**深圳大学实验报告**

**课程名称： 计算机系统(2)**

**实验项目名称： 数据表示实验**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 冯禹洪**

**报告人：黄志鹏 学号：2017303008 班级： 计科一班**

**实验时间： 2020 年 4 月 6 日**

**实验报告提交时间： 2020 年 4 月 7 日**

**教务处制**

|  |
| --- |
| **实验目的与要求：**   1. 了解各种数据类型在计算机中的表示方法 2. 掌握C语言数据类型的位级表示及操作 |
| **方法、步骤：**   1. 安装gcc-multilib：     或者：    2、根据bits.c中的要求补全以下的函数：  intbitXor(int x, int y);  inttmin(void);  intisTmax(int x);  ntallOddBits(int x);  int negate(int x);  intisAsciiDigit(int x);  int conditional(int x, int y, int z);  intisLessOrEqual(int x, int y);  intlogicalNeg(int x);  inthowManyBits(int x);  unsignedfloat\_twice(unsigned uf);  unsigned float\_i2f(int x);  int float\_f2i(unsigned uf);  3、在Linux下测试以上函数是否正确，指令如下：  \*编译：./dlcbits.c  \*测试：makebtest  ./btest |
| **实验过程及内容：**   1. 安装make与gcc-multilib   在/etc/apt/sources.list中更新国内源，并执行命令apt-get update后即可安装make与gcc-multilib。     1. 补全bits.c中的函数   2.1 intbitXor(int x, int y)  x^y = (~x&y) | (x&~y) = ~(~(~x&y)&~(x&~y))。    2.2 inttmin(void)  最小的二进制补码为10…00 所以将1左移31位。    2.3 intisTmax(int x)  先判断除了符号位之外其他位是不是全为1，所以将该数左移一位再加上2，若满足条件则结果应为0，再判断符号为是不是0，将该数右移31位再判断剩下的数字是不是0。    2.4 ntallOddBits(int x)  先获得010101010…101，然后用x与其进行 | 操作，若x为所求，则会得到111111…11，最后按位取反再取否。    2.5 int negate(int x)  按位取反再加一即可。    2.6 intisAsciiDigit(int x)  判断x的ASCII码是否在‘0’和‘9’之间，将x分别减去48，57，若前者为符号位0，后者符号位为1，则得到目标数。    2.7 int conditional(int x, int y, int z)  实现x?y:z，即当x为非0数时返回y，当x为0时返回z，可推出公式(a&y)|(b&z)，当a为!x+~1+1,b为~(!x)+1,此时a和b满足上述条件。    2.8 intisLessOrEqual(int x, int y)  判断x与y是否异号，若x与y异号且x的符号位为1，则x一定小于y，若x与y同号，则使用y减去x，判断结果的符号位，若为0说明y大于等于x。    2.9 intlogicalNeg(int x)  判断~x+1与x的符号位，若都为0则x为0，若都为1，则x为10…0，其他非零情况为10或01。    2.10 int howManyBits(int x)  二分法，通过n16+n8+n4+n2+n1。通过位移运算符，来判断高位的n位是否为零，为零则nx为零，不为零则nx=相对应的数并将x向右移nx。以此类推。    2.11 unsignedfloat\_twice(unsigned uf)  若指数exp==0xff，返回本身；  若指数exp==0，则：  若表示小数部分的首位为0，则直接将小数部分左移一位；  若不为0，则将exp加一，再将小数部分左移一位；  其余情况直接将exp加一，若加一之后exp==0xff，则令小数部分为0，否则不做处理。    2.12 unsigned float\_i2f(int x)  int型（二进制表示）可假想小数点在最低有效位右边，左移shiftnum-1位，使得小数点移动到第一个有效数字（1）的右边，那么小数点右边首位到第23位就是浮点数表示的尾数部分了，因为这里只要取23位所以存在精度问题，这里采取四舍五入的方式，阶码部分看小数点移动的位数，加上偏置置（float中为127）即可。即(127+32-shiftnum)<<23。    2.13 int float\_f2i(unsigned uf)  获取uf的符号位、指数和小数部分，若指数大于158，则直接返回0x80000000，若指数小于127，则直接返回0，其余情况，将小数部分右移（23-指数）（因为小数部分是从23位开始的），再根据获取的符号位判断正负然后返回结果。     1. 在Linux下测试以上函数是否正确   编译后运行(./btest)。 |
| **实验结果：**    得分：37分。 |
| **心得体会：**  通过本次实验，学习了进行基本的位运算操作，进一步掌握了计算机系统的位运算，对于计算机系统中整型数字和浮点型数字有了更深的理解。 |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  2018年 月 日 |
| 备注： |