CS105 – Computer Systems Fall 2023

Assignment 2: Data Lab

Due: Tuesday, September 19, 2023 at 11:59pm PT

The purpose of this assignment is to give you familiarity with bit-level representations of signed integers and floating point numbers and with various operations performed on these data types. Like last week, you will accomplish the goal by solving a series of programming "puzzles."

You must work in a group of two people in solving the problems; partners will be assigned for this assignment. You should complete this assignment using pair programming, and you and your partner should submit one solution. *I strongly recommend that you and your partner brainstorm before coding!*

Getting Started

The materials for this lab are available on the course web page and on the course VM. I strongly recommend that you complete this assignment on the VM.

First, ensure that you are connected to the Pomona network or the Pomona VPN. Then `ssh` to the VM using your Pomona username (e.g., abcd1234)::

```
% ssh USERNAME@itbdcv-lnx04p.campus.pomona.edu
```

and unpack the starter code into your home directory on the VM:

```
% tar xvf /cs105/starters/datalab.tar
```

This will cause a number of files to be unpacked in the directory. Like last week, the only file you will be modifying and is `bits.c`.

Begin by opening the file in an editor and *put both your names* in the comments at the top of the `bits.c` file. Do this right away!!

The `bits.c` file contains a skeleton for each of the 15 programming puzzles. Your assignment is to complete each function skeleton using only *straightline* code (no loops or conditionals) and a limited number of C arithmetic and logical operators. Each function heading tells you what operations are allowed. Further, you are not allowed to use any constants longer than 8 bits. See the comments in `bits.c` for detailed rules and a discussion of the desired coding style.

Compiling the Code

We strongly suggest that you do your work on the course VM. You can be sure that the support programs `btest` and `driver.pl` will work there.

The helper functions work the same as last week, but we've included this information in case you want a refresher.

We have given you a `Makefile` to ease the burden of running the compiler. Type

```
% make
```

to compile everything and

```
% ./btest
```

to test your puzzle solutions (note: btest doesn't test for compliance with coding style or operation limits). If something weird is happening, type

```
% make clean
```

to delete all compiled code and then re-compile from scratch using the `make` command.

The `driver.pl` Program

The `driver.pl` program will be used to grade your assignment. You can determine your current grade yourself by running

```
% ./driver.pl
```

Note that this progam will not award you points if you use disallowed operations or too many operations.

Evaluation

Your code will be run and tested on `project5`. Your score will be computed out of a maximum of 75 points. Each function will be evaluated separately for correctness and performance.

- **Correctness (16 points):** We will use the program `driver.pl` and `dlc`, supplied with the assignment materials, to evaluate your code. No points will be given for a function if `dlc` reports an illegal operator, too many operators, or another error.
- **Performance (12 points):** We will use the programs `driver.pl` and `dlc`, supplied with the assignment materials, to evaluate your code. No points will be given for a function if `dlc` reports an illegal operator, too many operators, or another error.
- **Style (2 points):** For this assignment, "good style" is easy to attain. It means that your files are submitted correctly, your names are present at the top of each file, that your code is understandable and consistently indented, that comments—when necessary to explain—are present and easy to read, and that there is no extraneous material.
- **Feedback (2 points):** An additional 2 points will be awarded for submitting a completed feedback file.

Submission Instructions

When you have finished, submit two files, `bits.c` and `feedback.txt`, on Gradescope. As always, you can download files from the VM to your local machine by running the `scp` command from your local machine. Be sure to tag your partner as your group member and submit both files in the same submission!

Part I: Two's Complement Arithmetic

Table 1 describes a set of functions that make use of the two's complement representation of integers.

Function `isNegative` returns 1 if `x < 0` and 0 otherwise.

Function `divpwr2` returns $x/2^n$, rounded towards zero.

Function `addOK` determines whether its two arguments can be added together without overflow.

Function `absVal` returns the absolute value `|x|` .

Name	Description	Rating	Max Ops
`isNegative(x)`	is `x` negative?	1	6
`divpwr2(x,n)`	$x/2^n$, rounded towards zero	2	15
`addOK(x,y)`	Does `x+y` overflow?	3	20
`absVal(x)`	returns `|x|`	4	10

Table 1: Arithmetic Functions

Part II: Float Arithmetic

Table 2 describes a set of functions that make use of single precision floating point representation of integers. For these puzzles, you may use

Function `float_abs` returns the absolute value of the argument.

Function `float_f2i` casts a float to an int.

Name	Description	Rating	Max Ops
`float_abs(f)`	Returns `|f|`	2	10
`float_f2i(f)`	Returns `(int) f`	4	30

Table 2: Arithmetic Functions

Important Note: The coding rules are relaxed for floats: you may use conditionals and large constants to solve these puzzles. See notes in the starter code for details.

Part III: Feedback

Create a file called `feedback.txt` that answers the following questions:

1. How long did each of you spend on this assignment?
2. Any comments on this assignment?

How you answer these questions **will not affect your grade**, but whether you answer them will.