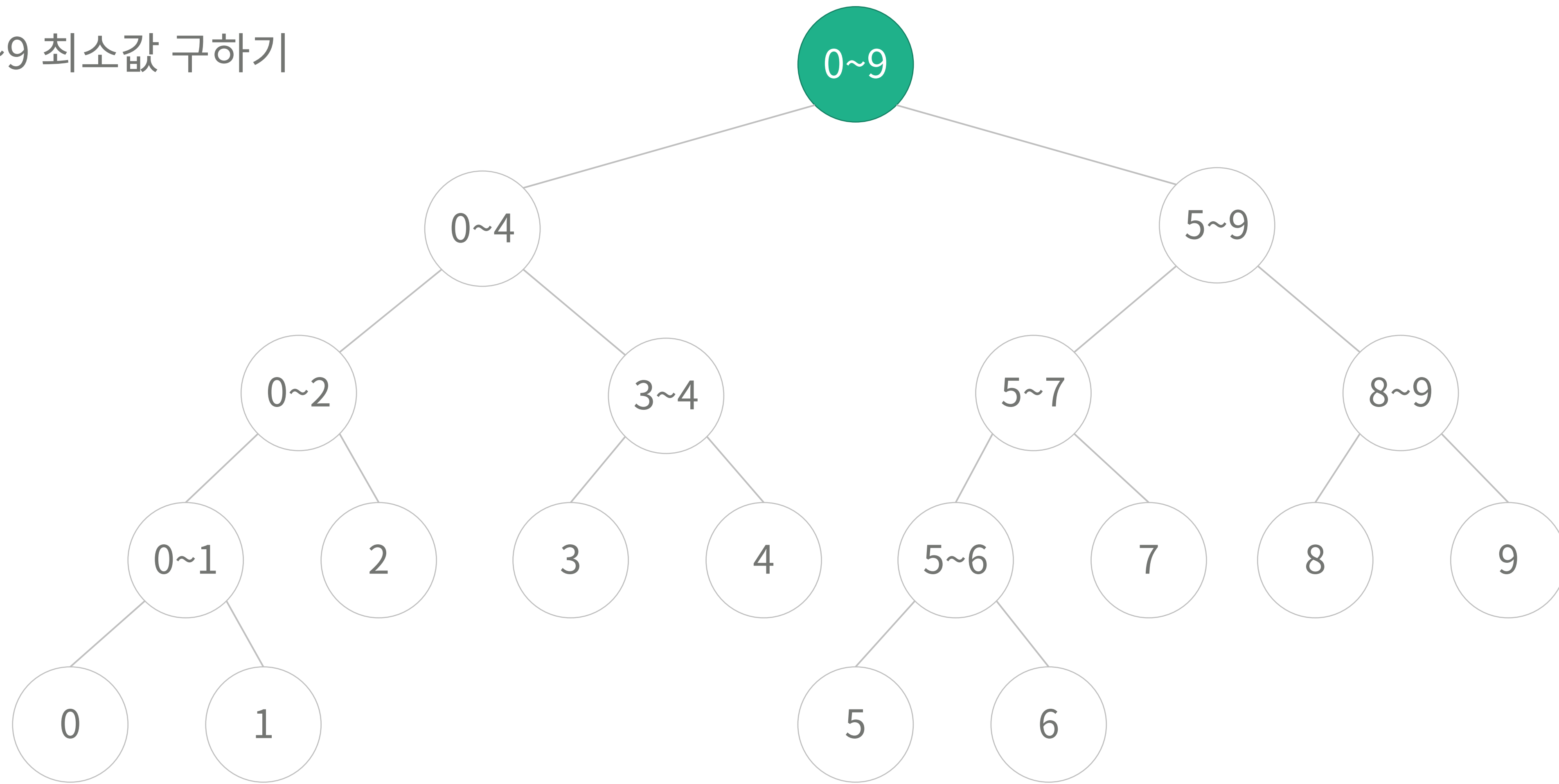
구간의 최소값 구하기 (RMQ)

- 트리 만드는 시간: $O(N \lg N)$

세그먼트 트리

구간의 최소값 구하기 (RMQ)

