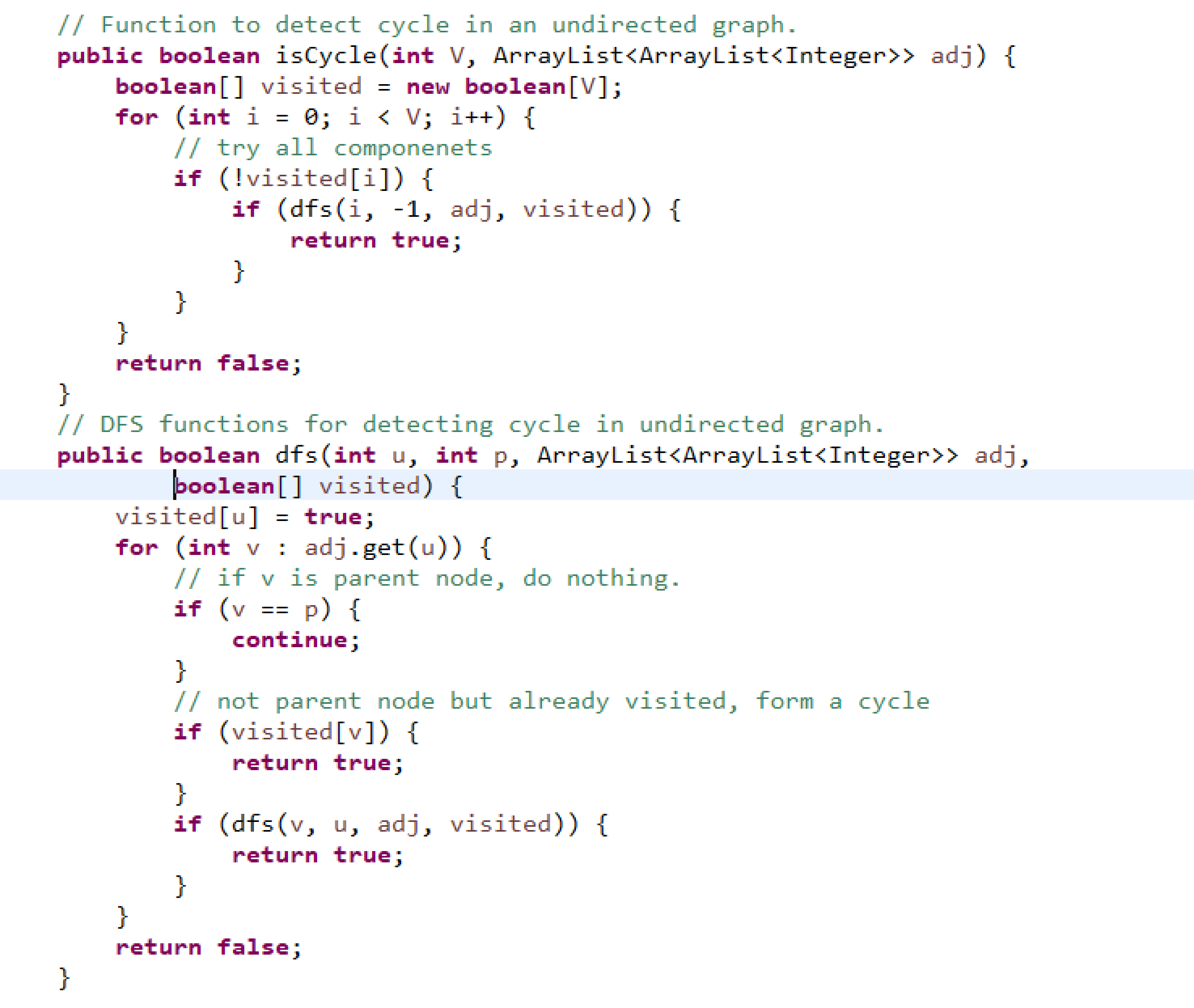

```java
    // Function to detect cycle in an undirected graph.
    public boolean isCycle(int V, ArrayList<ArrayList<Integer>> adj) {
        boolean[] visited = new boolean[V];
        for (int i = 0; i < V; i++) {
            // try all componenets
            if (!visited[i]) {
                if (dfs(i, -1, adj, visited)) {
                    return true;
                }
            }
        }
        return false;
    }
    // DFS functions for detecting cycle in undirected graph.
    public boolean dfs(int u, int p, ArrayList<ArrayList<Integer>> adj,
            boolean[] visited) {
        visited[u] = true;
        for (int v : adj.get(u)) {
            // if v is parent node, do nothing.
            if (v == p) {
                continue;
            }
            // not parent node but already visited, form a cycle
            if (visited[v]) {
                return true;
            }
            if (dfs(v, u, adj, visited)) {
                return true;
            }
        }
        return false;
    }
```