Comp 121: Data Structures (Spring 2002) **Programming Assignment 3 (** Due: 2002/04/16)

The dynamic dictionary abstract data type was specified in Programming Assignment 2. For this assignment, you are to implement this ADT in the C++ programming language using the AVL search tree as the underlying data structure. The find operation in your AVL tree can be implemented much the same as in Assignment 2. The insert operation must be considerably enhanced to incorporate the rebalancing of trees that become unbalanced. Implement the remove operation by **lazy deletion** – mark nodes corresponding to keys that are removed for deletion at a later point in time, and perform the actual removal of these nodes when a triggering condition (described below) is satisfied.

Here are further details on how you should handle lazy deletion:

- Keep a count (**dd_count**) of the number of "real" data items in the dictionary, and a count (**node_count**) of the number of nodes in your AVL tree. It is always the case that (node_count ≥ dd_count), with equality if there are no nodes in the AVL tree marked for lazy deletion.
- You will have two parameters **threshold** and **ratio** which together trigger the removal of nodes marked for deletion.
- The condition that triggers updates is:

  ((dd_count > threshold) and (node_count > ratio * dd_count))

  – i.e., the the dictionary is large enough, and the ratio of nodes marked for deletion is high enough, for us to want to trigger the actual removal of nodes marked for deletion.
- The removal of nodes marked for deletion is achieved by constructing a new AVL tree comprised of only the nodes in the original tree that had not been marked for deletion. This can be done by traversing all the nodes in the original tree (in pre-/ in-/ post order – which of three traversals would you prefer, and why?), and inserting each node not marked for deletion into the new tree.

Credit will be awarded for efficient implementation. Extensively **test** your ADT, using the **white/ glass box** testing technique. Focus your testing on (i) the balancing operations, and (ii) the lazy deletion.

As with all programming assignments, you must submit a cover sheet, design plan, etc.