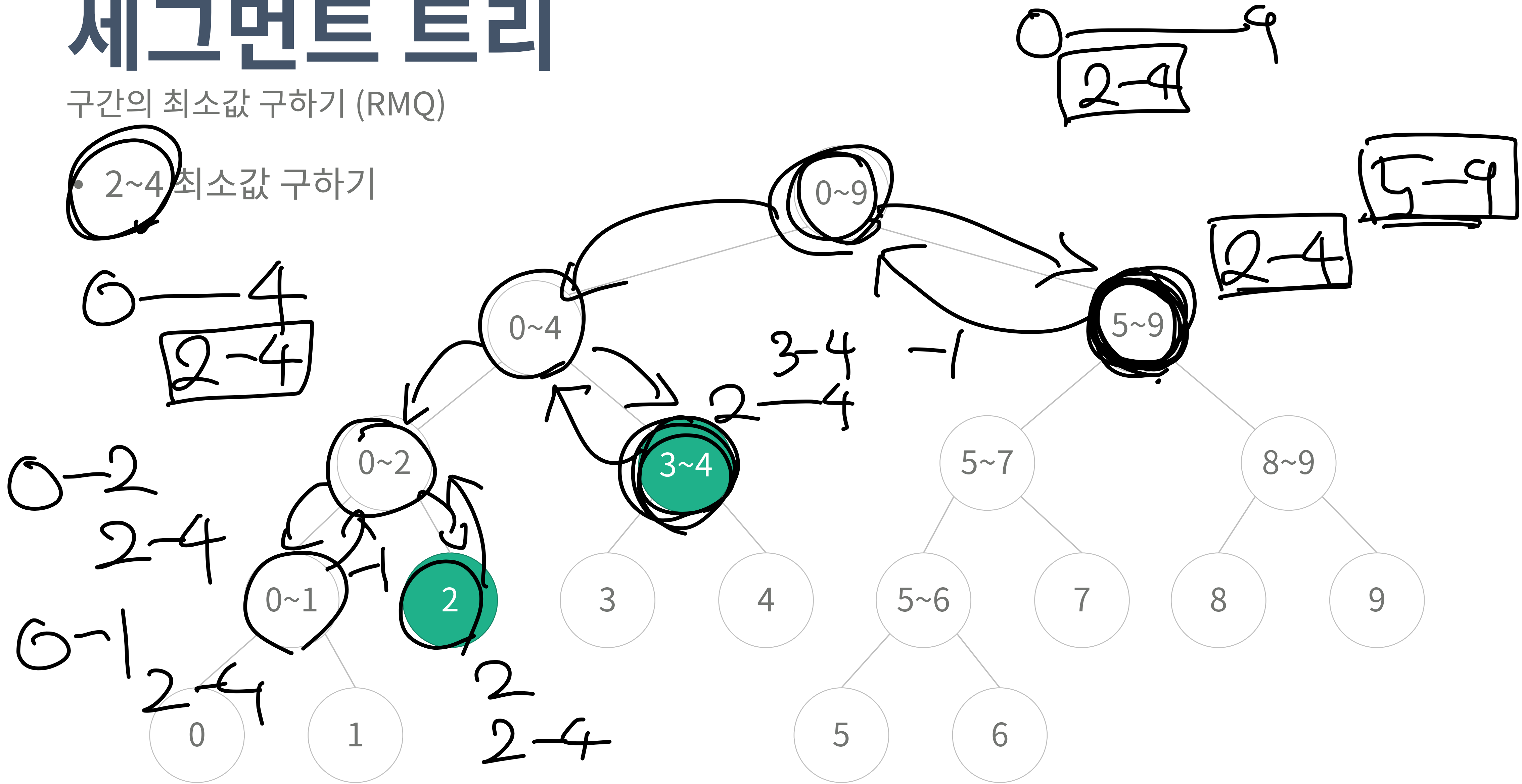
- 0~9 최소값 구하기



세그먼트 트리

구간의 최소값 구하기 (RMQ)

