

BFS

최백준 choi@startlink.io

숨바꼭질

2

<https://www.acmicpc.net/problem/1697>

- 수빈이의 위치: N
 - 동생의 위치: K
 - 동생을 찾는 가장 빠른 시간을 구하는 문제
-
- 수빈이가 할 수 있는 행동 (위치: X)
 1. 걷기: $X+1$ 또는 $X-1$ 로 이동 (1초)
 2. 순간이동: $2*X$ 로 이동 (1초)

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 수빈이의 위치: 5
- 동생의 위치: 17
- 5-10-9-18-17 로 4초만에 동생을 찾을 수 있다.

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 큐에 수빈이의 위치를 넣어가면서 이동시킨다
- 한 번 방문한 곳은 다시 방문하지 않는 것이 좋기 때문에, 따로 배열에 체크하면서 방문

숨바꼭질

<https://www.acmicpc.net/problem/1697>

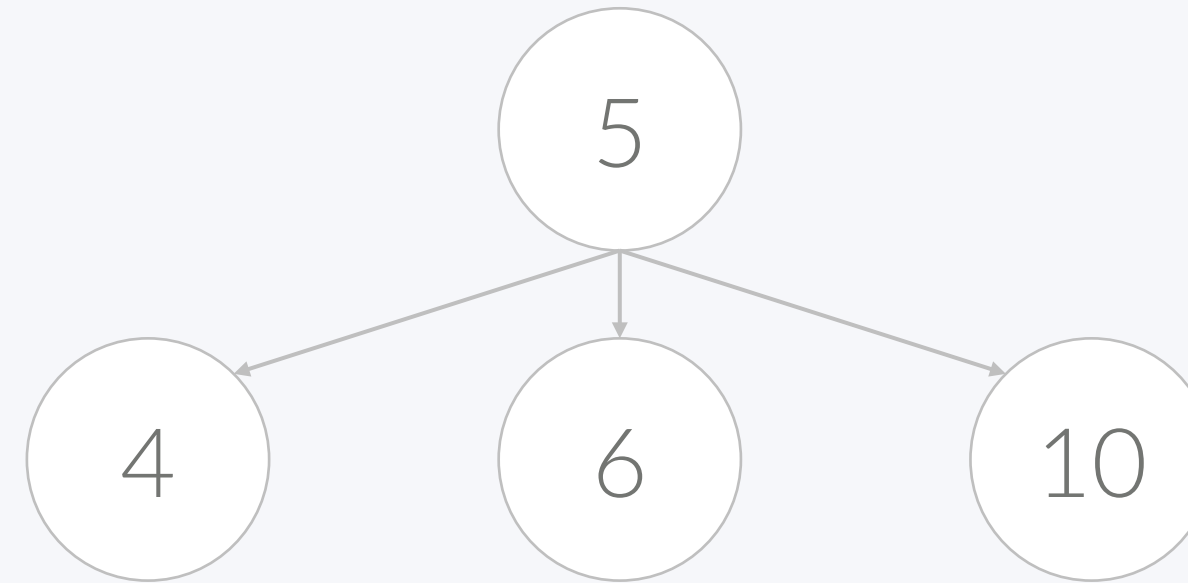
