CSCI 241 Data Structures
Fall 2017
Assignment 2

Submitting Your Work

This assignment is worth 15% of the grade for the course. Submit your Java via the **Assignment 2 Submission** item on the course web site. You must submit your assignment by 11:00am on Wednesday, November 15.

Your assignments will be evaluated on correct functionality and conformance to the coding standards described at the end of this assignment specification.

The Task

Your task is to write a Java class HashTable, which is to be used by a supplied program Assignment2.java. The objective of the complete program is to be able to find the *n* most frequently occurring words in one or more text files whose names are listed on the command-line. For example, we can find the 5 most frequently occurring words in the book "Alice In Wonderland":

```
> java Assignment2 5 AliceInWonderland.txt
the:  1643
and:  872
to:  729
a:  632
it:  595
```

Note that the first command-line argument must be an integer, representing the number of words to be displayed and the second and subsequent command-line arguments are the names of files that are to be read for their words to be counted.

Provided Java Program

The Java program `Assignment2.java` is provided for your use. It will also be used to test your hash table when the assignment is graded.

The `Assignment2.java` program does the following things:

1. Checks that the first command-line arguments is an integer.
2. Creates an empty hash table by calling the constructor of your `HashTable` class.
3. For each file listed on the command line:

   a. Opens the file and reads each line.

   b. Breaks each line into individual words.

   c. Adds each word to your hash table by invoking the `add()` instance method of your `HashTable` class.

4. Once all the words from all the input files have been added to the hash table, it finds the word with the highest occurrence count by calling the instance method `highcount()` of your `HashTable` class.
5. Displays that word and then deletes it from the hash table, using the instance method `delete()` of your `HashTable` class.
6. Steps 4 and 5 are repeated for the number of times specified by the integer found in the first command-line argument.

Also provided is the Java file `DataItem.java`, which defines the `DataItem` class, to be used in your hash table. Each `DataItem` object contains two instance variables:

- `String word`, the word stored in the data item
- `int count`, the occurrence count for the word

What You Need to Do

You must write a Java class `HashTable`, which provides:

1. A constructor, which takes no parameters and creates an empty hash table.
2. Instance method

   ```
   public void add(String word)
   ```

   This method must search for word in the hash table, if it finds word, it must increment the word's occurrence count; if it does not find the word, it must insert it into the hash table, giving it an initial occurrence count of 1.

3. Instance method

   ```
   private void rehash()
   ```

   This method must be called by the `add()` method when the occupancy of the hash table reaches 50%. That is, when the number of words stored in the hash table is greater than or equal to half the size of the table (the number of distinct index values). Method `rehash()` must create a new hash table about twice the size of the old table and move all entries from the old hash table to the new hash table. All subsequent words will then be added to the new hash table.

4. Instance method

   `public void delete(String word)`

   This method will delete `word` from the hash table. If `word` is not found in the hash table, this method must simply return without any error message.

5. Instance method

   `public DataItem highcount()`

   This method must return the `DataItem` containing the word with the greatest occurrence count of all the words stored in the hash table.

Other requirements:

1. Your hash table is to use the direct chaining method of collision-handling.

2. Your hash table must use a hash function which derives its hash table index using all the characters in the word that is being added or located in the hash table. That hash function should be of a form which can minimize collisions.

3. Your hash table must not exceed 50% occupancy and must be rehashed to a new table that is about twice the size whenever the table occupancy exceeds 50%.

Coding Standards

1. Use meaningful names for variables that give the reader a clue as to the purpose of the thing being named.
2. Use comments at the start of the program to identify the purpose of the program, the author and the date written.
3. Use comments at the start of each method to describe the purpose of the method, the purpose of each parameter to the method, and the return value from the method (if any).
4. Use comments at the start of each section of the program to explain what that part of the program does.
5. Use consistent indentation.

Provided Files

You are provided with the files, `Assignment2.java` and `DataItem.java`, described above, as a starting point for this assignment. You are **not** permitted to change these files.

The full text of several novels has been provided for you to use as test data.