CS 293 Compensation Assignment 1
Due, Sat Oct 29, 11:59 pm on Moodle

In this assignment, you will first construct a ***undirected*** graph according to the partial implementation given in the constructor function of class Graph (see Graph.cpp). The constructor takes in three parameters: number of nodes in the graph (numNodes) and number of undirected edges in the graph (numEdges). It then constructs a random graph with numNodes nodes and numEdges ***undirected*** edges. Each node in the graph has a value, which is an integer in the range 1 through N, both inclusive. You are required to implement the adjacency information in the graph using adjacency lists. You have to implement the adjacency lists and the Node class yourself.

Once the graph is constructed, you are required to do a modified depth-first search (DFS) on the graph. The modification from the usual DFS procedure is in respect of the following three points:

- In usual DFS, if a node that is visited once (i.e. entered into the DFS stack once) is reached a second time, the search procedure backtracks and doesn't explore the visited node again. In the modified DFS implementation, if a visited node is reached a second time, the search procedure must not backtrack, but must continue the search and enter this node into the stack again. Only after a node has been visited two times, if it is reached a third time during the modified DFS, should the modified DFS backtrack.
- In our modified DFS, each time a node is visited (including when it is visited a second time), we must insert the integer value in the node in a Binary Search Tree (BST).
- There is no deisgnated start vertex in the graph, and it is perfectly possible that the graph has multiple connected components. In such cases, your modified DFS must be run once on each connected component, so that eventually every vertex in the graph is visited at least once. Every time you complete your modified DFS on one connected component and start the search on another connected component, you must construct a new BST to insert the integer values of nodes in the new component. Thus, you should have as many BSTs as the number of connected components of the graph.

After the modified DFS is completed on the entire graph, you must print the following information in exactly the following format:

No. of connected components:
No. of nodes visited once:
No. of nodes visited twice:
No. of nodes that are present in a cycle:
Each BST that has been generated using the printBST function provided.

You must upload Graph.cpp, Graph.h and assumptions.txt as usual in a tar zipped folder named <roll_number>_C1.tgz