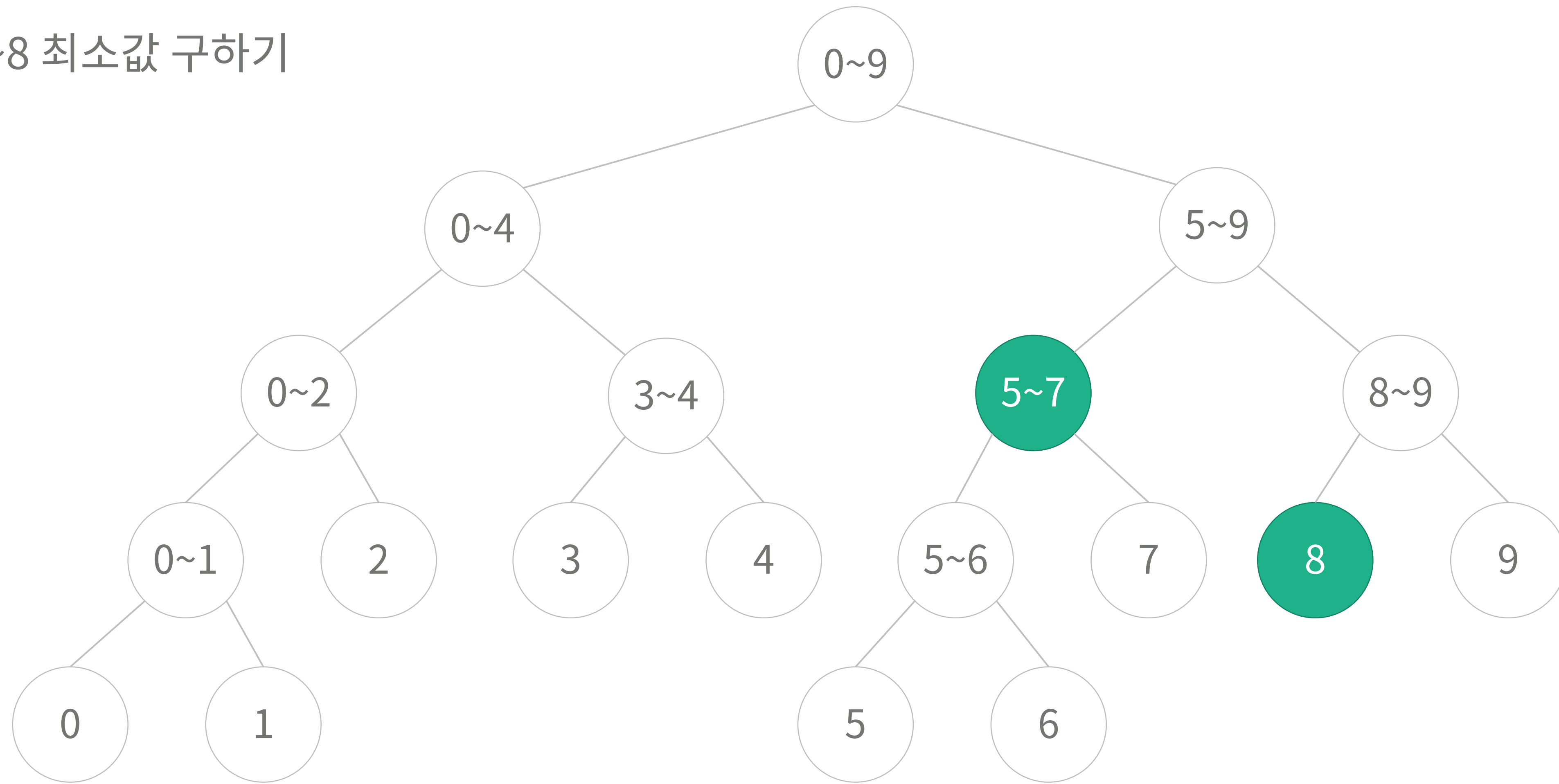
• 2~4 최소값 구하기



세그먼트 트리

구간의 최소값 구하기 (RMQ)

