

Fenwick Tree

Fenwick Tree

- i 의 마지막 자리 : i 를 2진수로 나타내었을 때, 가장 마지막 1이 나타내는 값

$$3 = 11_2 \rightarrow 1$$

$$10 = 1010_2 \rightarrow 2$$

$$12 = 1100_2 \rightarrow 4$$

$$- \sim N + 1 == -N$$

$$\text{ex) } N = 01001\textcolor{red}{1}00$$

$$\sim N = 10110011$$

$$\sim N + 1 = 10110100 : N \text{의 보수} + 1 \rightarrow N \text{의 음수}$$

- $N \& -N$: 마지막 비트 참조

$$\text{ex) } N = 01001100$$

$$\& -N = 10110100$$

$$N \& (-N) = 00000\textcolor{red}{1}00$$

Fenwick Tree

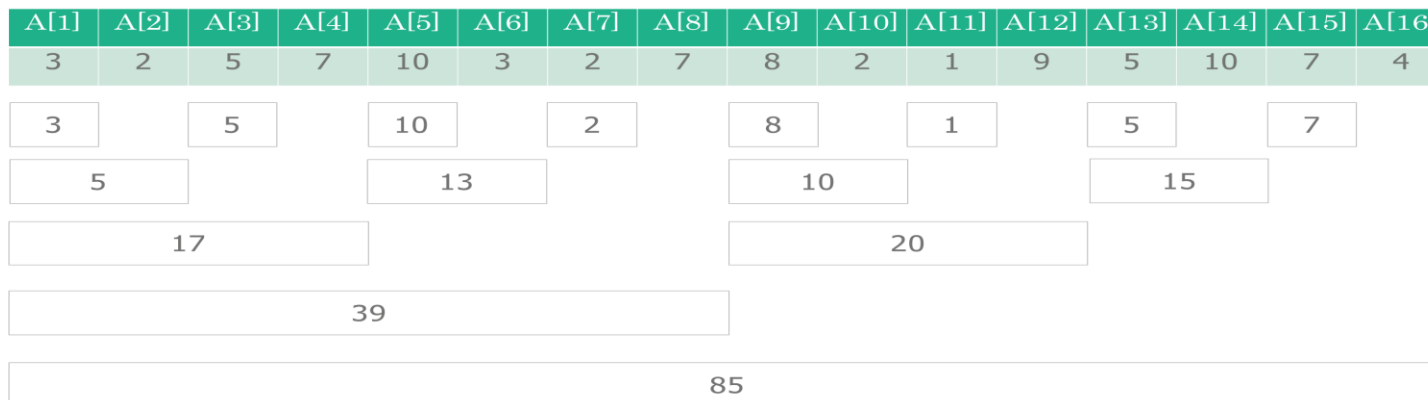
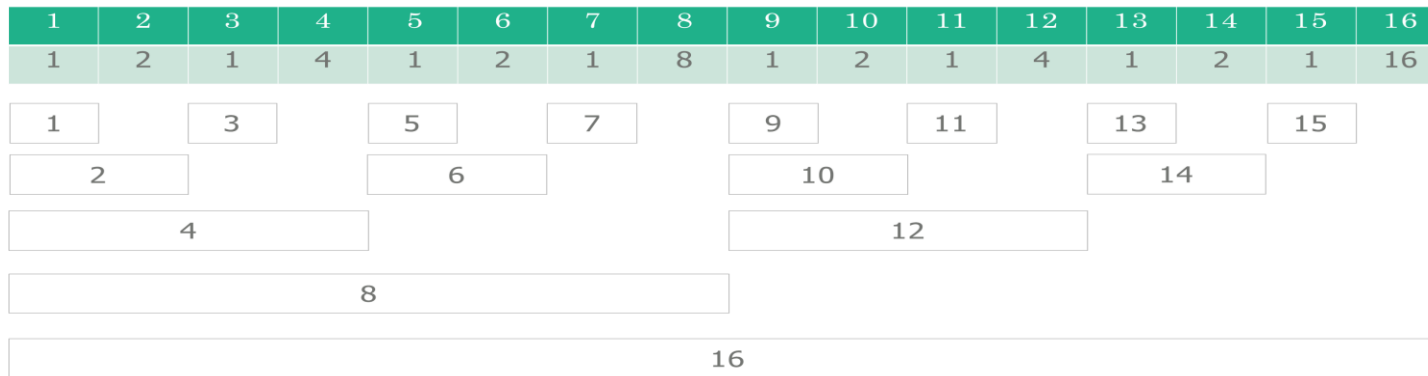
Fenwick Tree

- $tree[i]$: i 의 가장 마지막 비트 만큼 i 부터 그 앞까지 합을 저장한다.

