DTS Lab 5  
hash table

Contents

[Objective 2](#_Toc358034605)

[Prototypes 2](#_Toc358034606)

[Desired Output 4](#_Toc358034607)

[Submission 5](#_Toc358034608)

# Objective

Implement a template class. *HTable*, that will function as a hash table. You will use your SLList.h from day 3 to make implementation easier.

Place all your code in a file named *HTable.h*

**Do not use your DynArray class to implement this hash table.**

# Data Members

The HTable class will need the following private data members:

SLList<Type>\* table;  
unsigned int bucketCount;  
unsigned int (\*hashFunction)(const Type& v);

# Prototypes

The HTable class will need the following public interface:

/////////////////////////////////////////////////////////////////////////////  
// Function : Constructor   
// Parameters : numOfBuckets - the number of buckets  
// hFunction - pointer to the hash function for this table  
// Notes : dynamically allocates the table to numOfBuckets size and  
// assigns the other two data members  
/////////////////////////////////////////////////////////////////////////////  
HTable(unsigned int numOfBuckets, unsigned int (\*hFunction) (const Type &v))  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : Destructor  
// Notes : deletes the table  
/////////////////////////////////////////////////////////////////////////////  
~HTable()  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : Assignment Operator  
// Notes: Deep copy of that internal array of lists (similar to DynArray=)  
/////////////////////////////////////////////////////////////////////////////  
HTable<Type>& operator=(const HTable<Type>& that)  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : Copy Constructor  
// Notes: Call assignment operator  
/////////////////////////////////////////////////////////////////////////////  
HTable(const HTable<Type>& that)  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : insert  
// Parameters : v - the item to insert into the hash table  
// Notes: Use hashFunction to determine which list in the array to add  
// the item to  
/////////////////////////////////////////////////////////////////////////////  
void insert(const Type& v)  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : findAndRemove  
// Parameters : v - the item to remove(if it is found)  
// Notes: Need to use SLLIter to iterate and remove from the list  
/////////////////////////////////////////////////////////////////////////////  
bool findAndRemove(const Type& v)

/////////////////////////////////////////////////////////////////////////////  
// Function : clear  
// Notes : calls clear on each list in the array  
/////////////////////////////////////////////////////////////////////////////  
void clear()  
  
/////////////////////////////////////////////////////////////////////////////  
// Function : find  
// Parameters : v - the item to look for  
// Return : the bucket we found the item in or -1 if we didn’t find the item.  
// Notes: Need to use SLLIter to search the list  
/////////////////////////////////////////////////////////////////////////////  
int find(const Type& v) const

# Desired Output

Compile and run your code with the DTSLab5.cpp file provided via FSO. Your console output should match the following block identically:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*\* LAB 5: \*\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
\*\*\* TEST 1 \*\*\*  
Loading Hash Table 1 with values :  
5 89 2 45 37 9000 20 73 99 1 24  
Finding the given values, plus a couple of ones I didn't put in.. :  
5 89 29 2 45 37 9000 0 20 73 99 1 24 66  
The value 5 was found in bucket : 5  
The value 89 was found in bucket : 9  
The value 29 was found in bucket : -1  
The value 2 was found in bucket : 2  
The value 45 was found in bucket : 5  
The value 37 was found in bucket : 7  
The value 9000 was found in bucket : 0  
The value 0 was found in bucket : -1  
The value 20 was found in bucket : 0  
The value 73 was found in bucket : 3  
The value 99 was found in bucket : 9  
The value 1 was found in bucket : 1  
The value 24 was found in bucket : 4  
The value 66 was found in bucket : -1  
Clearing table..  
  
\*\*\* TEST 2 \*\*\*  
Testing insert, remove, and find..  
The value 0 was found in bucket : 0  
The value 1 was found in bucket : 1  
The value 2 was found in bucket : 2  
The value 3 was found in bucket : 3  
The value 4 was found in bucket : 4  
The value 5 was found in bucket : 5  
The value 10 was found in bucket : 0  
The value 14 was found in bucket : 4  
The value 20 was found in bucket : 0  
The value 30 was found in bucket : 0  
The value 32 was found in bucket : 2  
  
Removing middle 5 numbers..  
  
The value 0 was found in bucket : 0  
The value 1 was found in bucket : 1  
The value 2 was found in bucket : 2  
The value 3 was found in bucket : -1  
The value 4 was found in bucket : -1  
The value 5 was found in bucket : -1  
The value 10 was found in bucket : -1  
The value 14 was found in bucket : -1  
The value 20 was found in bucket : 0  
The value 30 was found in bucket : 0  
The value 32 was found in bucket : 2

# Submission

To submit the lab assignment:

1. Clean, build, and run DTSLab5.cpp with your HTable.h and SLList.h files in Visual Studio (debug mode).
   1. clear up any warnings you encounter.
   2. verify that your output is correct by comparing it to the lab document's Desired Output section, line-by-line.
   3. ensure there are no memory leaks.
2. On your desktop, create a new folder with your name in the following format:
   1. your last name
   2. a comma
   3. a single space
   4. your first name  
      **\* Appropriate capitalization for proper names should be used.**  
      Suitable examples include : "Pollack, Joey"; "De La Paz, Christhian"; "Tjarks, Matthew".
3. Copy your 'HTable.h' and 'SLList.h' files into the folder that you created in step 2. I need both of these files to grade the lab. These are the only files I need and should therefore be the only files you submit.
4. Right-click on the folder and select 'send to->compressed (zipped) folder'.
5. Submit the compressed folder via FSO.