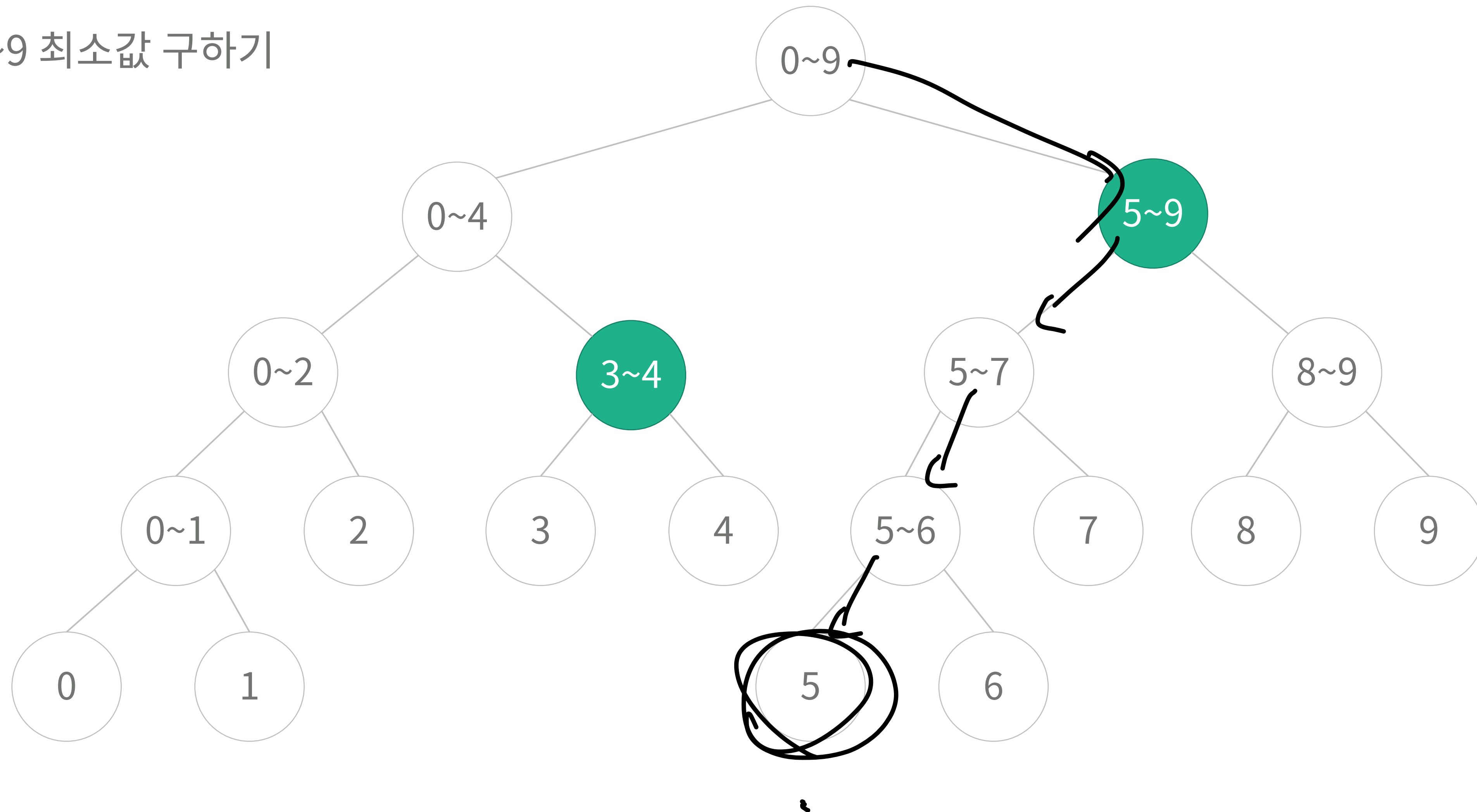
- 5~8 최소값 구하기



세그먼트 트리

구간의 최소값 구하기 (RMQ)

- 3~9 최소값 구하기



세그먼트 트리

구간의 최소값 구하기 (RMQ)

40

노드 정보

```
int query(vector<int> &tree, int node, int start, int end, int i, int j) {
```

```
    if (i > end || j < start) return -1;
```

```
    if (i <= start && end <= j) return tree[node];
```

```
    int m1 = query(tree, 2*node, start, (start+end)/2, i, j);
```

```
    int m2 = query(tree, 2*node+1, (start+end)/2+1, end, i, j);
```

```
    if (m1 == -1) {
```

```
        return m2;
```

```
    } else if (m2 == -1) {
```