$13 = 1101_2$: $tree[13] = A[13] \sim A[13]$ 까지의 합

$12 = 1100_2$: $tree[12] = A[9] \sim A[12]$ 까지의 합

$8 = 1000_2$: $tree[8] = A[1] \sim A[8]$ 까지의 합

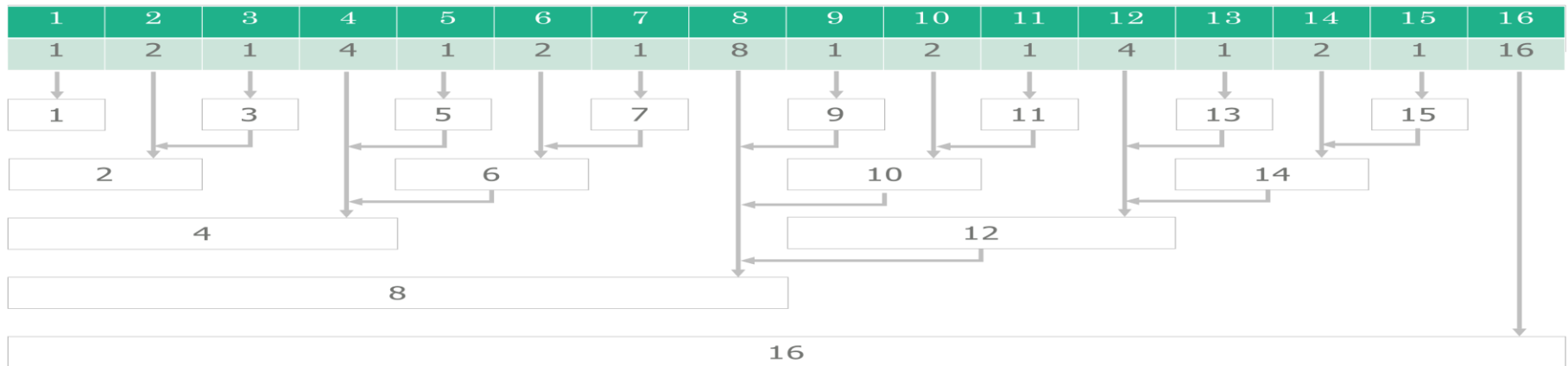
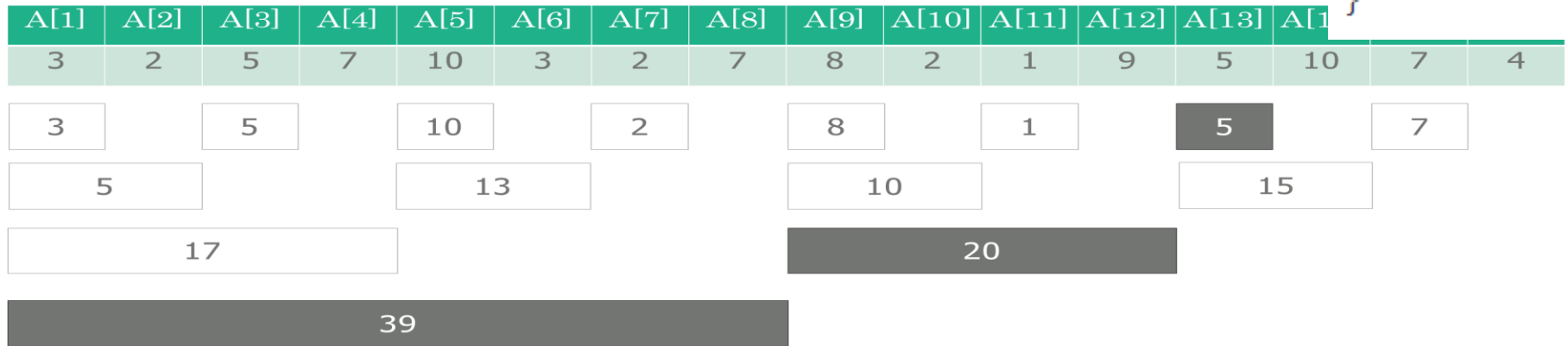


