Data Structures (810:052) Lab 13 - File Hash Name:_____

Objective: To experiment with advanced file operations in C++ and random-access file to perform hashing.

Download the following file to your desktop: http://www.cs.uni.edu/~fienup/cs052s10/labs/lab13.zip
Extract this file to the Desktop by right-clicking on lab13.zip icon and selecting Extract All. Code was borrowed from: http://cis.stvincent.edu/html/tutorials/swd/hash/hash.html

Part A: The `lab13.zip` file you downloaded and extracted contains a `hashFile` folder with a Visual Studio C++ project file: `hashFile.sln` inside. Double-click on it to open this project in Visual Studio.

The `makehash.cpp` file contains the `main` function which reads the `hash.txt` file information into a binary, random-access file `hash.dat` which contains a hash table. Read the C++ code to answer the following questions:

a) The "HashTableClass HashTable('w', "hash.dat");" statement in the `main` function invokes the HashTableClass constructor. Describe in English what the constructor does.

b) Study the code and explain how collisions are handled.

After you have answered the questions, raise your hand.

Part B: The `lab13.zip` file you downloaded and extracted contains a `hashFileRead` folder with a Visual Studio C++ project file: `hashFile.sln` inside. Double-click on it to open this project in Visual Studio.

The `readhash.cpp` file contains the `main` function which interactively retrieves records from the binary, random-access file `hash.dat` which contains a hash table. Read the C++ code to answer the following questions:

a) In the HashTableClass::Retrieve method, explain the statement: "DataFile.seekg(RecordNumber * NodeSize, ios::beg);".

- What does seekg do?

- Why the RecordNumber * NodeSize

- What does "ios::beg" mean?

b) How could deletion of a record be handled efficiently?

After you have answered the questions, raise your hand.

EXTRA CREDIT: Implement a Delete method in the HashTableClass.