CSC C78F 2000
Assignment 2

due: Wednesday, October 18, 2000

1. A binary tree is *ideally weight balanced* if, for every node x in the tree, the number of nodes in its two subtrees differs by at most 1.

   (a) Give an $O(n)$ time algorithm that, given a sorted array of n elements, constructs an ideally weight balanced binary search tree. Briefly justify the correctness of your algorithm and why it runs in $O(n)$ time in the worst case.

   (b) Give an $O(n)$ time algorithm that, given an n node binary search tree, transforms it into an ideally balanced binary search tree for the same elements.
   Briefly justify the correctness of your algorithm and why it runs in $O(n)$ time in the worst case.

2. Consider a binary search tree with *seven distinct* elements; the tree is perfectly balanced (that is, of height 2), and rooted at *root*. In this question, we will consider two slightly different methods for searching the tree for a key k. For each method, we will be interested in the *expected* time to search for k, when k is chosen at random from amongst the keys in the tree.

   (a) Consider the following search method SEARCH1($root, k$):

```
SEARCH1(r,k)
/*Return a pointer to a node with key k in subtree rooted at at r.*/
  if k = key(r) then
    return r
  elsif k > key(r) then
    return SEARCH1(rchild(r),k)
  else
    return SEARCH1(lchild(r),k)
  end if
end SEARCH1
```

   Counting each "=" or ">" test as a comparison, what is the *expected* number of comparisons to do a search, where the expectation is over the random choice of k from amongst the keys in the tree, with all of them being equally likely? Briefly explain your reasoning and show your work.

   (b) Next consider the search method SEARCH2($root, k$):

```
SEARCH2(r,k)
/*Return a pointer to a node with key k in subtree rooted at at r.*/
  if k > key(r) then
    return SEARCH2(rchild(r),k)
  elseif k = key(r) then
    return r
  else
    return SEARCH2(lchild(r),k)
```

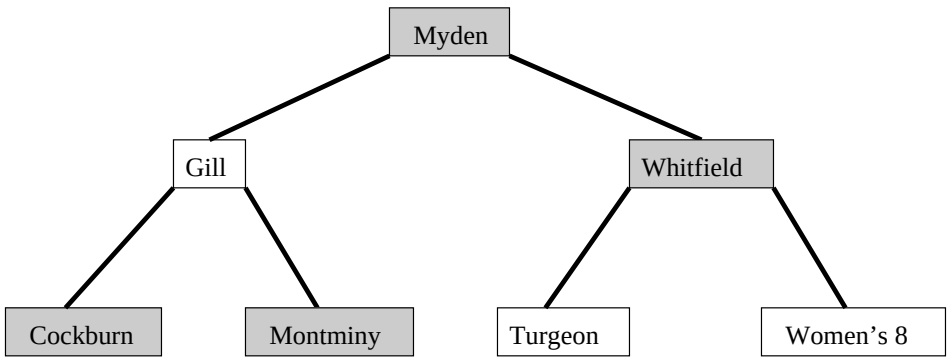
end if
end SEARCH2

Again, suppose that each time we do a search, the element to be sought is chosen at random from amongst the seven elements in the tree. Compute the *expected* number of comparisons using SEARCH2.

(c) What if, instead of a perfectly balanced tree of height 2, we started with a perfectly balanced tree of much larger height, say 10, and considered the same two experiments above. Which of the two procedures, SEARCH1 or SEARCH2, would yield the smaller expected number of comparisons? Justify your answer.

3. The Red-Black tree below stores all Canadian medal winners at the Olympic games up to Tuesday September 26, where the order is alphabetic according to last name:

Karen Cockburn, Ontario, Trampoline
Nicolas Gill, Quebec, Judo
Anne Montminy, Quebec, Diving
Curtis Myden, Alberta, Swimming
Mathieu Turgeon, Ontario, Trampoline
Simon Whitfield, Ontario, Triathlon
Women's Eight, majority from Ontario, Rowing



(a) Which nodes are red and which black?

(b) In a surprise move the athletes from Quebec declare their independence from Canada. Give the new Red-Black tree obtained by first deleting Nicolas Gill, and then Anne Montminy. Show all intermediate steps!

(c) Of course b) didn't happen – what really happened was that the US Softball team applied for political asylum in Canada. Give the new medal tree after their petition is granted, by inserting a node "Softball". Again show all intermediate steps.

(d) Trampoline rules! Rotate Matthew Turgeon to the top of the original medal tree and show that the resulting tree can be colored in exactly one way to obtain a valid Red-Black tree. Also give a Red-Black Tree in which Karen Cockburn is on top, or explain why such a tree doesn't exist.

4. Consider the abstract data type BARCHART. An object of the abstract data type is a set of vertical lines with their lower endpoints on the x axis. Each vertical line L has an x coordinate $x(L)$ and a height $y(L)$. In other words, $(x(L), y(L))$ is the coordinate of the upper endpoint of the line L. Both $x(L)$ and $y(L)$ are floating point numbers. No two vertical lines in a BARCHART have the same x coordinate.

   The operations of this ADT are:

   DELETE(B, L)**:** Deletes line L from the set B. If L was not in B, then this operation has no effect.

   ADD(B, L)**:** Adds line L to set B. If B previously contained a line with the same x coordinate, that line is no longer an element of B.

   DISPLAYHEIGHT(B, a, b)**:** return the maximum height of the lines in B with x coordinate greater than or equal to the floating point number a and less than or equal to the floating point number b. If there is no line in B with x coordinate in this range, return 0.

   (a) Give a data structure for this ADT such that all operations have worst case time complexity $O(\log n)$, where n is the size of the set B. Justify why your operations have the desired worst case time complexity.

   (b) Implement your data structure and test it thoroughly. Properly document your code and test cases.

   Details about input and output format and submitting your program can be found on the course website.