Fenwick Tree

Fenwick Tree

- $A[1] + \dots + A[13]$ 을 구하려면 $13 = 1101_2$
 $\Rightarrow \text{tree}[1101_2] + \text{tree}[1100_2] + \text{tree}[1000_2]$: 가장 마지막 1이 0이 되어

```
int range_sum(int
int ans =
for (int i
ar
}
return ans
```



Fenwick Tree

Fenwick Tree

```
int range_sum(int x) {  
    int ans = 0;  
    for (int i = x; i > 0; i -= i & -i) {  
        ans += tree[i];  
    }  
    return ans;  
}
```



Fenwick Tree

Fenwick Tree

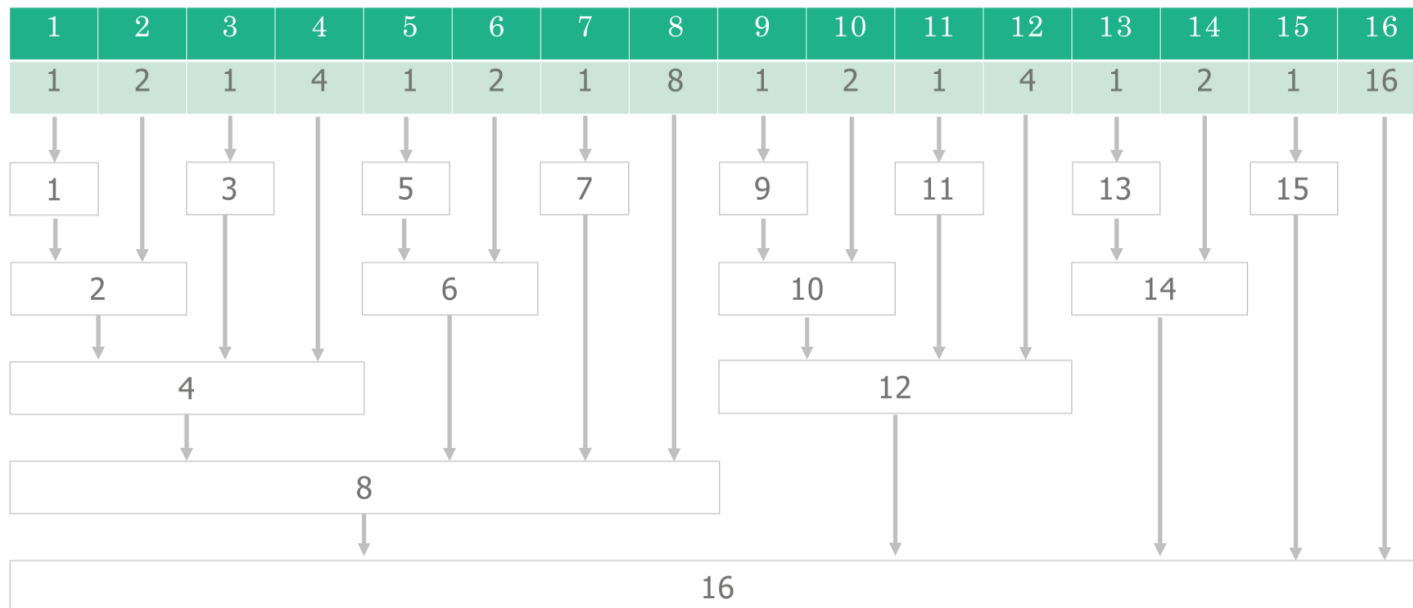
- update를 하려면 변경 분 만큼 변경되는 원소와 변경되는 원소의 마지막 비트를 변경되는 원소에 더한 값에 계속 더해주면 된다.

ex) 3번째 원소가 변경이 되면 변경되어야 하는 대상은 $3 \rightarrow 4 \rightarrow 8 \rightarrow 16$ 이 된다.

$3 = 11_2$ 의 마지막 비트 1_2 를 3에 더해주면 $4 = 100_2$ 이 된다.

$4 = 100_2$ 의 마지막 비트 100_2 을 4에 더해주면 $8 = 1000_2$ 이 된다.

$8 = 1000_2$ 의 마지막 비트 1000_2 을 8에 더해주면 $16 = 10000_2$ 이 되고 최대 인덱스와 같으므로 종료한다.



