CSC 225 - SPRING 2017
ALGORITHMS AND DATA STRUCTURES I
PROGRAMMING ASSIGNMENT 3
UNIVERSITY OF VICTORIA

1 Programming Assignment

The assignment is to implement a hash table for strings of characters which uses open addressing and quadratic probing to resolve collisions. The data structure used to represent the hash table must be an array.

Your implementation must be be able to insert an arbitrary number of strings into the hash table as well as remove strings from the hash table. (Hint. you cannot use a special indicator string to represent a deleted location in the hash table. Instead, each location will have to maintain a status and a value.) Furthermore, the load factor of your hash table must remain in the range $[0.25, 0.75]$ at all times. Therefore, when inserting or removing a string that would move the load factor outside this range, the hash table will need to be resized. You may use any resizing scheme you choose. Note, the size of the hash table must be prime after each resize operation.

A Java template has been provided containing empty functions `hash insert`, `find`, `remove`, and `resize`. Your task is to write the body of each of the functions. You must use the provided Java template as the basis of your submission, and put your implementation inside the provided functions in the template. You may not change the name, return type or parameters of any of the provided functions. You may add additional functions and classes as needed. Since you are are only permitted to submit one file, any extra classes must be contained in the `HashTable.java` file.

The `main` function in the template contains code to help you test your implementation by entering test data or reading it from a file. You may modify the `main` function, but only the contents of the provided functions (and any functions or classes you have added) will be marked, since the `main` function will be deleted before marking begins. Please read through the comments in the template file before starting.

2 Test Datasets

The dataset for this assignment will be a collection of place names (cities, towns, etc.). A set of input files containing test data are available on conneX. You should ensure that your implementation behaves correctly on these test files before submitting. It may also be helpful to test your implementation on a variety of other inputs, since the posted data may not cover all possible cases.

3 Evaluation Criteria

The programming assignment will be marked out of 20, based on a combination of automated testing (using test data similar to the ones posted on conneX) and human inspection.

Score	Description
0 - 5	Submission does not compile or does not conform to the provided template.
6 - 10	The implemented hash table is substantially inaccurate on the tested inputs.
11 - 16	The implemented hash table is partially complete, with some of the required functions not working correctly.
17 - 20	The implemented hash table is complete and implements each of the required functions correctly.

To be properly tested, every submission must compile correctly as submitted, and must be based on the provided template. **If your submission does not compile for any reason (even trivial mistakes like typos), or was not based on the template, it will receive at most 5 out of 20.** The best way to make sure your submission is correct is to download it from conneX after submitting and test it. You are not permitted to revise your submission after the due date, and late submissions will not be accepted, so you should ensure that you have submitted the correct version of your code before the due date. conneX will allow you to change your submission before the due date if you notice a mistake. After submitting your assignment, conneX will automatically send you a confirmation email. If you do not receive such an email, your submission was not received. If you have problems with the submission process, send an email to the instructor **before** the due date.