CSC 103 Programming Assignment #5 10/27/16

Due date – **Friday December 16nd @ Noon**

This project is to be done by a pair (2) of students. No groups of three or one without the instructor’s permission. The phrase “I like to work alone” is not an acceptable reason for not having a partner. You will need to learn to work together. Make sure that both of you understand all aspects of the assignment. Note: If the work is not evenly done, this will no doubt be seen during exams.

**Project # 5 Analysis of Hashing versus Double Hashing**

Your task for this program is to fill up a Table using hash techniques and count the number of collisions that happen in placing the elements in the table. You will report the average number of collisions that happened in placing all the elements and the maximum number of collisions that an individual element had.

You will then repeat this process using a different Table class that uses Double hashing techniques. You will once again report your average and maximum numbers of collisions. A comparison will hopefully show an improvement in the second technique.

You will be given an input file which contains 200 names and 9-digit numbers. The names will be your elements <String> and the numbers will be your keys <Integer> .

Summary of program:

Your program will start with reading the file and placing the names into the table based on the key given. The table will be created from the given table class (which you will be able to alter as needed. As prime numbers work the best for hashing, I recommend a table size of 241, because 239 is also a prime. As this table will be over 80% full, we will also get a fair amount of collisions. As you place each element, you will need to keep track of how many times the spot you wanted to place the item was already used. Each of these is a collision. (**I am not interested in how many collisions each location of the array is having, just the number of collisions per attempted placing into the table**.) Store this in an array. Repeat and create a second array for double hashing. Then report the results for both tests. With the same hash function and input, we should all get the same results.

You will need to alter the **Table** class with just enough code to be able to report back on the number of collisions for each time you place an element, but nothing else needs to be changed.

You will need to create a second Table class with the needed alterations for double hashing. Call this class **TableDoubleHash**. This class will have almost all of the same code, so just create a second file and make the alterations.

You will create these classes to finish the project:

* **Table** (template given)
* **TableDoubleHash** (table with a bunch of changes)
* **HashTesting**- main driver of program.

The inputfile used is data stored in text format (not binary, so that we can read it) with one name and 9-digit number on each line and spaces between fields. It is organized like this:

**Name 9DigitNumber**

The sample data file given is called **names.txt** and will be used as a starting point to test your program. A copy of this file is given in with these program directions. I recommend that you do some testing, such as using a small subset of the input and placing by hand the next name and then checking that it was placed where you expected.

Tips for good grades:

* Make sure you use comments where needed and use variable names that make sense, some of your grade will depend on program style as well as the use of your program.
* Update the comments in the class file, to include your names and any new information
* You will lose points for things like not indenting, or naming variables in non-descriptive ways. Do no leave in debugging code, or commented out code.
* I use jGrasp and the java version that is in the lab computers. So make sure that your programs work with this.
* Test your own projects thoroughly before you hand them in.
* Late projects will not be accepted so plan ahead.

The three classes you are using for this project should be in separate files. Name them **HashTesting.java, Table.java** and **TableDoubleHash.java**. If you do not name these files correctly, you will lose points. Javadoc is not needed for submission.

Hand in electronically – (NOT E-mail!!!)

In S-drive CSC 103 folder:

1. Create folder called **projectfive\_firstname\_lastname**
2. Place these three script files named above in folder.
3. Place any other documentation such as a readme file in here as well.