Fenwick Tree

Fenwick Tree

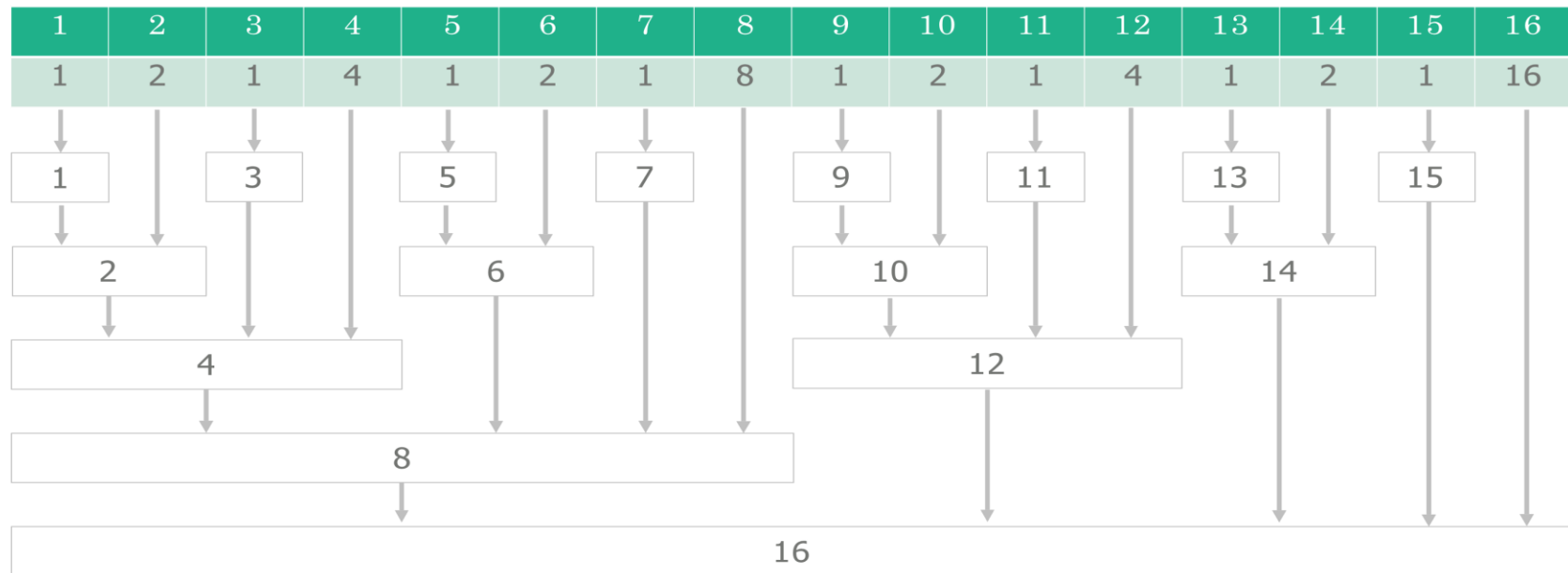
- update를 하려면 변경 분 만큼 변경되는 원소와 변경되는 원소의 마지막 비트를 변경되는 원소에 더한 값에 계속 더해주면 된다.

ex) 3번째 원소가 변경이 되면 변경되어야 하는 대상은 $3 \rightarrow 4 \rightarrow 8 \rightarrow 16$ 이 된다.

$3 = 11_2$ 의 마지막 비트 1_2 를 3에 더해주면 $4 = 100_2$ 이 된다.

$4 = 100_2$ 의 마지막 비트 100_2 을 4에 더해주면 $8 = 1000_2$ 이 된다.

$8 = 1000_2$ 의 마지막 비트 1000_2 을 8에 더해주면 $16 = 10000_2$ 이 되고 최대 인덱스와 같으므로 종료한다.



