지금 만나러 갑니다 풀이

- 그래프의 정점에 추가적인 정보를 더해서 정의해야하는 문제입니다. 문제에선 오리와 육리 둘이 등장하지만 이동하는 방법이 같으므로 한 오리에 대해서만 생각합시다.
- 그래프의 정점을 단순히 현재 위치한 점 x로 정의하면, x에서 $x-2^t$ 또는 $x+2^t$로 이동해야 하는데 지금이 몇 일차인지 알 수 없기 때문에 인접한 정점을 정의할 수 없습니다.
- 정점의 정의를 (x,t) = x번 정점에 도착했을 때 지금이 t일 차인 상태로 정의하면 각 정점을 방문할 때마다 인접한 정점을 정의할 수 있습니다.
- BFS나 DFS 그래프 탐색의 시간 복잡도는 $\mathcal{O}(V+E)$입니다. 오리는 20번 이상 점프할 일이 없으므로 정점의 개수 V는 최대 $500,000 \times 20 = 10,000,000$ 입니다.
- 오리와 육리는 서로 독립적으로 움직일 수 있으므로 시작 정점만 다르게 해서 그래프 탐색을 두 번 진행합니다. 오리나 육리가 둘 다 정점 (x,t)를 방문할 수 있다는 것은 t일차에 점 x에서 만날 수 있다는 의미입니다.
- 따라서 $1 \le x \le N, 0 \le t \le 20$ 인 정점 (x,t)에 대해서 오리와 육리 모두 방문할 수 있는 정점을 찾고, 그러한 정점이 여러 개라면 가장 작은 t를 찾으면 정답입니다.

소스 코드

```
#include <bits/stdc++.h>
```

```cpp
using namespace std;

int main() {
    int n, a, b;
    cin >> n >> a >> b;

    auto in_range = [&](int x) {
        return 1 <= x && x <= n;
        };

    auto bfs = [&](int start) {
        queue<pair<int, int>> q;
        // (x, t) = x번 정점에 도착했을 때 지금이 t일차이다.
        // 크기가 작은 vector를 여러 개 만들면 오버헤드가 커서 순서를 바꿈
        vector<vector<int>> v(20, vector<int>(n + 1));

        q.emplace(0, start);
        v[0][start] = true;

        while (!q.empty()) {
            auto [t, x] = q.front();
            q.pop();

            int y = x + (1 << t);
            if (in_range(y) && !v[t + 1][y]) {
```

```
                q.emplace(t + 1, y);
                v[t + 1][y] = true;
            }

            y = x - (1 << t);
            if (in_range(y) && !v[t + 1][y]) {
                q.emplace(t + 1, y);
                v[t + 1][y] = true;
            }
        }

        return v;
        };

    vector<vector<int>> da = bfs(a), db = bfs(b);
    for (int t = 0; t < 20; t++) {
        for (int x = 1; x <= n; x++) {
            if (da[t][x] != 0 && da[t][x] == db[t][x]) {
                cout << t << '\n';
                return 0;
            }
        }
    }

    cout << -1 << '\n';
```

```
}
```