Доказать сложность $O(|V| + |E|)$ используемого алгоритма

В задаче реализуется алгоритм поиска в ширину. Алгоритм основан на обходе вершин графа "по слоям". На каждом шаге есть множество "передовых" вершин, для смежных к которым производится проверка, относятся ли они к еще не посещенным. Все еще не посещенные вершины добавляются в новое множество "передовых" вершин, обрабатываемых на следующем шаге. Изначально в множество "передовых" вершин входит только вершина-источник, от которой и начинается обход.

Алгоритмическая сложность $= O(|V| + |E|)$, где $|V|$ - число вершин в графе, $|E|$ - число ребер в графе. При этом, для каждого ребра и каждой вершины алгоритм выполняет константное число действий.

```
while (!q.empty()) {
        int v = q.front();
        q.pop();
        for (int i = 0; i < countries[v].borders.size(); i++) {
            int to = countries[v].borders[i];
            if (countries[v].color == countries[to].color) {
                cout << "-1";
                exit(0);
            }
            if (countries[to].color == -1) {
                countries[to].color = (countries[v].color == 0 ? 1 : 0);
                q.push(to);
            }
        }
    }
```

Вопросы вызывала данная конструкция. А именно вложенные циклы. Внешний цикл

```
while (!q.empty()) {
```

отвечает за перебор всех узлов графа. **q -** это очередь в которую мы будем помещать узлы, и пока она не пуста (т.е. узлы существуют), мы будем их доставать. **v** – извлекаемый узел. Внутренний цикл

```
for (int i = 0; i < countries[v].borders.size(); i++) {
```

отвечает за все смежные узлы с проверяемым. Связь смежности выражается через рёбра графа. Получается, каждая итерация **while** проверять одно вершину $|V|$ **плюс** её связи $|E|$.