# Java week 11

**public** **class** ReferenceCountTopologicalSort<T> **extends** AdjacencyGraph<T> **implements** TopologicalSort<T>

{

HashMap<T, Integer> map = **new** HashMap<T, Integer>();

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

@Override

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

{

initialise();

setUpReferenceCounts();

addToSort();

System.***out***.println(map);

System.***out***.println(sortedNodes);

**return** sortedNodes;

}

**private** **void** initialise()

{

**for** (T node : getNodes())

{

map.put(node, 0);

}

}

**private** **void** setUpReferenceCounts() **throws** GraphError

{

**for** (T node : getNodes())

{

**for** (T successor : getNeighbours(node))

{

**int** references = map.get(successor);

**if** (map.get(successor) != **null**)

{

map.put(successor, ++references);

}

}

}

}

**public** **void** addToSort() **throws** GraphError

{

**while**(sortedNodes.size() < getNodes().size())

{

**for** (T node : getNodes())

{

**if** (map.get(node) != **null** &&map.get(node).intValue() == 0)

{

sortedNodes.add(node);

**for** (T successor : getNeighbours(node))

{

Integer references = map.get(successor);

**if** (references != **null**)

{

map.put(successor, references - 1);

}

}

map.remove(node);

**break**;

}

}

}

}