- 가장 처음
- Queue: 5

5

숨바꼭질

<https://www.acmicpc.net/problem/1697>

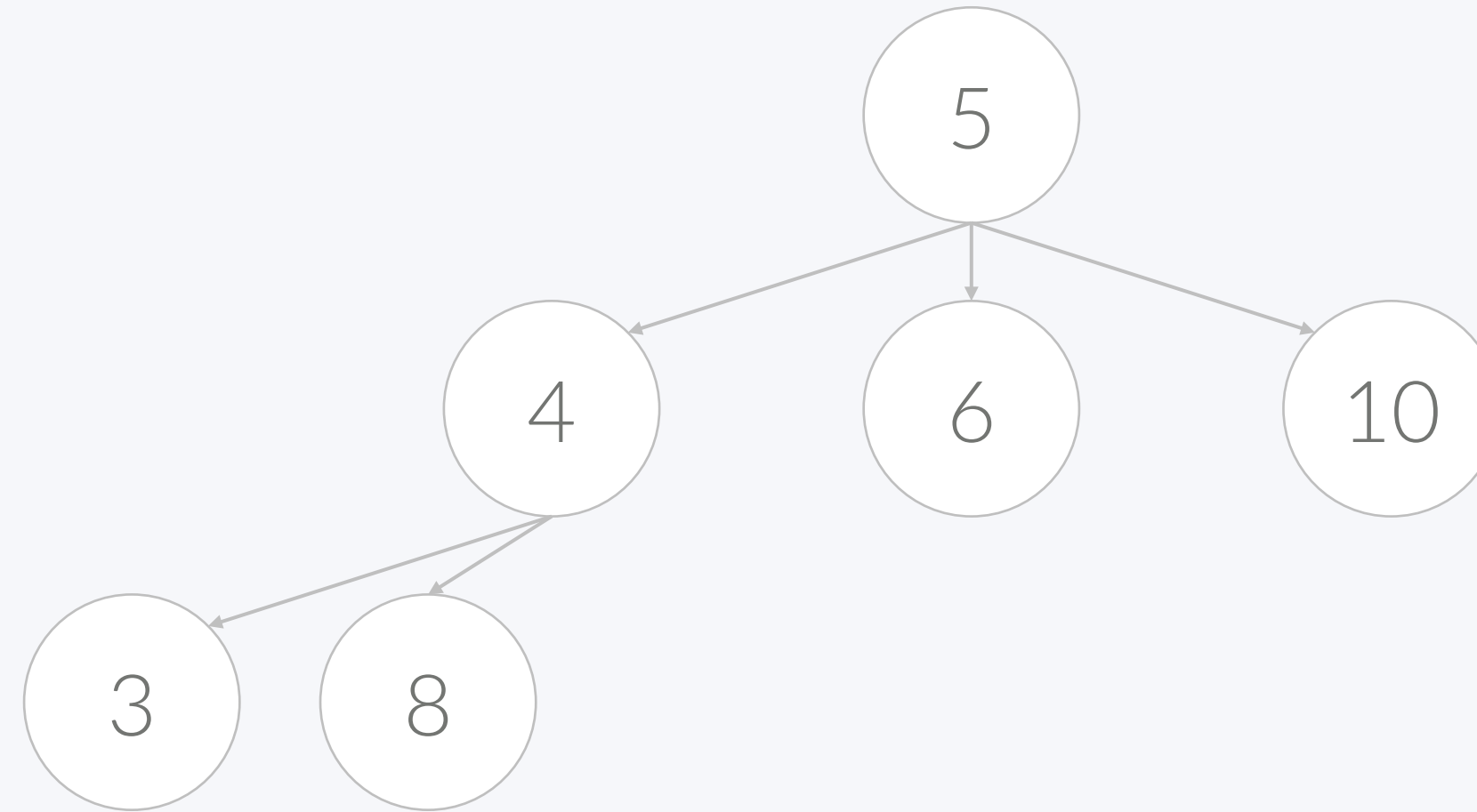
- 5에서 이동
- Queue: 5 4 6 10



숨바꼭질

<https://www.acmicpc.net/problem/1697>

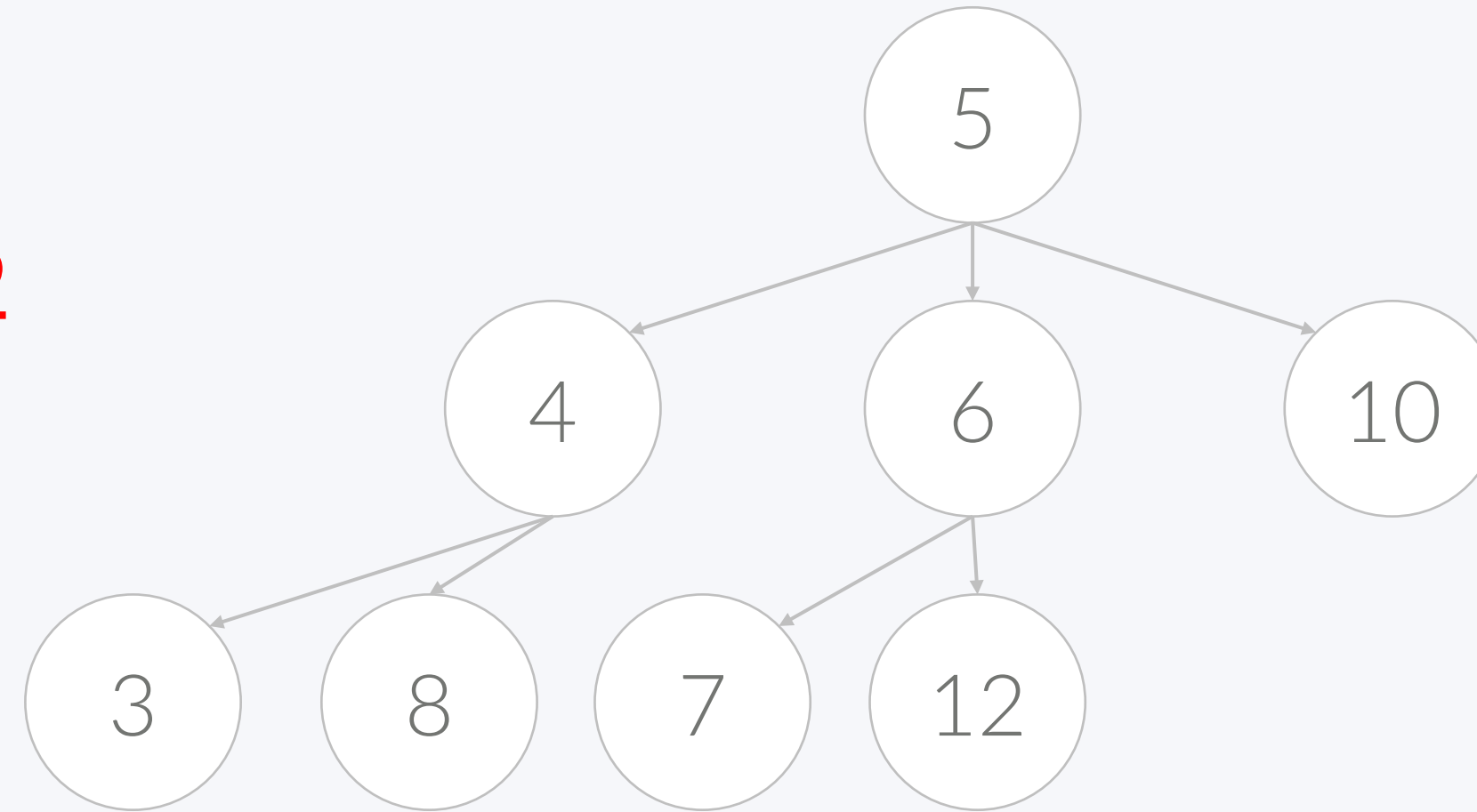
- 4에서 이동
- Queue: 5 4 6 10 3 8



