CSE 5311 Lab Assignment 1

Due July 7, 2009

Goals:

1. Review of binary search trees.
2. Understanding of randomized search trees (treaps).
3. Exposure to bin packing, a topic in Notes 18.

Requirements:

1. Write (and test) a C/C++ program implementing the first-fit decreasing method for bin packing. Your program should use a treap augmented in two ways:

   a. Subtree sizes to allow computing bin numbers by determining the rank of a node. In other words, a node's position in an inorder traversal is used in place of the bin number as the key.

   b. Each subtree root will store the maximum unused capacity appearing in the bins for that subtree.

   The input should be read from standard input (`stdin` or `cin`).

   a. The first line will contain the integer capacity of the bins followed by the number of objects to be packed.

   b. Each of remaining lines will include the integer size of one object. These will not exceed the bin capacity and will be in descending order. There could be repeats.

   The output is simply the bin numbers, one per line, assigned to the input objects. All other tracing should be disabled in the version you submit. Bin numbers should start at 1 (not 0).

   Your program must compile and execute on OMEGA. There should be a comment near the beginning of your code indicating how to compile on OMEGA.

2. Email your code (as attachments) to `miaoju@uta.edu` before 12:45 pm on July 7. The subject should include your name as recorded by the University.

Getting Started:

1. You may borrow from the code at `http://www.cs.fiu.edu/~weiss` (or other places - besides each other), but be sure to give appropriate credit in your comments.
2. The number of objects will not exceed 200,000.
3. Be careful with your randomly generated priorities.

4. It is trivial to write a table-based program to perform this task in $\Theta\left(n^2\right)$ time. Your program should run in $\Theta(n \log n)$ expected time.

5. The following pseudocode describes the processing for one object:

```
if treap root indicates the existence of a bin that can hold this
   object then
   Use a BST-style search to find the node with minimum rank (e.g.
         bin #) that can hold this object
   Only the maximum unused capacities will require updating
else
   Initialize a new bin to receive this object
   Insert this bin as leaf node on rightmost path in treap:
         Rotations will be determined by random priorities
         Subtree sizes will require updating
         Maximum unused capacities will require updating
```