2021

16주차 문제풀이
Sangmyung ICPC Team

구간 합 구하기 :

LEE JIWOO

나는 세그먼트 트리가 뭔지 모른다...

Step1 – 공부

- 트리를 초기화하는 기능 ---------- init(start, end, node)
- 값을 변경(갱신)하는 기능 ---------- update(start, end, node, idx, dif)
- 구간 합을 구하는 기능 ---------- sumOfInterval(start, end, node, left, right)

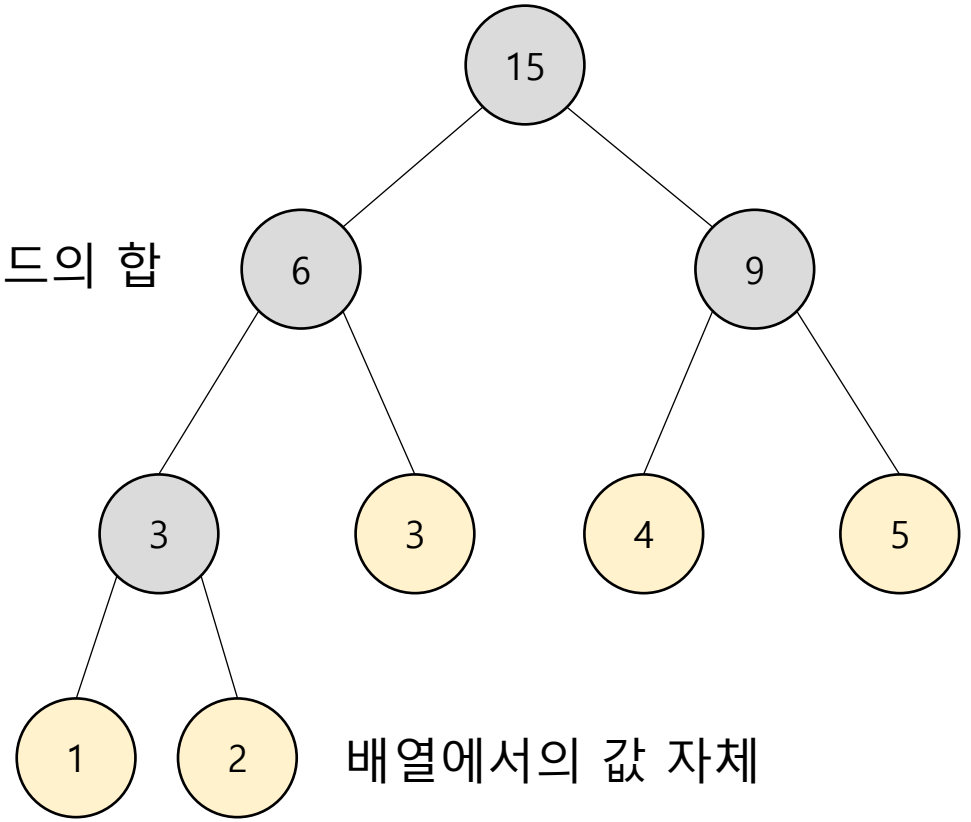
Step1 – 공부

세그먼트 트리에서의 **리프 노드**와 **리프 노드가 아닌 노드**

arr

1	2	3	4	5

>>



Step2 – 구현: init()

```python
def init(s, e, node):
    if s == e: # 리프 노드인 경우
        tree[node] = arr[s] # 실제 배열의 값이 해당 노드의 값
        return tree[node]
    # 리프 노드가 아닌 경우
    mid = (s+e)//2
    # 양쪽 자식 노드의 합이 해당 노드의 값
    tree[node] = init(s, mid, node*2) + init(mid+1, e, node*2+1)
    return tree[node]
```

Step2 – 구현: update()

