Semester Project Part 5: Storing Passwords using a Hash Table

Data Structures and Analysis of Algorithms, akk5

Objectives

- To strengthen student's knowledge of C++ programming
- To give student experience reading and parsing strings of commands
- To give student experience in writing a non-linear data structure
- To give the student experience implementing a hash table

Instructions

For this assignment you must write a program which manages a table of user ids, home directories, and passwords.

The table should be implemented as an **open addressing hash table using quadratic probing**. The hash table should have a default size of 23 and use user id as the key for the hash. This is a programming exercise and not an actual password management solution, so user ids, home directories, and passwords can be stored as strings of plain text.

Your program should implement a command prompt (text-based interface) capable of handling the following commands:

exit – exits the program

load *<file>* - parses the contents of the file as if they were entered from the command prompt.

load pwd *<file>* - loads a file containing user ids, home directories, and passwords into the table.

save pwd *<file>* – saves the current table of user ids, home directories, and passwords into a file.

new *<user>* *<pwd>* – creates a new user, sets the home directory to /users/***user***, and sets the password to ***pwd***

remove <user> – removes the specified ***user*** from the table, marks the cell as empty and modified.

clear – removes every entry from the table, resetting the size of the table to 23 and marking every cell as empty and unmodified.

display – displays the user id, home directory, and a * for each entry in the table. Empty cells should display whether they are empty and modified or empty and unmodified.

display *<user>* – displays the user id, home directory, and * for a given user

verify *<user>* *<pwd>* – determines if the password provided matches the user's password.

home <user> <home> – sets the home directory for the given user to home

passwd *<user> <pwd>* - sets the password for the given user to pwd

Guidance

Remember to maintain a list of primes for each doubling of the table's size; you can find one at http://naturalnumbers.org/primes.html

I recommend that you implement a class to store the details for each user; at a minimum this class should store the user id, home directory and password. Since this hash table supports deletion, you might want to consider having the user class support a mechanism for marking cells as empty and as modified or unmodified.

The save feature doesn't need to store empty cells of the table, just the cells that contain user data.

Don't do anything complicated to convert the user id into an integer before hashing it; simply loop over the length of the string and add the ascii values together to form an integer.

This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.

Grading Breakdown

Point Breakdown	
Structure	12 pts
The program has a header comment with the required information.	3 pts
The overall readability of the program.	3 pts
Program uses separate files for main and class definitions	3 pts
Program includes meaningful comments	3 pts
Syntax	16 pts
Implements the HashTable class/classes correctly	8 pts
Implements the User class/classes correctly	8 pts
Behavior	72 pts
Program handles all command inputs properly	
• Exit the program	6 pts
• Create a new user	6 pts
• Edit a user's password	6 pts
• Edit a user's home directory	6 pts
• Remove a user	6 pts
• Display a user's information	6 pts
• Display the user table	6 pts
• Verify a user's password	6 pts
• Load the user table	6 pts
• Save the user table	6 pts
• Load a command script	6 pts
• Clear the user table	6 pts
Total Possible Points	**100pts**
Penalties	
Program does NOT compile	-100
Late up to 72 hrs	-10 / day
Late more than 72hrs	-100

This is an individual assignment. Seeking direct help from students, tutors, and websites such as chegg or stack overflow will be construed as a violation of the honor code.

Header Comment

At the top of each program, type in the following comment:

/*

Student Name: <student name>

Student NetID: <student NetID>

Compiler Used: <Visual Studio, GCC, etc.>

Program Description:

<Write a short description of the program.>

*/

Example:

/*

Student Name: John Smith

Student NetID: jjjs123

Compiler Used: Eclipse using MinGW

Program Description:

This program prints lots and lots of strings!!

*/

Assignment Information

Due Date:

Files Expected:

This project should require the implementation of one or more classes as well as the function necessary for parsing the various inputs. At a minimum I am expecting Hash.cpp, Hash.h, User.cpp, User.h, and main.cpp; you might have more files depending upon your implementation.