计算机系统实验一

- 1. 实验概述
  - 实验目的
    - 通过解决编程谜题的方式来熟悉位级别（bit-level）的数的表示
  - 实验内容
    - 使用C实现bits.c中的13个函数并取得较好的性能
- 2. 环境配置
  - VMWare 安装虚拟机
    - 打开机房电脑中的镜像
    - 密码是16
  - 网络文件夹下载课件
    - 49.52.5.75 何老师文件夹
    - 下载实验包tar和实验手册pdf
    - datalab-handout.tar
      - 实验教材
      - 放进Ubuntu再解压
    - datalab.pdf
      - 实验手册pdf
  - 自己电脑建议使用64位系统
- 3. 谜题介绍（参考pdf第五节和bits.c中的注释）
  - Bit Manipulations
    -

Name	Description	Rating	Max Ops
`bitXor(x,y)`	`x ^ y` using only `&` and `~`	1	14
`allOddBits(x)`	Are all odd-numbered bits set to 1?	2	12
`isAsciiDigit(x)`	Is `x` an ASCII digit?	3	15
`conditional(x, y, z)`	Same as C's "`x ? y : z`"	3	16
`logicalNeg(x)`	Compute `!x` without using `!` operator	4	12

Table 1: Bit-Level Manipulation Functions.

  - Two’ s Complement Arithmetic
    -

Name	Description	Rating	Max Ops
`tmin()`	Return smallest two's complement integer	1	4
`isTmax(x)`	Is `x` the largest 32-bit two's complement intger?	2	10
`negate(x)`	`-x` without negation	2	5
`isLessOrEqual(x,y)`	`x <= y`?	3	24
`howManyBits(x)`	Compute minimum number of bits required to represent `x`	4	90

Table 2: Arithmetic Functions

  - Floating-Point Operations
    -

Name	Description	Rating	Max Ops
`float_twice(uf)`	Compute `2.0*f`	4	30
`float_f2i(uf)`	Compute `(int) f`	4	30
`float_i2f(x)`	Compute `(float)x`	4	30

        - 使用 fshow帮助理解浮点数（参考pdf第五节）
- 4. 评分
    - 共63分
    - 37分正确分
        - 难度越高的谜题分越高
        - 通过dlc的检查和BDDcheck的动态测试可以得分
    - 26分性能分）
        - dlc检查操作数，每个谜题有2分性能分
- 5. 命令与工具介绍
    - 命令
        - 参考pdf第七节，或者输入help
    - make
        - 要先用make生成btest和几个show相关的小工具
    - btest
        - 介绍
            - 用少量测试验证程序正确性
            - 会输出使得程序出错的输入
        - 用途
            - 逐个调试函数正确性
        - 注意点
            - 每次修改bits.c重新make
        - 推荐
            - 在函数中添加printf并使用单个函数+输入参数的测试方式以查找代码中的错误
    - dlc
        - 用途
            - 检查代码是否符合规范（如有没有不允许的操作符、操作数是否过多）
            - 检查操作数
        - 注意
            - 如果没有问题是不会输出任何信息的
    - BDDcheck
        - 介绍
            - 用全面的测试用例验证函数是否正确
        - 用途
            - 可测试单个函数正确性
    - driver.pl

        - 用途
            - 整合dlc和BDDcheck的结果，便于查看完整结果
        - 注意
            - 该文件对第一个异或的测试评分可能存在问题
- 6. 建议
    - 推荐工作流
        - 详见pdf第九节
        - Step1 使用（btest+自定义输入+printf）来测试、调试函数正确性
            - linux> ./btest -f isLess -1 23 -2 0xabcd
        - Step2 使用（btest）来测试函数正确性
            - linux> ./btest -f isLess
        - Step3 使用（dlc）来检查函数代码规范
            - linux> ./dlc bits.c
        - Step4 在通过（btest）的测试后，使用BDDchecker来测试函数正确性
            - linux> ./bddcheck/check.pl -f isLess
        - Step5 对各个函数重复Steps1-4，在觉得有点累的时候使用（driver.pl）查看评分鼓励一下自己
            - linux> ./driver.pl
    - 其他建议（建议看pdf第九节，可以避免掉dlc，BDDcheck的一些坑）
        - dlc的使用建议
        - BDDcheck的使用建议
    - 建议先完成功能，再优化性能
- 7. 提交报告or课堂检查（暂未确定）