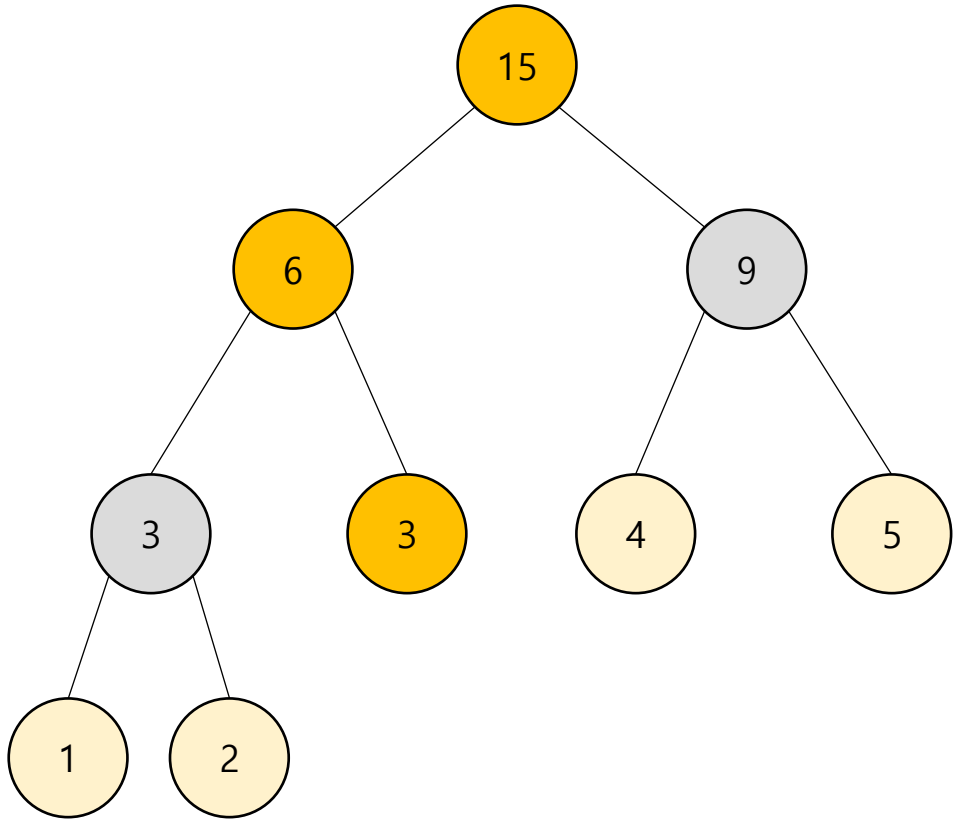
배열 arr의 2번 인덱스 값을 6로 변경

arr[2] = 6

arr

1	2	6	4	5

해당 인덱스를 포함하는 모든 노드의 값을 갱신해야 함
1. 리프 노드인 경우 → 갱신 종료
2. 리프 노드가 아닌 경우 → 자식 노드도 갱신하러 ㄱㄱ



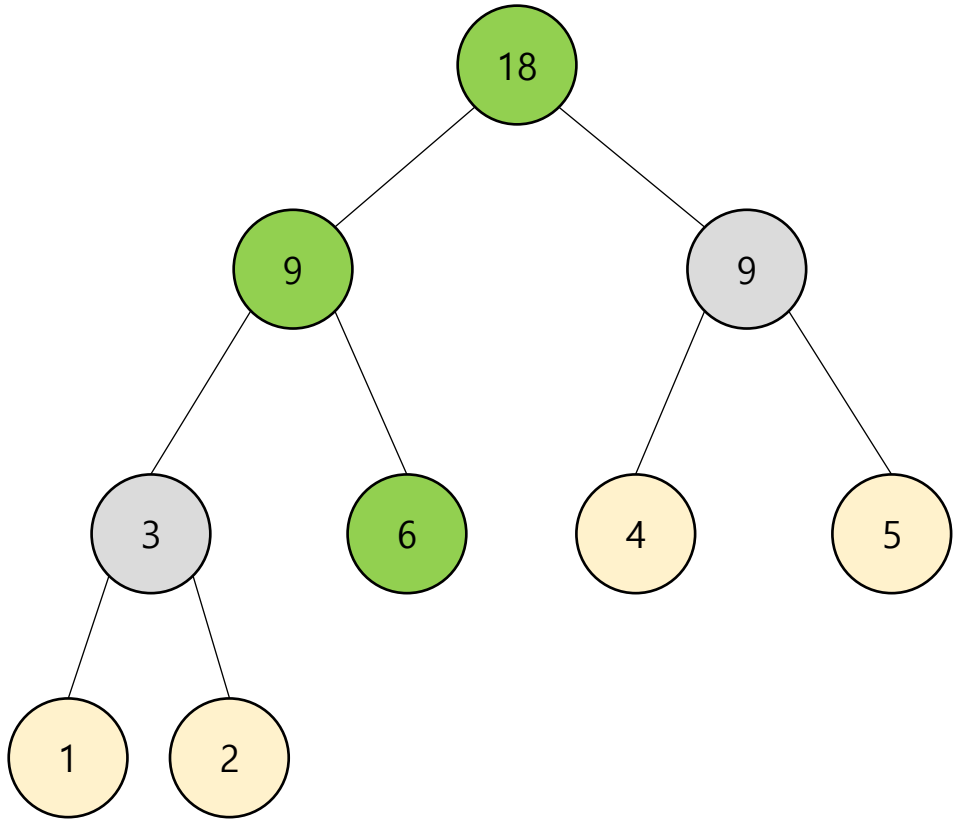
Step2 – 구현: update()

