CS 361 – Lab #5 – Skip Lists

Today we will practice implementing a skip list data structure. Specifically, you will write the code that will help us search for a key, and create a new node in the skip list. Refer to class notes and examples of how these operations should work.

1. Create a new directory called Lab05. Copy my source code files from http://cs.furman.edu/~chealy/cs361/lab05/. You only need to make changes to SkipList.java. The file Driver.java contains code for testing the skip list. You may want to change the set of input key values, trying a short sample initially, and later a much longer one.

2. Before starting your implementation, take some time to review how the skip list is set up. The SkipList class has some important attributes, and a default constructor. To simplify the implementation, I have assumed a maximum height. The highest and lowest possible integer values are used as the bookends for each level of the skip list. To save you from having to repeatedly call a random number generator, I have included a random number attribute that you should use for the insertion. This will also make it easy for you to test your output to make sure the list is being built correctly. Use Java's bitwise operators to help you cycle through the bits of this random number to decide whether to continue replicating nodes upward.

   Also take a minute to review Node.java. Each node in the skip list should have 4 pointers to its neighbors. I have included a level attribute for debugging purposes.

3. In SkipList.java, there is space for you to complete the implementations of search() and add(). For your convenience, I have included comments based on our class notes. What does the final skip list look like if we insert: 8, 17, 22, 27, 38, 50, 5, 12, 20, 25, 31, 39, 62, 55 ?

4. Just something to think about: how you would change the implementation so that after every 32nd insertion, we generate a new random number, and continue the insertions based on the bits of this new number?