숨바꼭질

<https://www.acmicpc.net/problem/1697>

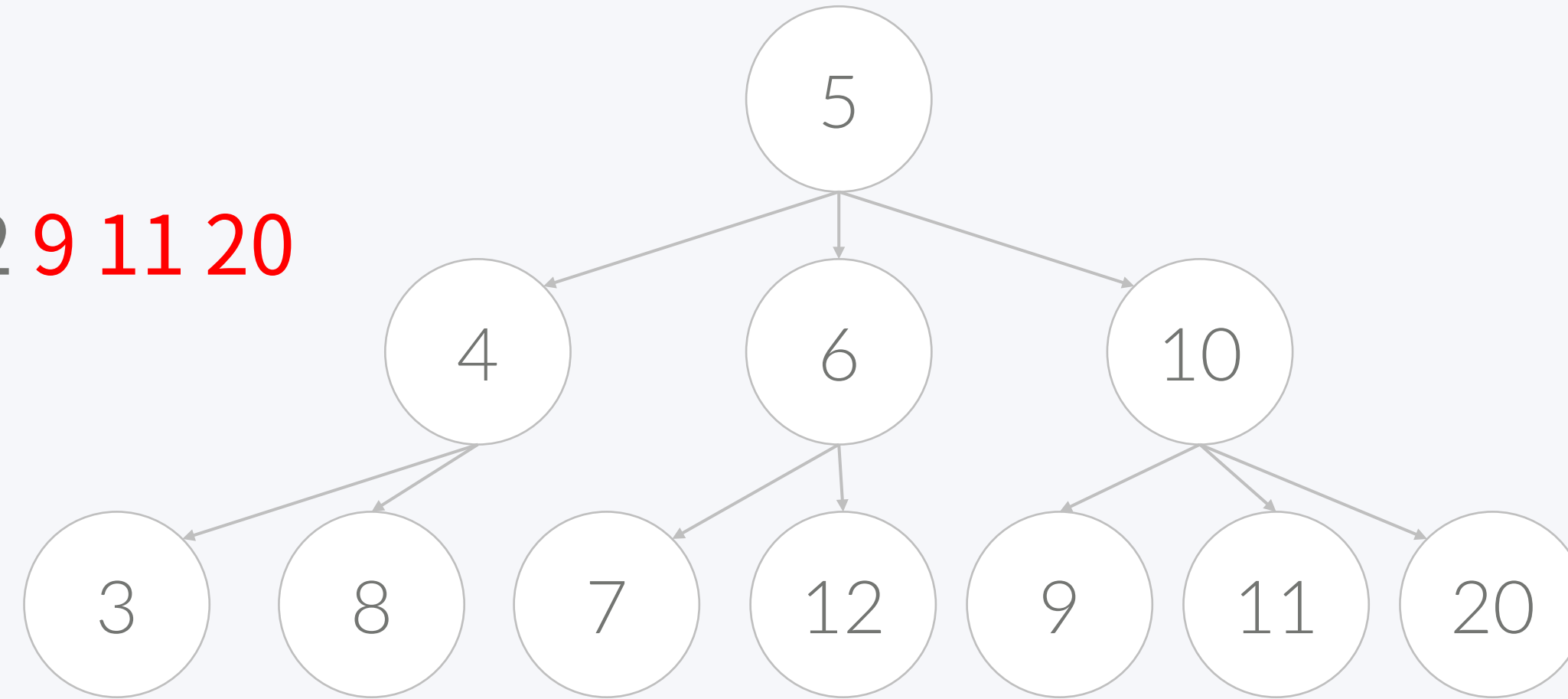
- 6에서 이동
- Queue: 5 4 6 10 3 8 7 12



숨바꼭질

<https://www.acmicpc.net/problem/1697>

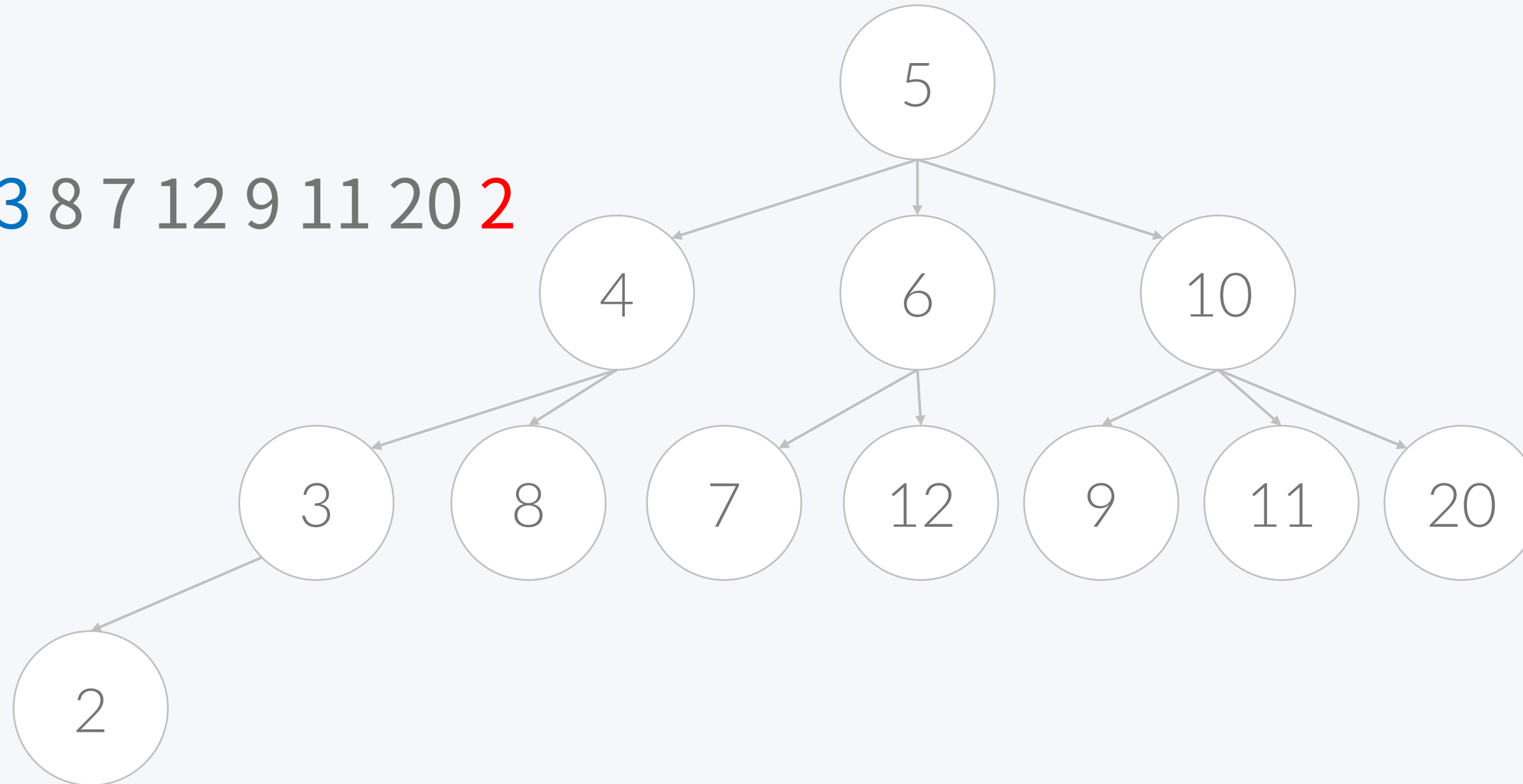
- 10에서 이동
- Queue: 5 4 6 10 3 8 7 12 9 11 20



