Problem Set 4

Assigned: June 18
Due: June 25

Problem 1

Describe an algorithm to compute `Next(x,T)` in a binary search tree `T`. If `x` is an element in the set `T`, `Next(x,T)` should return the next larger element. If `x` is not an element or is the largest element, then `Next(x,T)` should return **false**. The algorithm should run in time proportional to the height of the tree.

Problem 2

Suppose that we use a hash table of size m=11 and we insert the keys 10, 24, 20, 32, 49, 46, 42, 33. For convenience, use zero-based indexing (i.e. the indexes go from 0 to 10). Show the final state of the hash table, assuming that we use:

A. Chaining, with the hash function $h(k) = k \bmod 11$.

B. Linear probing, with with $h(k, i) = (k + i) \bmod 11$.

C. Double hashing, with the hash function $h(k, i) = (k + i \cdot (1 + k \bmod 9)) \bmod 11$.

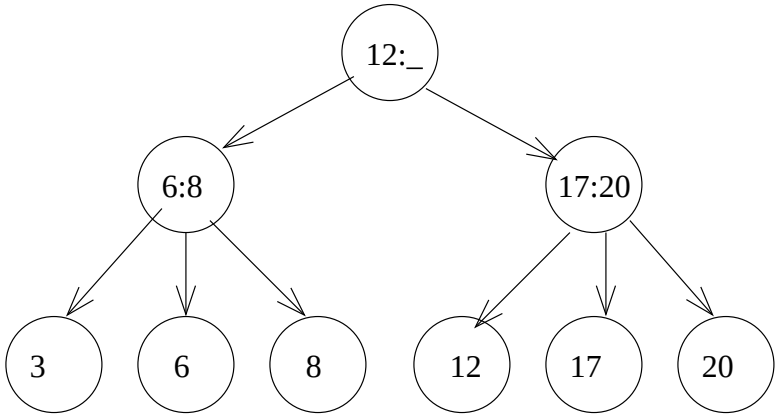
In (B) and (C), k is the number being hashed; i is an iterator over probes for a single Add operation. For instance, if you wish to add (or look for) the value 100, then you first try to place it at index $h(100, 0)$. If that is occupied, you next try $h(100, 1)$, then $h(100, 2)$ and so on.

Problem 3

Show what happens to the 2-3 tree below when the following operations are performed in sequence:

ADD(18), ADD(9), ADD(15), DEL(18), DEL(8).

Show the state of the tree after each operation. You need not show the transient states of the tree during the middle of the operations.



Problem 4

Describe a data structure for a set of integers of arbitrary size that supports the operations listed below with the given running times. You may assume that you are given an upper bound on the possible size of the set $|S|$. Give a brief description of what is involved in each operation. You may assume the data structures that have been described in class. (Thus, for instance, if you use a modified heap, you may say "Use the insert operations on heaps," without detailing what that involves.)

ADD(x,S) – Worst case time $O(\lg(\mid S \mid))$
DELETE(x,S) – Worst time $O(\lg(\mid S \mid))$
MEMBER(x,S) – Avg. time $O(1)$. Worst case time $O(\lg(\mid S \mid))$
MIN(S) – Worst case time $O(1)$.
MAX(S) – Worst case time $O(1)$.
NEXT(x,S) – Given an element x of S, find the next larger element in S. Avg. time $O(1)$.
Worst case $O(\lg(\mid S \mid))$
PRED(x,S) – Given an element x of S, find the next smaller element in S. Avg. time $O(1)$.
Worst case $O(\lg(\mid S \mid))$.