HW6 Hashing & Symbol Tables
Due Friday, April 6, 2018

1. Given the input {4371, 1323, 6173, 4199, 4344, 9679, 1989} and the hash function h(k) = k mod 10, show the results: (No programming, do this by hand)
   a) Separate chaining hash table
   b) Hash table with linear probing
   c) Hash table with secondary hash function h2(k) = 7 - (k mod 7)

2. What are the advantages & disadvantages of various collision resolution strategies identified in the textbook?

3. An empty hash table has the capacity of 100 and you add 6 elements with the hash codes:
   100, 0, 1199, 1299, 1399 and 2
Using linear probing and a division hash function, where will these elements be placed in the table? Note: a division hash function is one we've been using as an example, h(k) = k % m where m is the size of the array or table. Avoid powers and multiples of 2 for m! Really good values of m are primes not close to a power of 2. In other words, m = 13 - not so great because it's close to 16. But m = 23 is ok because it's far enough away from 16 and 32.

4. Same question as #3, but with double hashing where h2(k) = 1 + abs [h(k) mod 98] where h(k) is Java's hashCode() function.

5. A hash table has 180 elements in an array of size 200.
   a) What is the load factor?
   b) For each of the 3 hash methods, what is the average number of elements that are examined during a search?

6. You need to put 1000 elements into a hash table but one criteria is an average search to be 2 table elements. For each of the three hash methods, how large does the array need to be?

7. Programming problem.
Input: the number file from HW5 - Sorts (numbers.txt). Remember to ask the user for the file name. I'll be using my data file to test your program.

Process:
1) By hand, estimate the number of collisions for search hits and misses for size M = 11,000, 15707, 17111, and 25111. Create a matrix like this and fill in the blanks for numbers.txt:

Size M	11,000	15,707	17,111	25,111
N Keys				
10000				

2) Write a program that uses hashing with separate chaining on numbers.txt for the 4 conditions above and capture the following details:
a) how long to hash each of the 4 conditions
b) how many collisions for each of the 4 conditions
c) What is the key with the longest linked list (chain) and how long is the chain?

In order to determine collisions, you'll need to create an array of size M, hash each number, attempt to put it in the corresponding array location. If the location is already occupied, it's a collision and use linear probing to put the value into the related linked list.

3) Write a program that uses hashing with open-addressing on numbers.txt for the 4 conditions above and capture the following details:
a) how long to hash each of the 4 conditions
b) how many collisions for each of the 4 conditions

c) What is the "furthest" collision?

In order to determine collisions, you'll need to create an array of size M, hash each number, attempt to put it in the corresponding array location. If the location is already occupied, it's a collision and use open-addressing to put the value into the next available position.

For both 2 & 3, use this hash algorithm:

h(k) = k % 1231

Output:
1) the matrix you filled in
2) a), b) and c) for the two hash methods.

EXTRA CREDIT
8. Download the **morse.csv** file from the book's website. It's in CSV format so you'll need to convert to another format that can be read by a Java program. Write a Java program that converts letters & numbers to Morse code, and the reverse.

User Interface:

- A menu that asks the user which conversion they want
- The user can select the option to enter the name of a text file that contains the message or enter from the command line (input)
- The user can select the option to enter the name of a text file that contains the converted message (output) or display to the screen.

Input:

- If plain text input, no spaces between characters except between words
- If morse code, one space between characters, 2 spaces between words

Output:

- if plain text output, no spaces between characters in a word, spaces between words.
- If morse code output, 1 space between characters, 2 spaces between words.

Data Structure:
Implement as a hash table with two values: the morse code and the character, and the key to be hashed will be the ASCII code of the letter/number.