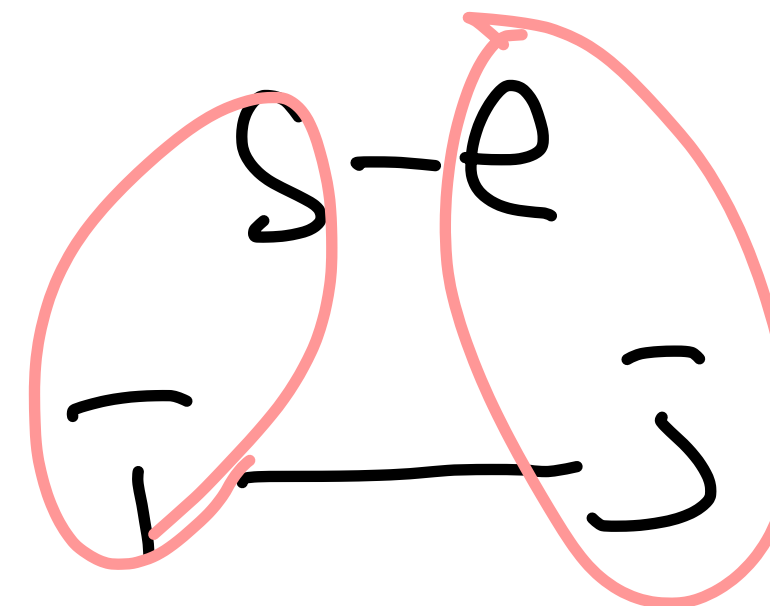
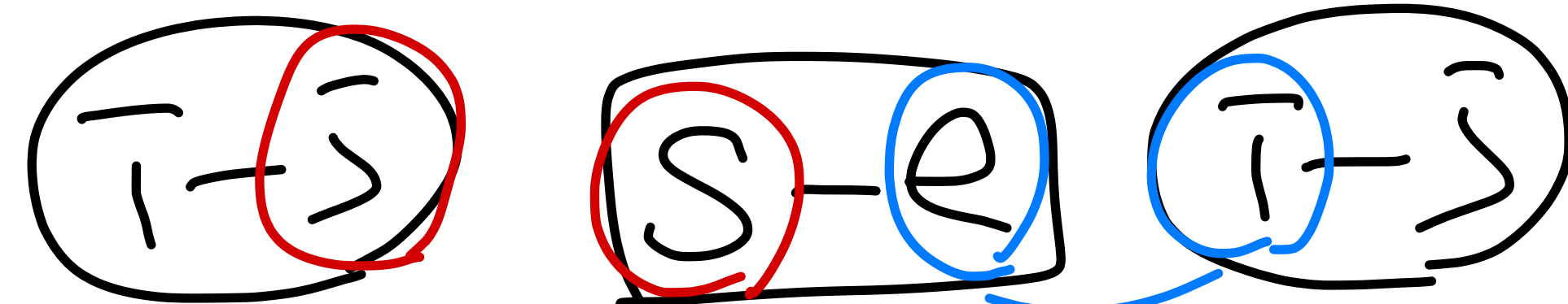
```
        return m1;
```

```
    } else {
```

```
        return min(m1, m2);
```

```
    }
```

```
}
```



세그먼트 트리

구간의 최소값 구하기 (RMQ)

- 쿼리 시간: $O(\lg N)$

세그먼트 트리

구간의 최소값 구하기 (RMQ)

$N \lg N$

- 트리의 루트부터 탐색을 시작하고
- 어떤 노드에서 왼쪽과 오른쪽 자식을 모두 호출하는 경우는 각 레벨마다 최대 2번씩이다.
- 따라서, $O(2 \lg N) = O(\lg N)$ 이다.

