# Java week 10

**public** **class** DepthFirstTraversal<T> **extends** AdjacencyGraph<T> **implements** Traversal<T>

{

List<T> ourGraph = **new** ArrayList<T>();

List<T> visited = **new** ArrayList<T>();

@Override

**public** List<T> traverse() **throws** GraphError

{

**for**(**int** i = 0; i<getNodes().size(); i++)

{

**if**(visited.size() <getNodes().size())

{

@SuppressWarnings("unchecked")

T startNode = (T) getNodes().toArray()[i];

**if**(!visited.contains(startNode))

{

recursiveDepthFirstTraversal(startNode);

}

}

**else** **break**;

}

**return** ourGraph;

}

**public** **void** recursiveDepthFirstTraversal(T node) **throws** GraphError

{

visited.add(node);

ourGraph.add(node);

Set<T> neighboursSet = getNeighbours(node);

@SuppressWarnings("unchecked")

T[] neighbouringNodes = (T[]) neighboursSet.toArray();

**for** (**int** i = 0; i < neighbouringNodes.length; i++)

{

T n = neighbouringNodes[i];

**if** (n != **null** && !visited.contains(n))

{

recursiveDepthFirstTraversal(n);

}

}

}

}