숨바꼭질

<https://www.acmicpc.net/problem/1697>

- 3에서 이동
- Queue: 5 4 6 10 3 8 7 12 9 11 20 2



숨바꼭질

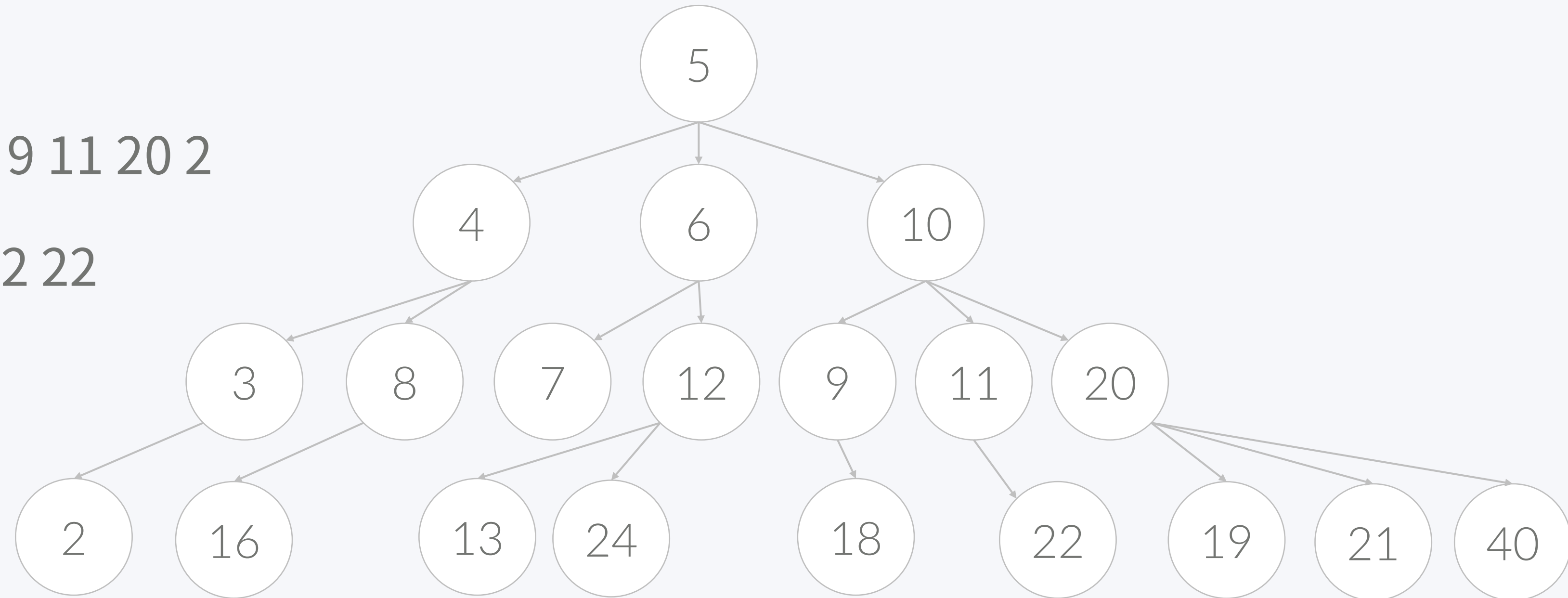
<https://www.acmicpc.net/problem/1697>

- 이런식으로...

