# 15-213, Fall 20xx

Data Lab: Manipulating Bits

Assigned: Aug. 30, Due: Wed., Sept. 12, 11:59PM

Harry Bovik (bovik@cs.cmu.edu) is the lead person for this assignment.

Harry Bovik（bovik@cs.cmu.edu）是此作业的负责人。

## Introduction

The purpose of this assignment is to become more familiar with bit-level representations of integers and floating point numbers. You’ll do this by solving a series of programming “puzzles.” Many of these puzzles are quite artificial, but you’ll find yourself thinking much more about bits in working your way through them.

该分配的目的是使您更加熟悉整数和浮点数的位级表示。 您可以通过解决一系列编程难题来做到这一点。 这些难题中的许多难题都是人为造成的，但您会发现自己在处理这些难题时会多加思考。

## Logistics

This is an individual project. All handins are electronic. Clarifications and corrections will be posted on the course Web page.

这是一个单独的项目。 所有handins都是电子的。 澄清和更正将发布在课程网页上。

## Handout Instructions

### SITE-SPECIFIC: Insert a paragraph here that explains how the instructor will hand out the datalab-handout.tar file to the students.

**现场说明：在此处插入一段说明教师如何将datalab-handout.tar文件分发给学生。**

Start by copying datalab-handout.tar to a (protected) directory on a Linux machine in which you plan to do your work. Then give the command

首先将datalab-handout.tar复制到计划在其中进行工作的Linux机器上的（受保护）目录。 然后发出命令

unix> tar xvf datalab-handout.tar.

This will cause a number of files to be unpacked in the directory. The only file you will be modifying and turning in is bits.c.

这将导致许多文件在目录中解压缩。 您将要修改并上交的唯一文件是bits.c。

The bits.c file contains a skeleton for each of the 13 programming puzzles. Your assignment is to complete each function skeleton using only *straightline* code for the integer puzzles (i.e., no loops or con- ditionals) and a limited number of C arithmetic and logical operators. Specifically, you are *only* allowed to use the following eight operators:

bits.c文件包含13个编程难题中每个难题的骨架。 您的任务是仅使用整数拼图的直线代码（即没有循环或条件）以及有限数量的C算术和逻辑运算符来完成每个函数骨架。 具体来说，只允许您使用以下八个运算符：

! ˜ & ˆ | + << >>

A few of the functions further restrict this list. Also, 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.

一些功能进一步限制了此列表。 另外，不允许使用任何大于8位的常量。 有关详细规则和所需编码风格的讨论，请参见bits.c中的注释。

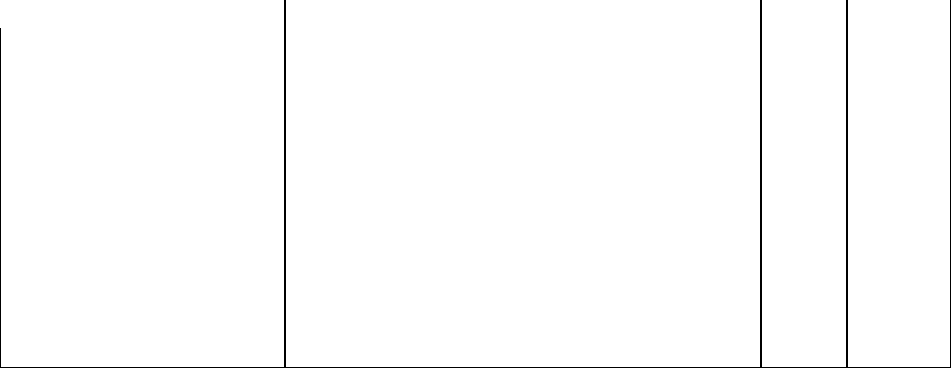
## The Puzzles

本节描述您将在bits.c中解决的难题。

表1按难度从最简单到最困难的顺序列出了难题。 “等级”（Rating）字段给出了难题的难度等级（点数），“最大操作数”（Max ops）字段给出了允许您使用以实现每个功能的最大运算符数。 有关功能期望行为的更多详细信息，请参见bits.c中的注释。 您也可以参考tests.c中的测试功能。 尽管它们不满足您函数的编码规则，但它们用作参考函数来表达函数的正确行为。

This section describes the puzzles that you will be solving in bits.c.

