CS 251: Intermediate Software Design

Program Assignment 4, part B
Due Monday, April 23rd, 2007

This assignment builds on your solution to part A of programming assignment 4 to use several more patterns to implement a program that will build and traverse a binary tree using various traversal strategies. The patterns will be used as follows:

- **Singleton**, which is used to implement an `Options` singleton that parses and keeps track of the command-line options.
- **Adapter**, which will be used to integrate your `LQueue` and `AQueue` into a new `Queue` and `Queue_Adapter` class hierarchy that can be used to dynamically select which type of queueing strategy to use in the program at runtime. The use of Adapter ensures that no changes are required to the existing `LQueue` and `AQueue` classes.
- **Factory**, which is used to create the appropriate types of queueing and traversal strategies indicated by the `Options` Singleton.
- **Iterator**, which is used to retrieve each element in the binary tree one item at a time, using various traversal orders, e.g., in order, pre order, post order, and level order.
- **Visitor**, which is used to perform an operation on each node that is visited.

Your implementations only need to define an iterator for level order traversal, though your design should be capable of being extended to handle the other traversal orders, as well. Moreover, your visitor implementation need only print the contents of the tree when visited, though again your design should be capable of being extended to handle other types of visitors.

Graduate students need to implement the following additional patterns (which are optional for undergraduates):

- **Abstract Factory** and **Factory Method**, which are used instead of individual `Factory` functions to consolidate all the factories into a single concrete factory class.
- **Bridge**, which is used to avoid exposing "naked" pointers and to simplify memory management, e.g., by reference counting throughout the program.