숨바꼭질

<https://www.acmicpc.net/problem/1697>

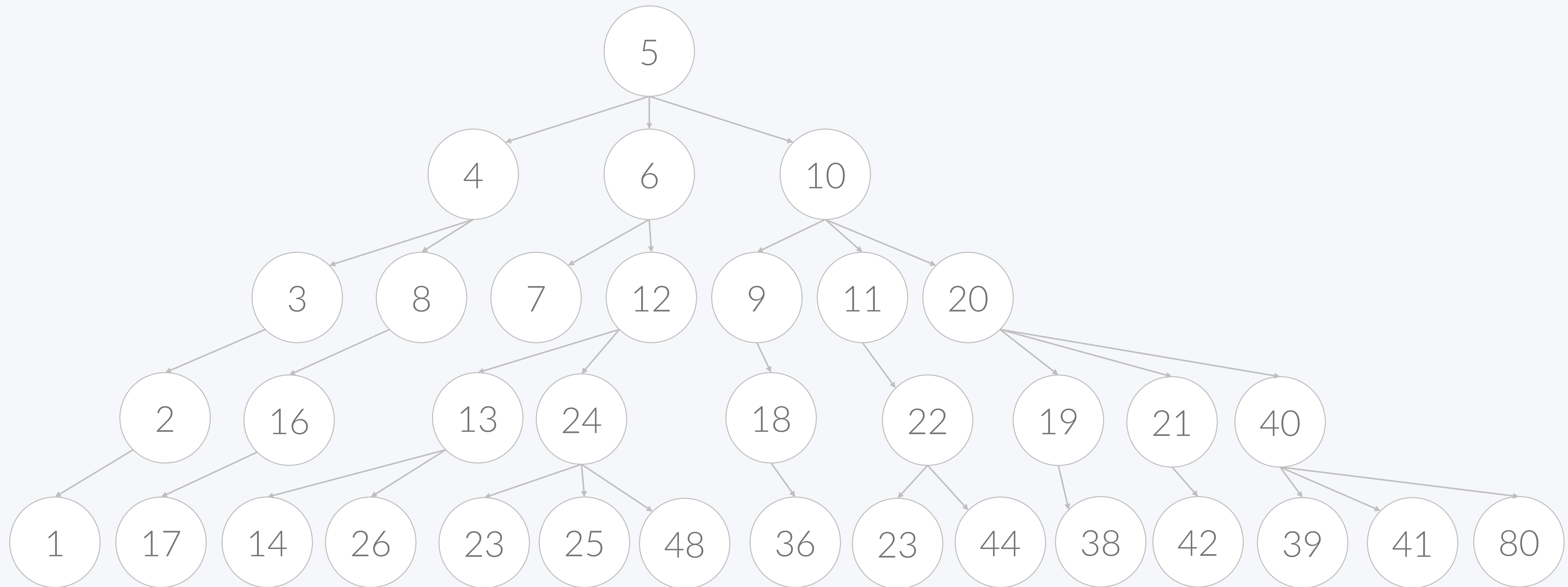
- Queue:
- 5 4 6 10 3 8 7 12 9 11 20 2
- 16 13 15 28 18 12 22
- 19 21 40



숨바꼭질

<https://www.acmicpc.net/problem/1697>

•



숨바꼭질

14

<https://www.acmicpc.net/problem/1697>

- $check[i]$ = i 를 방문했는지
- $dist[i]$ = i 를 몇 번만에 방문했는지

숨바꼭질

15

<https://www.acmicpc.net/problem/1697>

```
check[n] = true;
dist[n] = 0;
queue<int> q;
q.push(n);
while (!q.empty()) {
    int now = q.front();
    q.pop();
    if (now-1 >= 0) {
        if (check[now-1] == false) {
            q.push(now-1);
            check[now-1] = true;
            dist[now-1] = dist[now] + 1;
        }
    }
}
```

```
if (now+1 < MAX) {
    if (check[now+1] == false) {
        q.push(now+1);
        check[now+1] = true;
        dist[now+1] = dist[now] + 1;
    }
}
if (now*2 < MAX) {
    if (check[now*2] == false) {
        q.push(now*2);
        check[now*2] = true;
        dist[now*2] = dist[now] + 1;
    }
}
```

숨바꼭질

<https://www.acmicpc.net/problem/1697>

- C: <https://gist.github.com/Baekjoon/2505b37c55949c9487904f49b5346111>
- C++: <https://gist.github.com/Baekjoon/5b8924d3aec661746358>
- Java: <https://gist.github.com/Baekjoon/e1abcf6deb6796c1282a>

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- 숨바꼭질 문제 + 이동하는 방법을 출력하는 문제

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- now -> next를 갔다고 한다면

```
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now] + 1;  
}
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- now -> next를 갔다고 한다면

```
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    from[next] = now;  
    dist[next] = dist[now] + 1;  
}
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- $\text{from}[i]$ = 어디에서 왔는지
- 의미: $\text{from}[i] \rightarrow i$
- N에서 K를 가는 문제 이기 때문에
- K부터 from을 통해서 N까지 가야한다.
- 즉, 역순으로 저장되기 때문에, 다시 역순으로 구하는 것이 필요하다.

