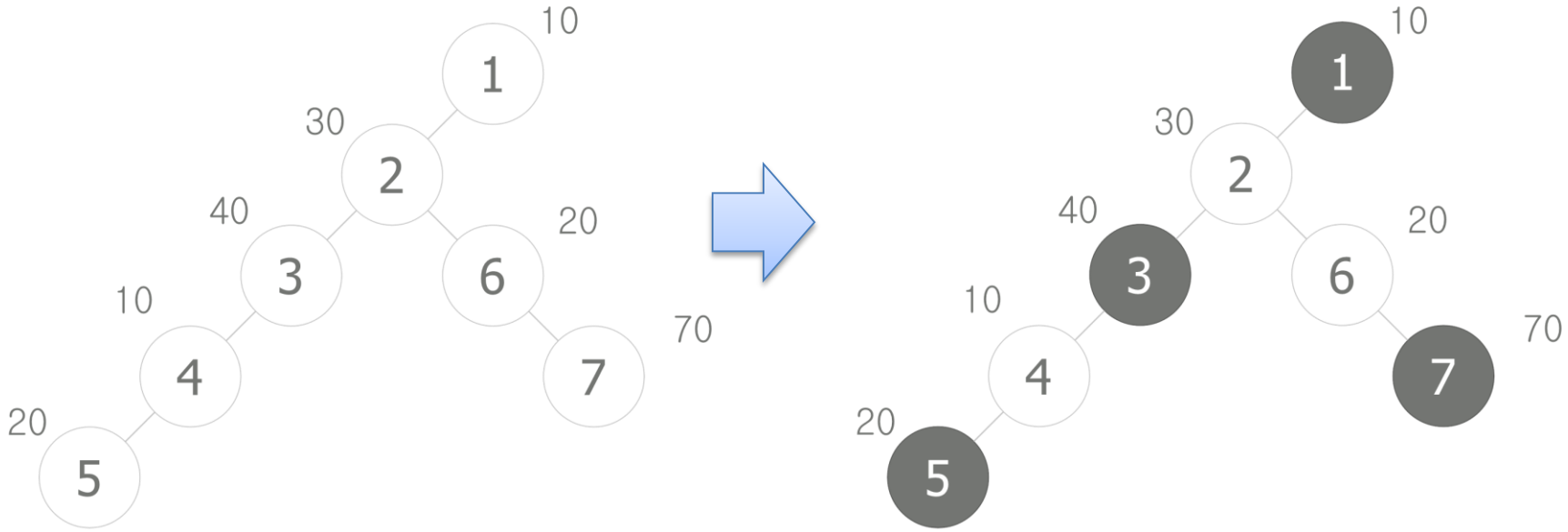
트리의 독립 집합

https://www.acmicpc.net/problem/2213

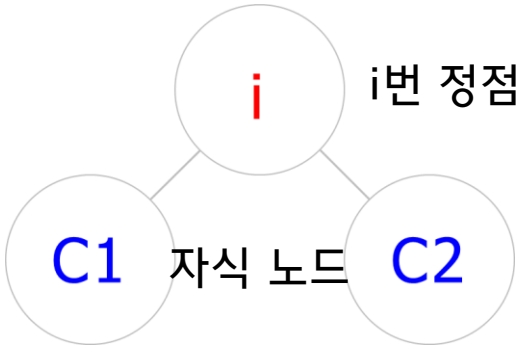


트리의 독립 집합

https://www.acmicpc.net/problem/2213

- 독립 집합에 포함되어 있는 정점끼리는 인접하지 않아야 한다.
- 포스트 오더로 탐색하면 다이나믹 프로그래밍을 수행한다.
- dp[i][0] = i번 정점을 독립집합에 포함시키지 않았을 때, 최대 크기
  - ☞ i번 정점을 독립집합에 포함시키지 않았기 때문에, 자식을 포함시켜도 되고, 포함시키지 않아도 된다.

  dp[i][1] = i번 정점을 독립집합에 포함시켰을 때, 최대 크기
  - ☞ i번 정점을 포함시켰기 때문에, 자식을 포함시키면 안된다.
- dp[i][0] = $\sum_{k=1} \max(dp[C_k][0],\ dp[C_k][1])$
- dp[i][1] = $\sum_{k=1}(dp[C_k][0])\ + A[i]$



- 독립집합의 최댓값을 구한 뒤, 트리를 순회하면서 어떤 값이 최대값인지 확인한다.

트리의 독립 집합

https://www.acmicpc.net/problem/2213

- C++ 코드

```
void solve(int here, int parent) {
        for (int child : AdjList[here]){
                if (parent == child) continue;
                solve(child, here);
        }

        dp[here][0] = 0;
        dp[here][1] = W[here];

        for (int child : AdjList[here]) {
                if (parent == child) continue;
                dp[here][0] += max(dp[child][0], dp[child][1])
                dp[here][1] += dp[child][0];
        }
}
```

```
void tracking(int here, int choose, int parent) {
        if (choose == 0) {
                for (int child : AdjList[here]) {
                        if (parent == child) continue;
                        if (dp[child][0] < dp[child][1]) {
                                tracking(child, 1, here);
                        }
                        else {
                                tracking(child, 0, here);
                        }
                }
        }
        else if(choose == 1){
                ans.push_back(here);
                for (int child : AdjList[here]) {
                        if (parent == child) continue;
                        tracking(child, 0, here);
                }
        }
}
```