Lab 5-2 Hash Table Code Reflection

I really enjoyed working with hash tables. The vector of linked list works beautifully but also requires quite a bit of careful thought when iterating. My general process was to use a for loop to iterate through the vector and a while loop to look for and iterate through the bucket. I picked up some good pointers from the video. I used the UINT\_MAX value throughout. I hadn’t done the three constructors building off the others so that was neat. I did have one question about how my code compared to what the professor did in the video and one question about the code provided.

My remove() method is much more involved then what was done in the video. It seems to me that his code would remove the first node in the bucket every time. I had iterated though looking for a match on the bidId. His line using the erase()method did help clean up one of my if statements.

I struggled for quite some time on my print method. First, I wasn’t skipping over empty spots in the vector. I got that taken care of but then realized that any bid over $1000 was printing as 0. After trying a few different things, I pulled up a prior lab and found it was doing the same thing. I then looked at one of the videos and saw that when the professor printed the bids on the video the bids over $1000 were coming up as 0. I concluded that the issue has to do with the strToDouble() method not addressing the comma. I tried to fix it but could not find a solution. I did go into my bid file and changed the formatting of the bid column to eliminate the comma. After doing that my bids all printed properly.

Psudocode

Task 1: Define structures to hold bids

Node Structure contains

DECLARE bid

DECLARE key

DECLARE node pointer to next node

Node constructor

DECLARE vector of nodes

DECLARE hash key

Task 2: Initialize the structures used to hold bids

SET vector size

Task 3: Implement logic to free storage when class is destroyed

CLEAR storage using erase() function

Task 4: Implement logic to calculate a hash value using the bid Id as the source for calculating the key

CALCULATE hash value using modulus

Task 5: Implement logic to insert a bid

DECLARE and INITIALIZE key

DECLARE and INITIALIZE node pointer

IF node at key is empty

INSERT node

IF node at key is not empty

ITERATE past last node in bucket

INSERT node

Task 6: Implement logic to print all bids

DECLARE and INITIALIZE node pointer

FOR LOOP to iterate through nodes vector

IF node is empty skip it

IF node contains bid print if

WHILE LOOP to iterate through linked list bucket

IF node next point to a node with bid print it

Task 7: Implement logic to remove a bid

DECLARE and INITIALIZE key to the bid to be removed

CHECK bid at key to see if bidId matches

IF bidId matches

REMOVE bid using erase() function

IF bidId does not match

ITERATE through bucket looking for right bid

REMOVE bid using erase() function

Task 8: Implement logic to search for and return a bid

DECLARE and INITIALIZE key to the bid to be removed

CHECK bid at key to see if bidId matches

IF bidId matches

RETURN bid

IF bidId does not match

ITERATE through bucket looking for right bid

RETURN bid

Here is sample output from running the completed program:

> ./HashTable ~/Downloads/eBid\_Monthly\_Sales\_Dec\_2016.csv

> HashTable.exe Downloads\eBid\_Monthly\_Sales\_Dec\_2016.csv

**Load bids from CSV and display the hash table contents:**

|  |  |  |
| --- | --- | --- |
| **Example Input** | **Choice: 1** | **Choice: 2** |
| **Display** | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 1 | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 2 |
| **Output** | Loading CSV file eBid\_Monthly\_Sales.csv  179 bids read  time: 3069 clock ticks  time: 0.003069 seconds | Key 2: 98094 | Credenza | 57 | General Fund  2: 98273 | Nike Tennis Shoes Size: 11.5 | 84 | Enterprise  Key 5: 98276 | Nike Tennis Shoes Size: 11.5 | 83.99 | Enterprise  Key 8: 98279 | Nike Tennis Shoes Size: 11 | 51.57 | Enterprise  Key 10: 98102 | Battery Cart | 42 | Enterprise  Key 12: 98104 | 3 Ticket Booths | 395.01 | Enterprise  12: 98283 | Jordan Tennis Shoes Size: 11 | 160 | Enterprise  Key 13: 98105 | 2 PS4 Games | 11 | Enterprise  13: 98284 | Jordan Tennis Shoes Size: 11 | 89.01 | Enterprise  ...  ...  Key 176: 98268 | Dayton Pallet Jack | 78.85 | Enterprise  Key 177: 98269 | 5 Extron Control Systems | 25 | General Fund |

Note that Keys 2, 12, and 13 highlighted above indicate key collisions occurred.

**Finding and Removing and existing bid:**

|  |  |  |
| --- | --- | --- |
| **Example Input** | **Choice: 3** | **Choice: 4** |
| **Display** | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 3 | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 4 |
| **Output** | 98109: Whirlpool Washer & Dryer | 225.46 | Enterprise  time: 59 clock ticks  time: 5.9e-05 seconds | {no output shown} |

**Finding a bid that no longer exists:**

|  |  |  |
| --- | --- | --- |
| **Example Input** | **Choice: 3** | **Choice: 9** |
| **Display** | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 3 | Menu:  1. Load Bids  2. Display All Bids  3. Find Bid  4. Remove Bid  9. Exit  Enter choice: 9 |
| **Output** | Bid Id 98109 not found.  time: 12 clock ticks  time: 1.2e-05 seconds | Good bye. |