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

```
void print(int n, int m) {  
    if (n != m) {  
        print(n, from[m]);  
    }  
    cout << m << ' ';  
}
```

숨바꼭질 4

22

<https://www.acmicpc.net/problem/13913>

```
stack<int> ans;
for (int i=m; i!=n; i=from[i]) {
    ans.push(i);
}
ans.push(n);
while (!ans.empty()) {
    cout << ans.top() << ' ';
    ans.pop();
}
cout << '\n';
```

숨바꼭질 4

<https://www.acmicpc.net/problem/13913>

- C: <https://gist.github.com/Baekjoon/3d918ef75f8598f4e73abf0fb5e972ef>
- C++: <https://gist.github.com/Baekjoon/914c1f46da5d4e662758ff6c452dc1e8>
- C++: <https://gist.github.com/Baekjoon/d823eaf585990e75ef5760dd80f232a6>
- Java: <https://gist.github.com/Baekjoon/7dc73e85fe343fc9dbea8b32c1347d7a>
- Java: <https://gist.github.com/Baekjoon/fc8c2fa94562827e8289d59a3fa4b96a>

DSLR

<https://www.acmicpc.net/problem/9019>

- 네 자리 숫자 A와 B가 주어졌을 때
- $A \rightarrow B$ 로 바꾸는 최소 연산 횟수
- $D: N \rightarrow 2 \times N$
- $S: N \rightarrow N - 1$
- L: 한 자리씩 왼쪽으로
- R: 한 자리씩 오른쪽으로

DSLR

<https://www.acmicpc.net/problem/9019>

- 앞의 두 문제와 동일하게 풀지만
- 이 문제는 최소값을 구해야 하는건 맞지만
- 어떠한 과정을 거쳐야 하는지를 구해야 한다
- 배열 두 개를 더 이용해서 어떤 과정을 거쳤는지를 저장해야 한다
- $\text{from}[i] = i$ 를 어떤 수에서 만들었는지
- $\text{how}[i] = i$ 를 어떻게 만들었는지

DSL

<https://www.acmicpc.net/problem/9019>

```
int next = (now*2) % 10000;  
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now]+1;  
    from[next] = now;  
    how[next] = 'D';  
}
```

DSLR

<https://www.acmicpc.net/problem/9019>

```
next = now-1;
if (next == -1) next = 9999;
if (check[next] == false) {
    q.push(next);
    check[next] = true;
    dist[next] = dist[now]+1;
    from[next] = now;
    how[next] = 'S';
}
```

DSLR

<https://www.acmicpc.net/problem/9019>

```
next = (now%1000)*10 + now/1000;
if (check[next] == false) {
    q.push(next);
    check[next] = true;
    dist[next] = dist[now]+1;
    from[next] = now;
    how[next] = 'L';
}
```

DSL R

<https://www.acmicpc.net/problem/9019>

```
next = (now/10) + (now%10)*1000;  
if (check[next] == false) {  
    q.push(next);  
    check[next] = true;  
    dist[next] = dist[now]+1;  
    from[next] = now;  
    how[next] = 'R';  
}
```

DSLR

<https://www.acmicpc.net/problem/9019>

```
string ans = "";  
while (B != A) {  
    ans += how[B];  
    B = from[B];  
}  
reverse(ans.begin(), ans.end());  
cout << ans << '\\n';
```

DSL R

<https://www.acmicpc.net/problem/9019>

```
void print(int A, int B) {  
    if (A == B) return;  
    print(A, from[B]);  
    cout << how[B];  
}
```

DSL R

<https://www.acmicpc.net/problem/9019>

- C++: <https://gist.github.com/Baekjoon/05b6e7a9c0bf6d4742ab>
- C++: <https://gist.github.com/Baekjoon/82ed59e4713f5286001ddcbba644849b>
- Java: <https://gist.github.com/Baekjoon/8c41844868ecb0c58a44>

퍼즐

<https://www.acmicpc.net/problem/1525>

- 8퍼즐을 푸는 문제
- 배열로 상태를 저장할 수가 없다

퍼즐

<https://www.acmicpc.net/problem/1525>

- 상태를 저장하는 방법
- 같은 수가 없기 때문에, 순열로 생각해서 몇 번째 순열인지를 저장하는 방법
 - 1727번 문제 응용
- map을 이용해서 저장하기
 - `map<vector<int>,int>`
 - `map<string,int>`
 - `map<int,int>`