총 $O(\lg N)$

최솟값

43

<https://www.acmicpc.net/problem/10868>

- 세그먼트 트리를 이용해서 RMQ를 구현하는 문제

최솟값

<https://www.acmicpc.net/problem/10868>

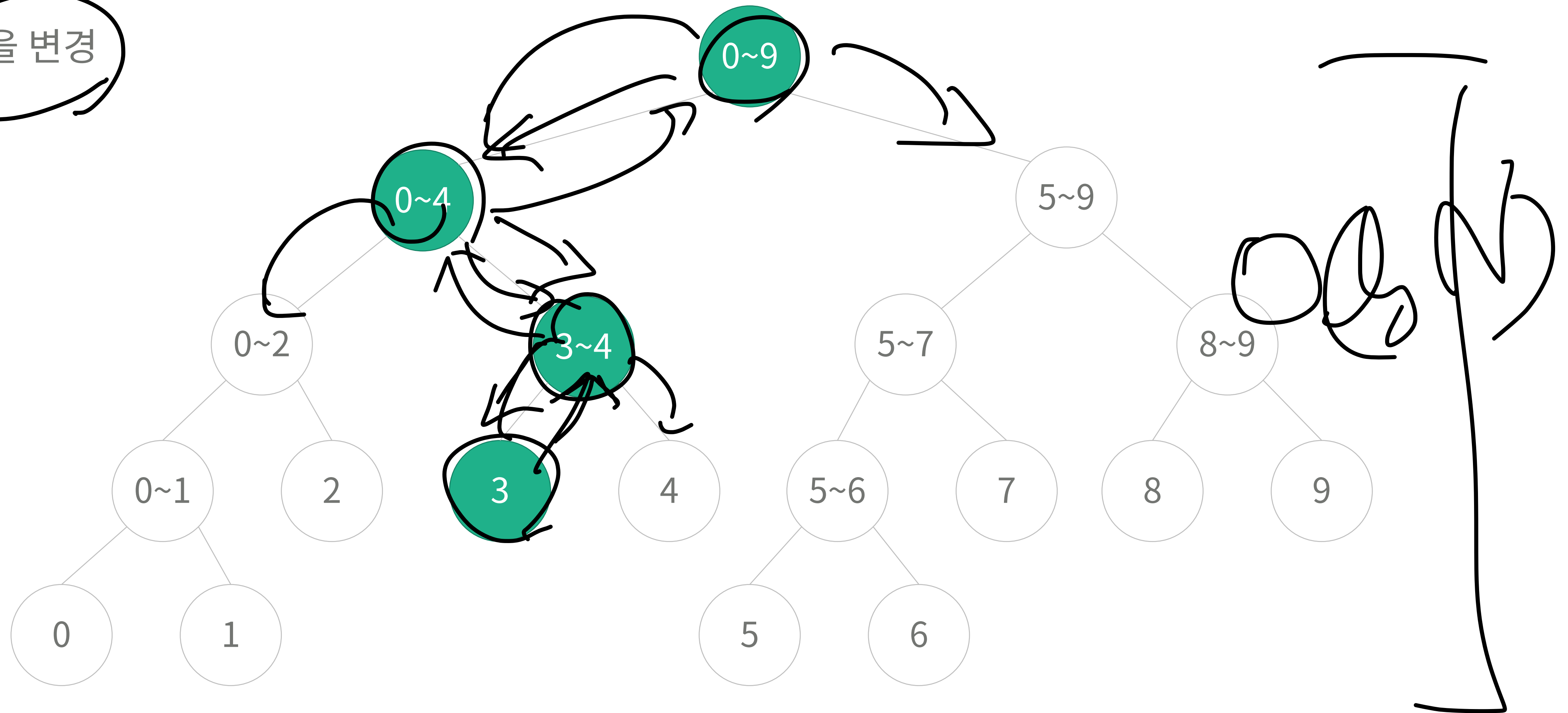
- 루트 N: <http://codeplus.codes/ea6b855c44864cf79d1be7112fad2c6f>
- DP: <http://codeplus.codes/ba4cbd4c3e7e4023b4832b0f22ee05dd>
- 세그먼트 트리: <http://codeplus.codes/cb627400fe2a4df2aa7456ee8696d793>

세그먼트 트리 변경

45

구간의 최소값 구하기 (RMQ)