Table 1 lists the puzzles in rought order of difficulty from easiest to hardest. The “Rating” field gives the difficulty rating (the number of points) for the puzzle, and the “Max ops” field gives the maximum number of operators you are allowed to use to implement each function. See the comments in bits.c for more details on the desired behavior of the functions. You may also refer to the test functions in tests.c. These are used as reference functions to express the correct behavior of your functions, although they don’t satisfy the coding rules for your functions.



|  |  |  |
| --- | --- | --- |
| Name Description | Rating | Max ops |
| bitXor(x,y) x || y using only & and ˜. | 1 | 14 |
| tmin() Smallest two’s complement integer | 1 | 4 |
| isTmax(x) True only if x x is largest two’s comp. integer. | 1 | 10 |
| allOddBits(x) True only if all odd-numbered bits in x set to 1. | 2 | 12 |
| negate(x) Return -x with using - operator. | 2 | 5 |
| isAsciDigit(x) True if 0x30 ≤ x ≤. | 3 | 15 |
| conditional Same as x ? y : z | 3 | 16 |
| isLessOrEqual(x, y) True if x ≤ y, false otherwise | 3 | 24 |
| logicalNeg(x)) Compute !x without using ! operator. | 4 | 12 |
| howManyBits(x) Min. no. of bits to represent x in two’s comp. | 4 | 90 |
| floatScale2(uf) Return bit-level equiv. of 2\*f for f.p. arg. f. | 4 | 30 |
| floatFloat2Int(uf) Return bit-level equiv. of (int)f for f.p. arg. f. | 4 | 30 |
| floatPower2(x) Return bit-level equiv. of 2.0ˆx for integer x. | 4 | 30 |

Table 1: Datalab puzzles. For the floating point puzzles, value f is the floating-point number having the same bit representation as the unsigned integer uf.

Datalab难题。 对于浮点拼图，值f是与无符号整数uf具有相同位表示形式的浮点数。

For the floating-point puzzles, you will implement some common single-precision floating-point operations. For these puzzles, you are allowed to use standard control structures (conditionals, loops), and you may use both int and unsigned data types, including arbitrary unsigned and integer constants. You may not use any unions, structs, or arrays. Most significantly, you may not use any floating point data types, operations, or constants. Instead, any floating-point operand will be passed to the function as having type

unsigned, and any returned floating-point value will be of type unsigned. Your code should perform the bit manipulations that implement the specified floating point operations.

对于浮点拼图，您将实现一些常见的单精度浮点运算。 对于这些难题，可以使用标准控制结构（条件，循环），并且可以使用int和unsigned数据类型，包括任意的unsigned和integer常量。 您不得使用任何联合，结构或数组。 最重要的是，您不得使用任何浮点数据类型，操作或常量。 取而代之的是，任何浮点操作数都将以具有类型的形式传递给函数

The included program fshow helps you understand the structure of floating point numbers. To compile

fshow, switch to the handout directory and type:

附带的程序fshow可帮助您了解浮点数的结构。 编译

fshow，切换到讲义目录并键入：

unix> make

You can use fshow to see what an arbitrary pattern represents as a floating-point number:

您可以使用fshow查看任意模式代表的浮点数：

unix> ./fshow 2080374784

Floating point value 2.658455992e+36

Bit Representation 0x7c000000, sign = 0, exponent = f8, fraction = 000000 Normalized. 1.0000000000 X 2ˆ(121)

You can also give fshow hexadecimal and floating point values, and it will decipher their bit structure.

