Project 4 Report

1. No bugs to report that I know of.

2. All of my functions satisfies the given big-O requirements

3. High Level pseudocode for:

Note: I implemented hash table with an array of pointers to nodes (for each key) and I also made a “newly created linked list” that kept track of how recent the non-permanent nodes were tampered with. I will call this “newly created linked list” NLCC from this point on.

Hashtable’s set():

-if the hashtable is full, return false

-create the new Node called “add” with next, newer, older pointers = nullptr and set value / key/ permanent

Case 1: (add isn’t permanent and there is nothing in the NLCC originally)

I will set the “head” and “tail” pointers aka m\_newest and m\_oldest of the NLCC to add

I now find the node that the given key hashes to and put it in **temp**

Case 2: (there’s nothing in that bucket yet…)

If add isn’t permament, I will update the NLCC

I will put add as the first item in the bucket

I will iterate the amt of items in the hashtable

Return true;//completed the task

Case 3: (there was something already inside the bucket)

I will constantly iterate through the linked list that starts with temp

If I find a matching key…

If the existing item is permanent, I return false

Else

I will overwrite the value, and update the NLCC given these cases

Case 1:temp is neither the oldest or newest

Case 2: temp is the oldest

Case 3: temp is the newest so I don’t do anything

I will now update the newest so that it’s temp

Case 4: (I need to add a new association in this bucket)

If I exit the above loop, that means I didn’t find a matching key, so I set temp->next to add

I iterate the number of items inside the hashtable

If add isn’t permanent, I will update NLCC with the same cases as above

Return true!//I successfully added something to the hashtable

Hashtable’s touch():

I set **temp** to be the first node inside the given hashed key’s bucket

I iterate through the linked list starting with temp

If I find a matching key…

If temp is permanent, I return false //cannot change anything in NLCC

Case 1: (temp is neither oldest nor newest in the NLCC)

I link temp’s older and newer together

Case 2: (temp is the oldest item)

if there is only one item in the NLCC aka newest=oldest, I return true

I set oldests->newer->older to null and set the new oldest to be

Current oldest’s newer

Case 3: (temp is the newest item)

Return true //nothing to do here

I update the newest

I make current newest’s next temp

I link temp’s older to current newest and temp’s newer to nullptr

I reset newest to temp

Return true;

If I have exited the above loop, I didn’t find matching key…

Return false;

Hashtable’s discard():

If oldest doesn’t exist

Return false

Pass in key = oldest’s key

Pass in value = oldest’s value

I find oldest’s key, and hash it to get the bucket.

I delete the node in the bucket so there is no longer that association

Case 1: there is only one item in the NLC

I delete oldest and set oldest / newest to nullptr

I decrement amount of items in the hashtable

Return true;//I have successfully deleted

Case 2: there isn’t only one item in the NLCC

I have a temp node hold current oldests’s newer node;

I delete oldest

I decrement amount of items in hashtable

I reset oldest to the temp node

I set oldest’s older node to nullptr

Return true; //I have successfully deleted the oldest!