퍼즐

<https://www.acmicpc.net/problem/1525>

- 0을 9로 바꾸면, 항상 9자리 숫자가 나오기 때문에, 이를 이용해서 문제를 풀 수 있다

퍼즐

<https://www.acmicpc.net/problem/1525>

```
queue<int> q; q.push(start);
map<int,int> d; d[start] = 0;
while (!q.empty()) {
    int now_num = q.front();
    string now = to_string(now_num);
    q.pop();
    int z = now.find('9');
    int x = z/3;
    int y = z%3;
    // 다음 페이지
}
```

퍼즐

<https://www.acmicpc.net/problem/1525>

```
for (int k=0; k<4; k++) {
    int nx = x+dx[k];
    int ny = y+dy[k];
    if (nx >= 0 && nx < n && ny >= 0 && ny < n) {
        string next = now;
        swap(next[x*3+y], next[nx*3+ny]);
        int num = stoi(next);
        if (d.count(num) == 0) {
            d[num] = d[now_num] + 1;
            q.push(num);
        }
    }
}
```

퍼즐

<https://www.acmicpc.net/problem/1525>

- C++: <https://gist.github.com/Baekjoon/6fa3fdc760b4ffc95d75>
- Java: <https://gist.github.com/Baekjoon/1b496dcf92f128468aca>

이모티콘

<https://www.acmicpc.net/problem/14226>

- 화면에 이모티콘은 1개다
- 할 수 있는 연산
 - 이모티콘을 모두 복사해서 클립보드에 저장
 - 클립보드에 있는 모든 이모티콘을 화면에 붙여넣기
 - 화면에 있는 이모티콘 중 하나를 삭제
- S개의 이모티콘을 치는데 걸리는 시간의 최소값을 구하는 문제

이모티콘

<https://www.acmicpc.net/problem/14226>

- 화면에 이모티콘의 개수 s 와 클립보드에 있는 이모티콘의 개수 c 가 중요하다
- 복사: $(s, c) \rightarrow (s, s)$
- 붙여넣기: $(s, c) \rightarrow (s+c, c)$
- 삭제: $(s, c) \rightarrow (s-1, c)$
- $2 \leq S \leq 1,000$ 이기 때문에 BFS 탐색으로 가능하다.

이모티콘

<https://www.acmicpc.net/problem/14226>

- C++: <https://gist.github.com/Baekjoon/195fcf8798fe3b5b9ae8e46765734265>
- Java: <https://gist.github.com/Baekjoon/5fb20656b011a9842b8e68a44328a398>

물통

<https://www.acmicpc.net/problem/2251>

- 세 물통 A, B, C가 있을 때
- C만 가득차있다
- 어떤 물통에 들어있는 물을 다른 물통으로 쏟아 부을 수 있는데, 이 때에는 앞의 물통이 빌 때까지 붓거나, 뒤의 물통이 가득 찰 때까지 붓게 된다
- 이 과정에서 손실되는 물은 없다
- 이 때, A가 비어있을 때, C에 들어있을 수 있는 양을 모두 구하는 문제

물통

<https://www.acmicpc.net/problem/2251>

- 3차원 배열을 만들 필요는 없다
- 중간에 물이 손실되지 않기 때문에
- 첫 번째 물통, 두 번째 물통에 들어있는 물의 양만 알면 세 번째 물통에 들어있는 물의 양을 알 수 있다

<https://www.acmicpc.net/problem/2251>

```
queue<pair<int,int>> q;
q.push(make_pair(0, 0)); check[0][0] = true; ans[c] = true;
while (!q.empty()) {
    int x = q.front().first, y = q.front().second;
    int z = sum - x - y;
    q.pop();
    // x -> y
    // x -> z
    // y -> x
    // y -> z
    // z -> x
    // z -> y
}
```

<https://www.acmicpc.net/problem/2251>

```
// x -> y
ny += nx; nx = 0;
if (ny >= b) {
    nx = ny-b;
    ny = b;
}
if (!check[nx][ny]) {
    check[nx][ny] = true;
    q.push(make_pair(nx,ny));
    if (nx == 0) {
        ans[nz] = true;
    }
}
```

<https://www.acmicpc.net/problem/2251>

- C++: <https://gist.github.com/Baekjoon/f1f802661214359bd2ff>
- C++: <https://gist.github.com/Baekjoon/68ea2da7363fc088972a>
- Java: <https://gist.github.com/Baekjoon/5c7aba583dd3bc298b5d>