CS 50 Homework 5:
hashtable.c

We all remember in Homework 0 we used a hashtable to implement string lookups and checks in our **philspel** program. Now it is time to convert a slightly simplified version of the hashtable code[1] into ~~RISC-V~~ X86-64 assembly, implementing the functions **createHashTable**, **insertData**, and **findData**.

We provide several files for you: **main.c**, **hashtable.c**, **hashtable.h**, and **hashtable_asm.s**. The **Makefile** in the directory builds two separate binaries, **hashtable_c** and **hashtable_asm**. The two binaries are identical except that one builds from the **hashtable.c** and one from **hashtable_asm.s**. Just as in homework 3 the main test file is comprehensive enough that it will be the autograded version.

You will need to implement the three functions in **hashtable_asm.s** so that the tests will pass and the output will be correct. Your createHashTable function can call **malloc** and it will be properly linked (you can see it calling puts in the "need to implement" notification.

This assignment is due on May 31th at 2000 PDT (10:00 pm Davis time). You must work on this assignment individually.

Expected Homework Outcomes

1. Understand the process of converting C code into X86-64 assembly language programs

Hints

This assignment is configured to build on a X86-64 Linux system such as the CSIF Ubuntu systems. If you have WSL it should work fine locally, and it should work fine on an older x86 Mac, but neither have been tested at this time.

The structure layout is straightforward: Pointers (regardless of type) are 8 bytes, while size and used are 4 bytes.

[1] This version excludes the automatic resizing and the freeing of memory functionalities.