Fenwick Tree

Fenwick Tree

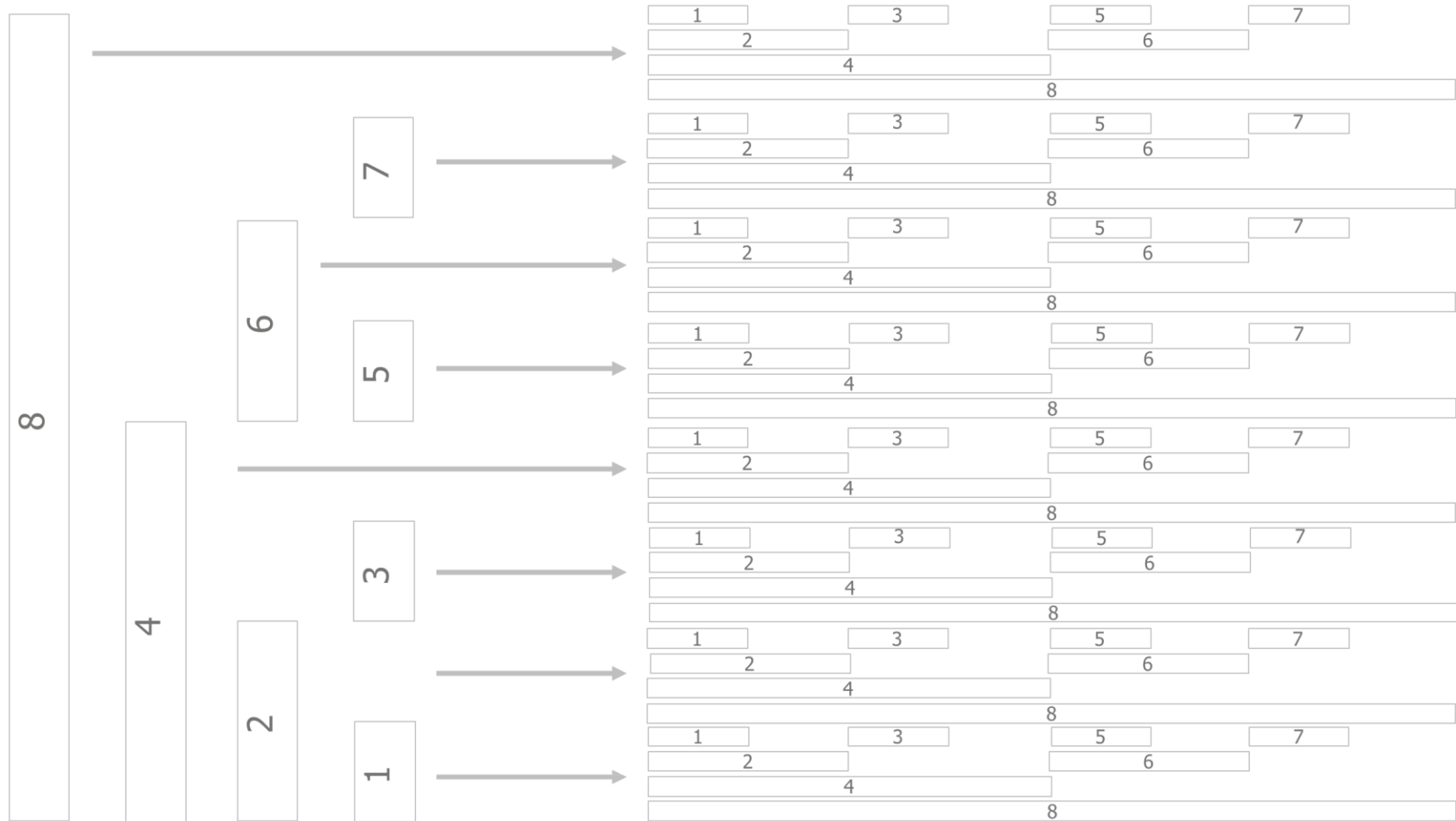
```
int update(int x, int diff) {  
    for (int i = x; i <= N; i += i & -i) {  
        tree[i] += diff;  
    }  
}
```



2D Fenwick Tree

Fenwick Tree

- 1차원을 2차원으로 확장하여 해결한다.



2D Fenwick Tree

Fenwick Tree

- 1차원을 2차원으로 확장하여 해결한다.

```
int range_sum(int y, int x) {  
    int ans = 0;  
    for (int i = y; i > 0; i -= i & -i) {  
        for (int j = x; j > 0; j -= j & -j) {  
            ans += tree[i][j];  
        }  
    }  
    return ans;  
}
```

```
void update(int y, int x, int diff) {  
    for (int i = y; i <= N; i += i & -i) {  
        for (int j = x; j <= N; j += j & -j) {  
            tree[i][j] += diff;  
        }  
    }  
}
```

