CSCI 2110 Data Structures and Algorithms
Laboratory No. 8
Week of November 20th, 2017

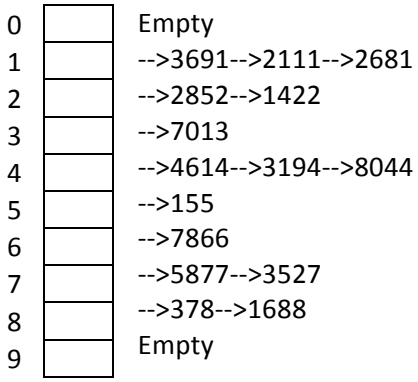
This is technically your last lab in CS 2110. Next week's lab will be a makeup one.
Deadline for submission: Sunday, November 26th, 2017, 11.55 P.M.

Review of hashing (from the lectures): A hash table is a simple array structure. Whenever a key needs to be mapped onto the hash table, it uses a hash function. A common hash function used is key%table_size.

For example, if the table_size is 10 keys are 1008, 2540, 3467 and 789, they get mapped to locations 8, 0, 7 and 9, respectively.

However, multiple keys may map to the same location. For instance, in the above example, 3467 and 2487, will both map to location no. 7. A hash collision is said to occur. One way of resolving a hash clash is called separate chaining. Rather than storing the keys in the array, each array location contains a linked list. The keys are stored in the linked list.

The following is an example in which 15 keys 3527, 7013, 2681, 7866, 8044, 1688, 5877, 1422, 3194, 4614, 2852, 155, 2111, 3691, and 378 are mapped onto a table of size 10:



In the above example, the number of times a collision has occurred is 7. The longest linked list has a size of 3. We say that the hash table has a load factor of 15/10 = 1.5 (Load factor = number of keys mapped/table size. Note that this is not integer division).

Exercise: Your task is to write a program that creates a hash table of various sizes and various numbers of keys and collect statistics on the load factor, number of collisions and the longest list length. Follow these steps.

1. Get the user to enter the table_size.
2. Declare an arraylist of linked lists of that size. Assume that the linked list store integer objects.
3. Get the user to enter the number of keys to be hashed. Generate that many ***random*** keys in the range 1 to 10000. You don't need to check for duplicate keys.
4. For each key generated, determine where it should be mapped (pos = key%table_size).
5. Go to the arraylist index pos and add the key into the linked list in that position.
6. After all the keys have been added, enumerate all the linked lists, find the number of collisions and the longest list length.
7. Run the program a number of times, experimenting with different number of keys (10, 20, 30, 40, …, 100). Keep the table size as 10. Create a table showing your experimental values:

Table size	Number of keys	Load factor	Number of collisions	Longest list length
10	10			
10	20			
10	30			
etc.				

A sample run of the program will be as follows:

```
Enter the size of the hash table: 10
Enter the number of keys to be hashed: 20

Hash Table created:
-->8180-->6490
-->empty
-->3802-->3272
-->9563-->9493
-->324-->3004-->3714
-->2375-->635-->2595-->4655-->2045
-->876
-->497-->3427
-->2258
-->529→3419

Statistics:
Table size: 10
Number of keys: 20
Load factor: 2
Number of collisions: 11
Longest list: 5
```

What to submit:
Your source code, sample runs of the program in a text file, and the table of experimental results in a text file.