Start work early.

Name: _____

This assignment focuses on understanding and implementing a data structure that is described at a high level. Please refer to the previous assignments for important instructions on the allowable use of resources and the requirements for electronic and hardcopy submission. Use the newsgroup for questions, clarifications, and general discussion related to this homework and class. Ensure that everything you submit is neat and easy to comprehend; otherwise, it will be ignored, with zero credit.

1. (1 pt.) Package your solutions to the programming questions as in previous assignments. Submit your work via `http://chaw.eip10.org/u/`. *After* submitting your work, complete the following:
   File name: _____ Size, in bytes: _____
   MD5 checksum: _____
   Timestamp: _____

> We define a *simple digital trie*[1] to be a nonempty rooted labeled tree in which each non-root node is labeled with a single digit $(0, \ldots, 9)$, and in which no two siblings have identical labels. The root r has an empty label.
>
> We associate a trie node $n(z)$ with every non-negative integer z as follows: If $z < 10$ then $n(z)$ has label z and parent r. Otherwise $n(z)$ has label $z \bmod 10$ and parent $n(\lfloor z/10 \rfloor)$.
>
> A *marked simple digital trie* is a simple digital trie in which each node may be associated with an optional *mark* (separate from its label).
>
> A marked simple digital trie is said to *contain a key* (non-negative integer) z if it contains a *marked* node $n(z)$ (and the other nodes implied by the recursive definition of $n(z)$, either marked or unmarked).
>
> A marked simple digital trie is said to *represent a set K of keys* if it contains exactly the keys in K (i.e., all the keys in K and no others).

[1] usually pronounced "try," although some prefer "tree."

2. (25 pts.) Depict a marked simple digital trie that represents the following set of keys, using the usual graphical notation for labeled trees, and using an asterisk $*$ to adorn marked nodes: $\{1, 3, 343, 2939, 48902, 22, 983001, 344, 35, 129\}$.

3. (25 pts.) Describe simple algorithms for the following, by providing pseudocode or very precise English descriptions. Explain briefly why the algorithms are correct.

   (a) to determine whether a marked simple digital trie T contains a key k.

   (b) to add a key k to a digital trie T.

   (c) to remove a key k from a marked simple digital trie T. [Hint: Check that the algorithm removes *only* k.]

[additional space for answering the earlier question]

[additional space for answering the earlier question]

4. (25 pts.) Characterize, as precisely as possible, the running time of the algorithms you describe for Question 3 as a function of the inputs T and k. Prove your answers as precisely as you can.

5. Prove or disprove: If T_1 and T_2 are two marked simple digital tries representing a set K of keys, then T_1 is isomorphic to T_2. [Informally, we may interpret "isomorphic" as "identical" in this context.)

6. (99 pts.) Modify your implementation of a record manager for Homework 2 to use a marked digital trie (instead of a k-search-tree) as the underlying data structure. [Hint: Instead of a binary mark, we may use a suitably typed mark to store the value part of a key-value pair.]

   The API-like text user interface should be identical to that of the earlier version.