3을 변경

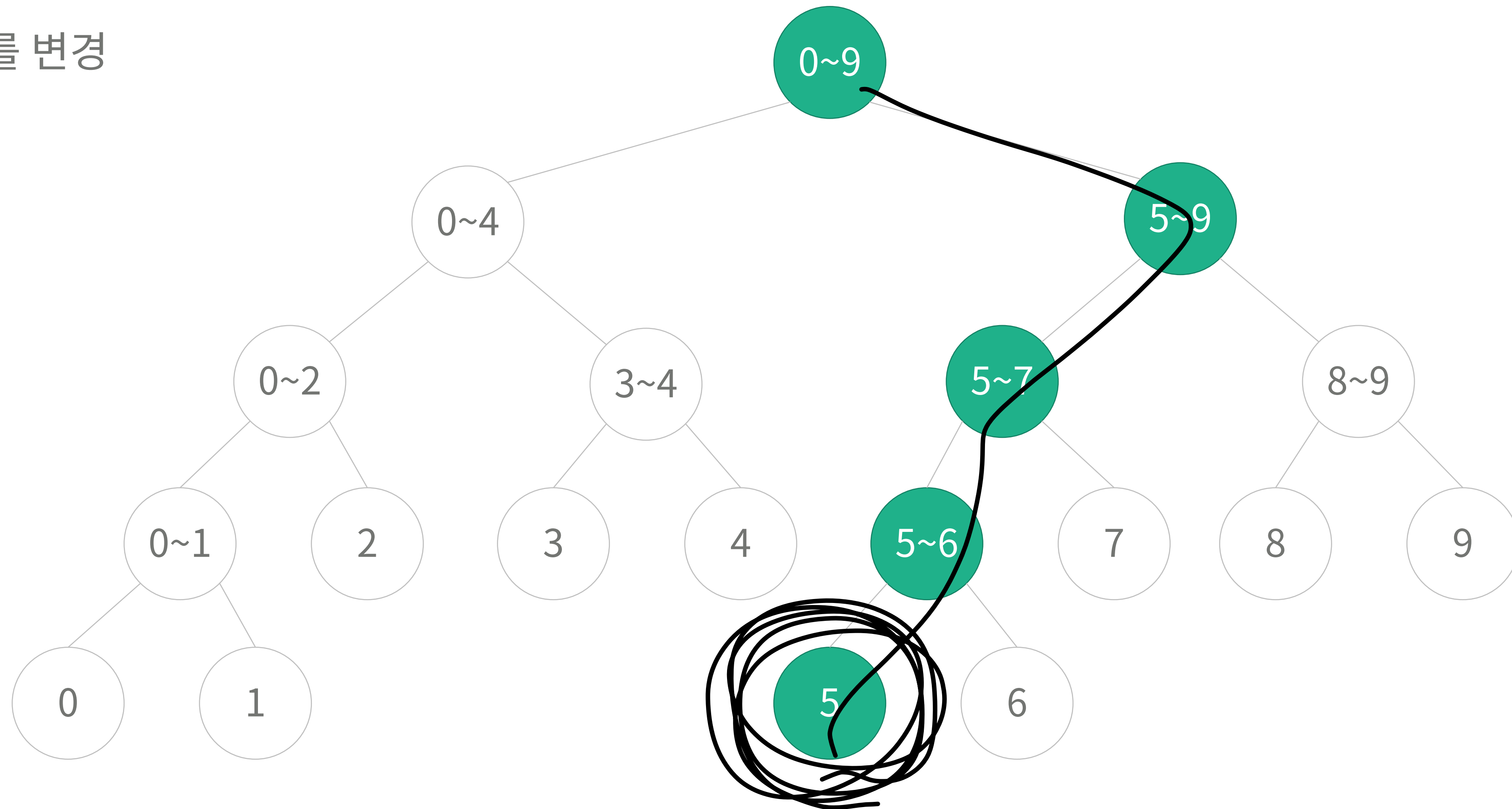


세그먼트 트리 변경

46

구간의 최소값 구하기 (RMQ)

- 5를 변경



세그먼트 트리 변경

구간의 최소값 구하기 (RMQ)

```
void update(vector<int> &tree, int node, int start, int end, int
index, int value) {
```

```
    if (index < start || end < index) return;
```

```
    if (start == end) {
```

```
        tree[node] = value;
```

```
        return;
```

