This code defines a Graph class with methods for DFS, BFS, cycle detection, and bipartite checking. The DFS and BFS methods print the nodes in the specified order (left or right). The detect_cycles method identifies cycles in the graph.

For bipartiteness, the is_bipartite method uses BFS to color nodes alternately and checks for any conflicts in coloring. If there are no conflicts, the graph is bipartite.

Now, let's discuss how to determine whether an undirected graph is a tree. A graph is a tree if:

1. It is connected.
2. It is acyclic.
3. The number of edges is exactly one less than the number of nodes

The running time for checking whether an undirected graph is a tree or not depends on the algorithm used. A simple approach involves DFS or BFS, and both have a time complexity of O(V + E), where V is the number of vertices, and E is the number of edges in the graph.