Milestone 2 – Hash Table, Due February 26, 2025

The following files will be provided to you, for completion of your milestone:

- hash_node.h // header file containing hash node structure
- hash_table.h // header file containing hash table class
- json.hpp // header file for processing json files
- milestone2.json // json file containing test cases and its transactions
- milestone2_config.json // json configuration (properties) file
- milestone2.cpp /* cpp file containing main, which does the following:
    - Reads configuration file (json format) to:
        - retrieve inputFile (test case file (json format)
        - retrieve outputFile (text file containing generated output)
        - retrieve errorLogFile (text file containing error messages)
    - process inputFile test cases
    - write output to outputFile */

Write a basic hash table implementation, which uses the files listed above, and includes the following in a separate cpp file:

- hash_table.cpp – implementation file that contains the following methods:

    1. getTable - return the hash table
    2. getSize - return the size of the hash table
    3. calculateHashCode – perform hashing function
    4. isEmpty - Check if the HashTable is empty
    5. getNumberOfItems - return number of items in the hash table
    6. add - adds a new node to the hash table
    7. remove – remove node with key value
    8. clear - remove all entries from the table
    9. getItem - returns pointer to the HashNode
    10. contains - check if a node with key exists in the table
    11. printTable - print out contents of hash table to console and output file

The total number of points for this milestone is 74, which will be based upon the following:

- Each submitted/modified file must have student's name (-10% of total milestone points if missing)
- Each submitted/modified file must include description of changes made to a program, and its change date (1)
- Program compiles with all of the provided files (1)
- The following methods run without errors:

1. getTable - return the hash table (2)
2. getSize - return the size of the hash table (2)
3. calculateHashCode – perform hashing function (2)
4. isEmpty - Check if the HashTable is empty (2)
5. getNumberOfItems - return number of items in the hash table (2)
6. add - adds a new node to the hash table (2)
7. remove – remove node with key value (2)
8. clear - remove all entries from the table (2)
9. getItem - returns pointer to the HashNode (2)
10. contains - check if a node with key exists in the table (2)
11. printTable - print out contents of hash table to console and output file (2)

- The following test cases are processed, and produce expected output (10 per test case; 50 total)
- Extra Credit – use industry standard test program and/or extract test cases, in separate json test file

Please accept this GitHub Assignment: https://classroom.github.com/a/lECB3-XC