CS 277 Lab 6
Choose Your Own Adventure w/ Big Integers
Due 11:59pm Thursday April 10, 2014

Purpose

What do we do on a computer system if we need to support numbers that are larger than the range supported by the primitive data types? For example, what do we do on a 32-bit system if we need to compute values that are much larger than the maximum value for an unsigned 32-bit integer ($2^{32} - 1$)? Believe it or not, this is a common task and is used for important functions like encryption. This lab will explore the challenges of providing support for integers with bit-widths larger than what is supported directly by the ISA.

Assignment

In this lab you will implement a C library that adds support for a new data type **`big_int`** that extends features of the 32-bit **`unsigned int`** data type to support 4096-bit *unsigned integers*. See **`bigint.h`** and **`bigint.c`** to see function descriptions, prototypes and more. Use the provided **`driver.c`** file to write test code and the provided **`makefile`** to build your test executable named **`bigadventure`**. To help get you started:

Step 1: Decide how you wish to define and store your new data type. Edit **`bigint.h`** and change the type definition for **`big_int`** to reflect your design for the data type. You might want to spend some time thinking through what data you will need in order to complete all of the functions in the assignment and define your **`big_int`** data type accordingly.

Step 2: Implement **`new_big_int()`** and **`destroy_big_int()`**. You can think of these functions as the constructor and the de-constructor for a big integer. **`new_big_int()`** should allocate a **`big_int`** and set it to the value of the integer passed as an argument. It should return a pointer to the new **`big_int`**. Use your test driver and GDB to examine your data type and verify it is initialized correctly. **`destroy_big_int()`** should free any memory created to support the data type.

EXTRA CREDIT!

There are three opportunities for extra credit in this lab:

Package #1 Complete **`big_int_equals()`**, **`big_int_lt()`** and **`big_int_gt()`** for 10 points.
Package #2 Complete **`to_hex()`** and **`parse_hex()`** for 5 points.
Package #3 Complete **`to_decimal()`** and **`parse_decimal()`** for 5 points.

All other functions are required. Extra credit points for each package are atomic, meaning you have to implement all of the functions in the package to be eligible for any of the points for the pacakge.

Submission

To submit, create a .tar file named lab6-yourusername.tar containing your `.c` and `.h` files. Email the .tar file to mharmon@lclark.edu before the deadline for the assignment. You do not need to submit your `driver.c` file. This is just provided to you as a tool for writing test code as you develop your libray.

Evaluation

+10 compiles and runs
+10 for each correct function implemented (there are 9).