Computer Science B63 | February 1, 2007
Scarborough Campus | University of Toronto

Homework Assignment #2
Due: February 14, 2007, by 1:10 pm

1. *Please complete and attach (with a staple) an assignment cover page to the front of your assignment. You may work alone or with one other student. If you work in a group, write both your names on the cover sheet and submit only one copy of your homework.*

2. *If you do not know the answer to a question, and you write "I (We) do not know the answer to this question", you will receive 20% of the marks of that question. If you just leave a question blank with no such statement, you get 0 marks for that question.*

3. *Your paper will be marked based on the correctness and completeness of your answers, and the clarity, precision and conciseness of your presentation.*

Question 1. (10 marks) In this question, you must use the insertion and deletion algorithms as described in the "Balanced Search Trees: AVL trees" handout posted on the course web site.

Insert into an initially empty AVL tree each of the following keys, in the order in which they appear in the sequence: 17, 7, 8, 14, 19, 6, 10, 21, 15, 12, 9, 11.

Show the resulting AVL tree T. (Only the final tree should be shown; any intermediate trees shown will be disregarded, and not given partial credit.)

From AVL tree T, delete 14, and show the resulting tree.

In the two trees you must show the key and balance factor of each node.

Question 2. (30 marks) A seaport has a number of ships awaiting cargo. Each ship is moored at a pier p and has a remaining loading capacity of c tonnes. You must design a data structure D that stores the ships (including their pier location and remaining capacity) and supports efficient implementation of the four operations listed below.

When cargo arrives at the seaport to be loaded on a ship, you will need to decide which ship to load it on. We assume that all ships are sailing to the same destination. We don't want a single shipment to get split up, so a shipment of s tonnes must be loaded completely onto one ship whose remaining capacity is at least s. Of course, there may be several ships with sufficient capacity: you must choose to load on a ship with the smallest possible pier number (since our "better customers" get the lower pier numbers).

Thus your data structure D must support the following four operations:

INSERT(D, x): If x is a pointer to a new ship that is docking, insert the ship pointed to by x into D.

DELETE(D, x): If x is a pointer to a ship in D that's about to set sail, remove that ship from D.

FIND(D, s): Return a pointer to a ship in D such that: (a) its remaining capacity is at least s tonnes **and** (b) its pier number is minimum over all ships in D with capacity of at least s. If no ship in D has a remaining capacity of at least s tonnes, then return NIL.

LOAD(D, x, s): If x is a pointer to a ship in D, load s tonnes on it. If the capacity remaining on this ship was c, it is thus decreased to $c - s$.

In this question, you must explain how to implement D as an *augmented AVL tree* and explain how each of the above operations is executed. The worst-case run time of each of the above operations should be $O(\log n)$ where n is the number of ships in the data structure D.

Since this implementation of D is based on a data structure and algorithms described in class and in the AVL handout, you should focus on the extensions and modifications needed here (you do not have to reproduce code or details provided in class or in the handout). Note that you *cannot* assume that pier locations are distinct: there may be several ships that have the *same* location (and may have different capacities).

a. (6 marks) Give a precise and full description of your data structure. Illustrate this data structure by giving an example of it on some collection of docked ships of your own choice.

b. (24 marks) Explain how to implement each of the above four operations in $O(\log n)$ time and explain why, in each case, your algorithm achieves this time complexity.

Question 3. (20 marks) This question is a programming assignment. To see its description follow the link given in the "Assignments" section of the course web page.