您还可以给fshow十六进制和浮点值，它将解密它们的位结构。

## Evaluation

Your score will be computed out of a maximum of 67 points based on the following distribution:

根据以下分布，您的分数最多可以得到67分：

**36** Correctness points.

**26** Performance points.

1. Style points.

36个正确点。

26性能点。

5样式点。

*Correctness points.* The puzzles you must solve have been given a difficulty rating between 1 and 4, such that their weighted sum totals to 36. We will evaluate your functions using the btest program, which is described in the next section. You will get full credit for a puzzle if it passes all of the tests performed by btest, and no credit otherwise.

正确点。 您必须解决的难题的难度等级在1到4之间，因此它们的加权总和为36。我们将使用btest程序来评估您的功能，这将在下一部分中进行描述。 如果它通过了btest进行的所有测试，您将获得一个拼图的满分，否则，将没有积分。

*Performance points.* Our main concern at this point in the course is that you can get the right answer. However, we want to instill in you a sense of keeping things as short and simple as you can. Furthermore, some of the puzzles can be solved by brute force, but we want you to be more clever. Thus, for each function we’ve established a maximum number of operators that you are allowed to use for each function. This limit is very generous and is designed only to catch egregiously inefficient solutions. You will receive two points for each correct function that satisfies the operator limit.

性能要点。 在此过程中，我们目前主要关心的是您可以获得正确的答案。 但是，我们希望向您灌输一种使事情尽可能短而简单的感觉。 此外，有些难题可以通过蛮力解决，但我们希望您更加聪明。 因此，对于每个功能，我们都建立了允许您为每个功能使用的最大数量的运算符。 此限制非常宽泛，仅用于捕获效率极低的解决方案。 对于满足操作员限制的每个正确功能，您将获得2分。

*Style points.* Finally, we’ve reserved 5 points for a subjective evaluation of the style of your solutions and your commenting. Your solutions should be as clean and straightforward as possible. Your comments should be informative, but they need not be extensive.

风格点。 最后，我们保留5分，以便您对解决方案的风格和您的评论进行主观评估。 您的解决方案应尽可能简洁明了。 您的评论应该是翔实的，但不必广泛。

## Autograding your work

We have included some autograding tools in the handout directory — btest, dlc, and driver.pl — to help you check the correctness of your work.

我们在讲义目录中包括了一些自动分级工具-btest，dlc和driver.pl-帮助您检查工作的正确性。

* + **btest:** This program checks the functional correctness of the functions in bits.c. To build and use it, type the following two commands:

•btest：该程序检查bits.c中功能的功能正确性。 要构建和使用它，请键入以下两个命令：

unix> make unix> ./btest

Notice that you must rebuild btest each time you modify your bits.c file.

You’ll find it helpful to work through the functions one at a time, testing each one as you go. You can use the -f flag to instruct btest to test only a single function:

注意，每次修改bits.c文件时，都必须重新构建btest。

您会发现一次完成一项功能，并在进行过程中对每项功能进行测试会有所帮助。 您可以使用-f标志来指示btest仅测试单个功能：

unix> ./btest -f bitXor

You can feed it specific function arguments using the option flags -1, -2, and -3: 您可以使用选项标志-1，-2和-3为其提供特定的函数参数：unix> ./btest -f bitXor -1 4 -2 5

Check the file README for documentation on running the btest program.

检查文件README，以获取有关运行btest程序的文档。

* + **dlc:** This is a modified version of an ANSI C compiler from the MIT CILK group that you can use to check for compliance with the coding rules for each puzzle. The typical usage is:
  + •dlc：这是MIT CILK组的ANSI C编译器的修改版本，可用于检查是否符合每个难题的编码规则。 典型用法是：

unix> ./dlc bits.c

The program runs silently unless it detects a problem, such as an illegal operator, too many operators, or non-straightline code in the integer puzzles. Running with the -e switch:

该程序将以静默方式运行，除非它检测到问题，例如非法运算符，过多的运算符或整数难题中的非直线代码。 使用-e开关运行：

unix> ./dlc -e bits.c

causes dlc to print counts of the number of operators used by each function. Type ./dlc -help

for a list of command line options.

