CSC 228 Data Structures and Algorithms, Spring 2018
Instructor: Dr. Natarajan Meghanathan

Project 7: Binary Tree: Design and Implementation of an Iterative Algorithm (using the Stack ADT) to do a Preorder Traversal of the Vertices

Due: March 28, 11.59 PM (submission through Canvas)

The objective of this project is to design and implement an iterative algorithm (using the Stack ADT) to do a preorder traversal of the vertices in a binary tree and find the vertex ID with the maximum data as well as the maximum data. Each vertex in the binary tree have an associated data value.

You are given the code for a binary tree implementation (including all the associated classes) and the main function that will accept two files as input: binaryTree-2000001.txt and data-2000001.txt.

The binaryTree-2000001.txt contains the entries storing the structure of a binary tree of 1000000 internal nodes plus a root node as well as another 1000000 leaf nodes (a total of 2000001 nodes). Assume the root NodeID is 0 and the preorder traversal starts from the root node.

The data-2000001.txt file contains the data associated with the 2000001 nodes.

The code given to you has the implementation of the recursive procedure to do a preorder traversal of the binary tree and determine the vertex (nodeID) with the maximum data as well as the associated data value. You could go through the recursive implementation to get an idea for finding the node with the maximum data and the associated maximum data value. The code for the Stack ADT (class Stack and class Node) is also included along with the other classes in the file given.

Your task in this project will be to design and implement the code for an iterative algorithm (non-recursive) that uses the Stack ADT to traverse the nodes of a binary tree in a preorder fashion and determine the vertex (nodeID) with the maximum data as well as the associated data value. The iterative algorithm is to be implemented as a member function of the BinaryTree class as indicated in the code provided.

The main function has the code setup to call the recursive and iterative procedures on the binary tree object (setup based on the two input files) as well as the timers setup to calculate the run times of these two procedures.

After you implement the iterative algorithm, you will run the code for five trials and report the running times for both the procedures for each of the trials as well as the average value. Compare the average running times for the recursive and iterative procedures and interpret the results.

What to submit (one PDF file in Canvas):

(a - 15 pts) Pseudo code of the iterative procedure of using a Stack to do a preorder traversal of the vertices in a binary tree and show that its time complexity in terms of the number of push (or pop) operations is $\Theta(n)$ where 'n' is the number of vertices in the graph.

(b - 15 pts) Illustrate the working of your pseudo code/algorithm of (a) on a small binary tree of 10 vertices (just use the node IDs) and show the listing of the vertices visited in the preorder fashion.

(c - 45 pts) Code for all the classes and functions (including the code for the iterative algorithm implemented).

(d - 10 pts) Screenshots of your execution of the code for five trials showing the run times of both the recursive and iterative procedures as well as the maximum data value and nodeID with the max. data.

(e - 15 pts) Tabulate the results of (d) and determine the average run times of the recursive and iterative procedures as well as compare the magnitude of the average run times? Which procedure (recursive or iterative) takes more time and why?