Assignment 6
Hashing

Release Time	Due Date
11/27/2016	12/08/2016

Objectives

- To practice using hash functions
- Practice developing high-performance solutions.
- Analyze the advantages and the disadvantages of hashing method.

Problem Specification

YG is a teaching assistant in WMU, with your excellent help in assignment 4, his supervisor decides to give him a promotion to take over the security of CS3310. And his first task is to protect student data. Please help him develop a JAVA tool to encrypt student names using a hash function. Here we go:

- Create a hash function to encrypt the students' name, the input of the hash function should be one name, and the output should be an integer. Please design your hash function as below, so that it has as few collisions as possible (collision means different names hash to same integer value).

  Note: the hash function:

  a. Sets up a hash key (any value you want),

  b. For each name do "exclusive or" operation with your hash key,

  c. Then return the result of "exclusive or" , and we treat this value as the hash value for the name

  i.e.

```
int hash = 7;
for (int i = 0; i < strlen; i++) {
    hash = hash^charAt(i);
}
```

- Then save the hashed values and names into a hashTable.txt file (not a good idea since this will expose encryption, but to keep our task simple, we will make do with this simplification).

- Besides, your tool should also implement the function, getInfo(hashValue), which returns the real name given an input hash value. If the value is not in the database, just return "student does not exist". Similarly, function getInfo(name) should return the hashValue.

- Please make your tool handle even billions of data.
- Make sure your tool is dependable, readable, and has a friendly UI.
    a. When you run your code, it should load the hashTable.txt automatically (if it does not exist, just create a new one)
    b. Then the welcome page should have 5 options:
        1. Add student (get a name, then compute its hashValue and insert into your simple textual database)
        2. Find student (your program should handle both hashValue and name searching, i.e., if hashValue is input, then find the corresponding name, or if a name is input then output the corresponding hashValue)

           For example, If Jack's hash value is 109098, then after a call to getInfo(109098) print "Jack 109098", or after a call to getInfo(Jack) the result should be the same (i.e., print "Jack 109098").
        3. Delete student (just combine the find student and delete operation)
        4. Show all information (just print the updated HashTable)
        5. Exit
- Additionally, time and space complexity analysis into a report is also required.

Design Requirements

Code Documentation

For this assignment, you must include documentation for your code as generated by JavaDoc. You should have JavaDoc comments for every class, constructor, and method. By default, JavaDoc should output html documentation to a subfolder within your project (/dist/javadoc). Make sure this folder is included when you zip your files for submission. You do not need to submit a hard copy of this documentation.

Hint: http://stackoverflow.com/questions/4468669/how-to-generate-javadoc-html-in-eclipse

Coding Standards

You must adhere to all conventions in the CS 3310 Java coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming classes, variables, method parameters and methods.

Testing

Make sure you test your application with several different values, to make sure it works.

Assignment Submission

- Generate a .zip file that contains all your files, including:
    - Source code files
    - Including any input or output files
- Javadocs
- A report containing analysis of your implementation (theoretical as well as empirical)