배열 arr의 2번 인덱스 값을 6로 변경

arr[2] = 6

arr

1	2	6	4	5



해당 인덱스를 포함하는 모든 노드의 값을 갱신해야 함
1. 리프 노드인 경우 → 갱신 종료
2. 리프 노드가 아닌 경우 → 자식 노드도 갱신하러 ㄱㄱ

Step2 – 구현: update()

```python
def update(s, e, node, idx, dif):
    if idx < s or idx > e: # idx가 포함되지 않은 범위
        return

    tree[node] += dif
    if s != e: # 리프 노드가 아닌 경우, idx를 포함하는 자식 노드들에서의 값도 변경
        mid = (s+e)//2
        update(s, mid, node*2, idx, dif)
        update(mid+1, e, node*2+1, idx, dif)
```

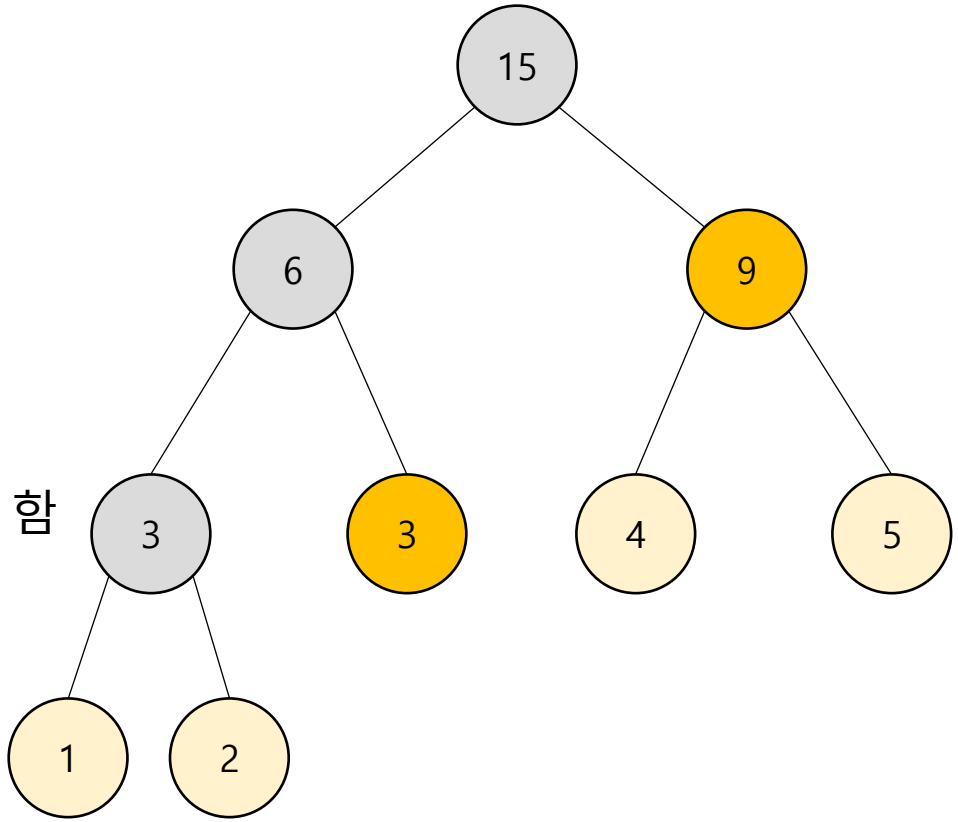
Step2 – 구현: sumOfInterval()

배열 arr의 2~4번 인덱스 구간의 합을 구하기
→ left: 2, right: 4

arr

1	2	3	4	5

각 노드가 나타내는 범위(s, e)와 left, right와의 관계를 살펴야 함



Step2 – 구현: sumOfInterval()

```
def sumOfIntervals(s, e, node, left, right):
    if left > e or right < s: # 범위에서 벗어나는 경우
        return 0
    if left <= s and e <= right: # 구간이 범위를 포함하는 경우
        return tree[node]

    mid = (s+e)//2
    return sumOfIntervals(s, mid, node*2, left, right) + \
           sumOfIntervals(mid+1, e, node*2+1, left, right)
```

Step3 – 입출력

```python
n, m, k = map(int, input().split())
arr = [int(stdin.readline()) for _ in range(n)]
commands = [list(map(int, stdin.readline().split())) for _ in range(m+k)]
tree = [0]*(n*4)

# 트리 초기화
init(0, n-1, 1)

for cmd in commands:
    if cmd[0] == 1: # 값 변경
        dif = cmd[2] - arr[cmd[1]-1] # 기존 값과의 차
        update(1, n, 1, cmd[1], dif) # 트리에서의 값 변경
        arr[cmd[1]-1] = cmd[2] # 실제 배열의 값 변경
    else: # 구간 합 출력
        print(sumOfIntervals(1, n, 1, cmd[1], cmd[2]))
```

Thanks