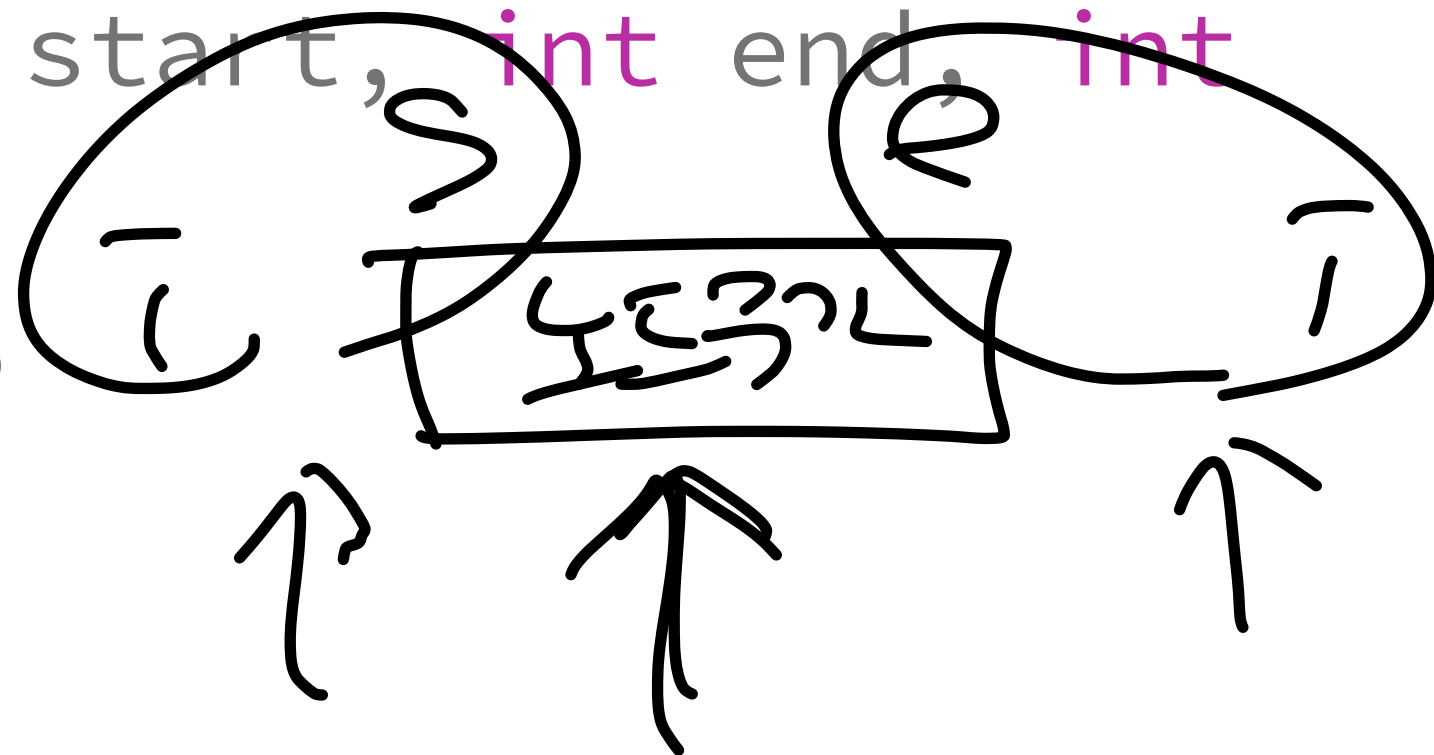
```
    }
```

```
    update(tree, node*2, start, (start+end)/2, index, value);
```

```
    update(tree, node*2+1, (start+end)/2+1, end, index, value);
```

```
    tree[node] = min(tree[node*2], tree[node*2+1]);
```

```
}
```



252

call

수열과 쿼리 17

<https://www.acmicpc.net/problem/14438>

- RMQ 문제 + 수정

수열과 쿼리 17

<https://www.acmicpc.net/problem/14438>

- 소스: <http://codeplus.codes/26ce40834f6c4fc48d7d7d520f9f2eac>

합 구하기

구간 합 구하기

<https://www.acmicpc.net/problem/2042>

- 구간의 최소값이 아니고 합을 구하는 경우에는

• min 대신 +를 하면 된다

구간 합 구하기

52

<https://www.acmicpc.net/problem/2042>

- 소스: <http://codeplus.codes/3b643289518e48d89f377dc077a4c411>