CSE211

DATA STRUCTURES

FALL 2021

ASSIGNMENT 5

Aim: To learn and implement basics of hashing and to improve your understanding of working within constraints (i.e. being have to use the asked function types, learning from the comments…)

What is hashing?

Hashing is the process of converting a given key into another value. It is used for a fast-access data storage in implementations called "Hash Tables".

Let's assume that you have values 5, 9, 23, 4, 78, 8, 42, 7, 1, 271, 251, 420 in a linked list and you want to find 251. Traditionally, you iterate from the beginning and only access 251 in your 11^{th} iteration. However, a hash table with key value of 10 stores data like this:

0: 420
1: 1, 271, 251
2: 42
3: 23
4: 4
5: 5
6:
7: 7
8: 78, 8
9: 9

Simply, takes modulus of these variables and puts them accordingly. Therefore, if you look up for 251, you simply 251 % 10 and get 1 and only check 1's to find 251 for a much higher search speed. This is a simple hash table application.

Another implementation of hashing is for validating data within different data types. Such as encryption and cyrptographic operations. There will be some future details given after your assignment explanation.

Q (100pts): Write a **hash** implementation for both **integer** and **up-to-4-digit string** type inputs into the main with the prototypes given in the source file **main.cpp**. Also, you should use linked list implementation given to you previously this semester: IntSLList. The functions you are asked to build are:

void insertIntoHashTable(int input);
void insertIntoHashTable(const char* input);
bool searchInHashTable(int input);
bool searchInHashTable(const char* input);

Important: Your work should compile & run along with the example main file provided to you. You can compile multiple cpp files using:

```
g++ main.cpp intSLList.cpp
```

Also, make no changes to the given main class except corresponding areas for your functions. You can use existing source and header files **intSLList.cpp** and **intSLList.h** which are totally complete.

You may also want to use comments efficiently, especially if your assignment isn't 100% done for helping us to understand your studies better.

So, if you realized, you are being asked to implement a hash table that can store both integer and up-to-4-digit string values. Yet, your list implementation that should hold values only store integers. How are you supposed to store strings within integers, right? Maybe stoi, but then you cannot differentiate "ASD" from "ADS" since the totals would be the same. Then how?

With hashing. You can hash the string's character values individually into an integer one by one. For example, ASCII value of character A is 65 and B is 66. If you were to store ABB, you can simply store it as "656666" or "666665" within an integer. And if you unhash it you get ABB. That's another use of hashing for a very simple cryptographic implementation.

Wish you all good luck with your last assignment of the semester!

So, here is an execution snippet for helping you to visualize:

```
insertIntoHashTable(1);
insertIntoHashTable(11);
insertIntoHashTable(6);
insertIntoHashTable(121);
insertIntoHashTable("AAA");
insertIntoHashTable("Y");
insertIntoHashTable("30E");
printHashTable();
cout << "search begin:" << endl;
cout << searchInHashTable(11) << endl;
cout << searchInHashTable(10) << endl;
cout << searchInHashTable(66) << endl;
cout << searchInHashTable("AAA") << endl;
cout << searchInHashTable("AA") << endl;
pleaseHelpMeWithHashValues();
```

```
ugurn@ubuntu:~/Desktop/cse211asg5$ g++ main-soln.cpp intsllist.cpp
^[[Augurn@ubuntu:~/Desktop/cse211asg5$ ./a.out
        Printing hash table for hash 0:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 1:
Hash Table values are: 1 11 121 694851
Value types are (0 for int, 1 for string): 0 0 0 1
        Printing hash table for hash 2:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 3:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 4:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 5:
Hash Table values are: 656565
Value types are (0 for int, 1 for string): 1
        Printing hash table for hash 6:
Hash Table values are: 6
Value types are (0 for int, 1 for string): 0
        Printing hash table for hash 7:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 8:
Hash Table values are:
Value types are (0 for int, 1 for string):
        Printing hash table for hash 9:
Hash Table values are: 89
Value types are (0 for int, 1 for string): 1
search begin:
1
0
0
1
0
Printable ASCII [48..90]:
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z
ASCII val. of 'A' is: 65
Value 65 equals to character: A
ugurn@ubuntu:~/Desktop/cse211asg5$
```