使dlc打印每个函数使用的运算符数量的计数。 键入./dlc -help

有关命令行选项的列表。

* + **driver.pl:** This is a driver program that uses btest and dlc to compute the correctness and performance points for your solution. It takes no arguments:
  + •driver.pl：这是一个驱动程序，使用btest和dlc计算解决方案的正确性和性能。 它不带任何参数：

unix> ./driver.pl

Your instructors will use driver.pl to evaluate your solution.

您的讲师将使用driver.pl评估您的解决方案。

## Handin Instructions

### SITE-SPECIFIC: Insert text here that tells each student how to hand in their bits.c

**solution file at your school.**

**现场说明：在此处插入文字，告诉每个学生如何在学校上交他们的bits.c解决方案文件。**

1. **Advice**
   * Don’t include the <stdio.h> header file in your bits.c file, as it confuses dlc and results in some non-intuitive error messages. You will still be able to use printf in your bits.c file for debugging without including the <stdio.h> header, although gcc will print a warning that you can ignore.

•不要在您的bits.c文件中包含<stdio.h>头文件，因为它会混淆dlc并导致一些不直观的错误消息。 尽管gcc会打印警告，您可以忽略，但仍然可以在bits.c文件中使用printf进行调试，而无需包含<stdio.h>标头。

* + The dlc program enforces a stricter form of C declarations than is the case for C++ or that is enforced by gcc. In particular, any declaration must appear in a block (what you enclose in curly braces) before any statement that is not a declaration. For example, it will complain about the following code:

•dlc程序比C ++或gcc强制执行更严格的C声明形式。 特别是，任何声明必须出现在不是声明的任何语句之前的块中（用大括号括起来）。 例如，它将抱怨以下代码：

int foo(int x)

{

int a = x;

a \*= 3; /\* Statement that is not a declaration \*/ int b = a; /\* ERROR: Declaration not allowed here \*/

}

## The “Beat the Prof” Contest

For fun, we’re offering an optional “Beat the Prof” contest that allows you to compete with other students and the instructor to develop the most efficient puzzles. The goal is to solve each Data Lab puzzle using the fewest number of operators. Students who match or beat the instructor’s operator count for each puzzle are winners!

为了娱乐，我们提供了一个可选的“击败教授”竞赛，让您与其他学生和教练竞争，以开发出最有效的拼图。 目标是使用最少的运算符来解决每个Data Lab难题。 在每个谜题中达到或击败教练操作员人数的学生都是赢家！

To submit your entry to the contest, type:

要将您的参赛作品提交竞赛，请输入：

unix> ./driver.pl -u ‘‘Your Nickname’’

Nicknames are limited to 35 characters and can contain alphanumerics, apostrophes, commas, periods, dashes, underscores, and ampersands. You can submit as often as you like. Your most recent submission will appear on a real-time scoreboard, identified only by your nickname. You can view the scoreboard by pointing your browser at

昵称限制为35个字符，并且可以包含字母数字，撇号，逗号，句号，破折号，下划线和“＆”号。 您可以随意提交。 您最近提交的内容将显示在实时记分板上，仅由您的昵称标识。 您可以通过将浏览器指向来查看计分板

http://$SERVER\_NAME:$REQUESTD\_PORT

**SITE-SPECIFIC: Replace** $SERVER\_NAME **and** $REQUESTD\_PORT **with the values you set in the** ./contest/Contest.pm **file.**

**现场说明：将$ SERVER\_NAME和$ REQUESTD\_PORT替换为您在./contest/Contest.pm文件中设置的值。**