CS 5008 Lab 10: Hash tables

The purpose of this lab is to give you experience with implementing functions with hash tables.

Objectives

Upon completion of the laboratory exercise, you will be able to do the following:

- read an implementation of a hash table that uses open address linear probing
- implement the find operation for a hash table
- print the contents of a hash table

Part 1: Pull from class Github

Follow the procedure from prior labs to pull the starter code from Github to your local repository. You will find Lab10 folder that contains the pdf describing the lab and the starter code. These are the files for the starter code:

```
hash.h
hash.c
hash_main.c
```

Part 2: Understand the existing code

Open `hash.c`. In this lab, the hash table is implemented using open address linear probing. The keys are strings (`char *`).

1. What are the members of the `HT` struct? _____

2. What value is used to represent a `DELETE`d cell? _____

Read through the function definitions for `initTable` and `freeTable`. These produce the free the hash table structure, respectively.

Read through the function definition for `insert`.

3. What location is used for inserting the key? (circle answer)
    a. the hashed location (always)
    b. the location of the first null cell in the table
    c. the location of the first deleted cell in the table
    d. the location of the first null cell OR the first deleted cell in the table

Read the function definition for `delete`.

Now, read the code in `hash_main`. The main function creates a hash table, inserts items, finds items, and deletes items. Right now, the code is not completely working. It should compile, though:

```
gcc -Wall hash_main.c hash.c
./a.out
```

Part 3: Implementing functions

As you work through the function definitions, you may want to compile often, as shown above.

1. Lab Exercises 1: Complete the function definition for `printTable`. The function should print the non-null values in the table, along with their locations, in the following format:

```
HASH TABLE CONTENTS:
Num items: 11
----------------------
0 : peanuts
1 : chocolate
2 : tangerine
3 : pineapple
4 : strawberry
5 : rice
6 : papaya
7 : corn
8 : hummus
9 : green beans
10 : yogurt
----------------------
```

2. Lab Exercises 2: Complete the function definition for `find`. It should return -1 if the key is not found in the hash table. It should return the location of the key if it is found in the hash table. Remember, it uses open address linear probing, so the key may not be in the location to where it is immediately hashed.

Pseudocode:

```
If key or t is null, return -1
Hash the key
Calculate the location in the hash table
Assign this location as the originalLocation
While t[location] is not null:
        if(t[location] == key)              // use strcmp for strings
                return location
        location++
        location = location % sizeOfHashTable  // for wrap-around
        if(location == origLocation)
                return -1                   // processed all cells in table
return -1                                   // did not find key
```

3. Compile and run the code. It should be inserting, finding, and deleting properly now.

4. Change the size of the hash table to 17. Compile and run the code to be sure it still works.

5. Create a makefile and ensure that it works. Make sure to test it. Also run valgrind to ensure that there are no leaks.

Checkpoint [50 points]: Show the TA and or the professor, the answers to the questions above and results of your program running. Show the hash table contents at the end of main for size 11 and size 17 hash tables.

Part 4: Finish up & Submission

Test the code in the khoury servers and then push your code: all the files (including the makefile) to your Github repo.