CIS2168 Fall 2019 003-004 Assignment 9
Hash Table: Implementation

1. Objectives

This assignment will help you to:

- Understand how Hash Table with Open Addressing and Linear Probing works
- Understand how HashMap works
- Learn to how to implement Hash Table

2. Overview

You are given the classes KWHashMap.java, HashtableOpen.java, and HashtableOpenTest.java. The codes are almost the same as what were given for the lectures. In HashtableOpen.java, the ordering of methods are changed. The purpose is to facilitate your completion of this assignment.

Modify HashtableOpen class in the following ways

- Complete size()
  - Return the number of living keys currently in the hash table.
- Add toString() in Entry class inside HashtableOpen class
  - It is an override the toString() in class Object.
  - Returns a String that contains the key and value pair separated by a comma.
    Example when key is the String type, value is the String type:
    key, value
    “E, excited”
- Add toString() in HashtableOpen class
  - It is an override the toString() in class Object.
  - Returns a String that contains all living entries currently in the hash table. Separate two adjacent entries using a new line character and enclose all entries within a pair of brackets [].
    So if key is the String type, value is the String type, when the return String is printed, you will see something like this:
    ```
    [
    E, excited
    G, guilty
    ]
    ```
- Add two data fields for counting the following
  - The number of times method find(….) has been called so far
  - The total number of probes made in all the calls to find(…)
- Modify find(…) as follows
  - Count and display the number of probes made in this call to find(…)
  - Include this count of probes in the data field for the total probe count.
  - Update the total number of calls to find(…) so far
- Add method averageProbes()
  - Compute and return the average number of probes made per call to find(…)
- Complete remove(…)
  - Remove the given key from the hash table and return its associated value
  - If the key is not found, return null.

Modify HashtableOpenTest.java in the following ways

1. **Optionally (NOT REQUIRED)** add the code to create and populate your own hash table.
2. Add the code to call the following methods at least once
   - size()
   - toString() for the HashtableOpen class
3. Add the code to call averageProbes() at least two times
   - Once after inserting 5 entries
   - Once after searching for the 5 keys in the array keys
4. Add the code to call remove(…) at least two times
   - Once for removing an existing key
   - Once for removing a non-existing key

3. Coding Requirements and Hints

You must use the given classes KWHashMap, HashtableOpen, HashtableOpenTest.

HashtableOpen.java:

Add or revise the methods/data fields in the following order:

- public int size()
  - **DO NOT INCLUDE the deleted keys.**
- toString() in Entry class inside HashtableOpen class
  - **Use the toString() in K and V passed to Entry<K,V> class in you implementation**
- toString() in HashtableOpen class
  - Call the toString() in Entry<K, V> class to get the String representation of each living entry
  - **DO NOT INCLUDE the dummy entries or nulls in virgin slots**.
- Modify find(…)
  - This method need just minor modification for the required task
- averageProbes()
  - Compute and return the average number of probes made per call to find(…)
  - **NOTE: the average may not be a whole number.**
- remove(…)
  - You can call find(…) to locate the given key to be removed
  - Once the key is found in a slot, write a dummy entry in this slot by having this slot reference the dummy entry in the constant DELETED.

HashtableOpenTest.java

- You don't have to create your own hash table.
- You can just use the hash code in the given code.

4. Major Steps

- Understand the related classes I gave you in the lectures and template.
  i. KWHashMap.java, HashtableOpen.java, HashtableOpenTest.java
- Revise the class HashtableOpen.java. At one method at a time. HashtableOpenTest.java.
- The order of revision in Item 3 is recommened.

5. Submission Requirements & Grading

This assignment is **due by 11:50PM,**
Both sections: 11:59PM, Monday, Nov. 18, 2019.

Please see the file CIS2168 003-004 Assign9 Submission Requirements.pdf for more details.

NOTE:

- The point distribution for this assignment is:
  - CODE REVIEW: is required. If you don't do the code review, you will get 0 in this assignment.
  - CORRECTNESS: 90 points if code review is completed. Otherwise it's 0.
  - PROGRAM STYLE: 90 points if code review is completed. Otherwise it's 0.