**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：**

**专 业：**

**指 导 教 师： 罗秋明**

**报告人： 学号： 班级：**

**实 验 时 间： 2017年3月31日**

**实验报告提交时间： 2017年4月13日**

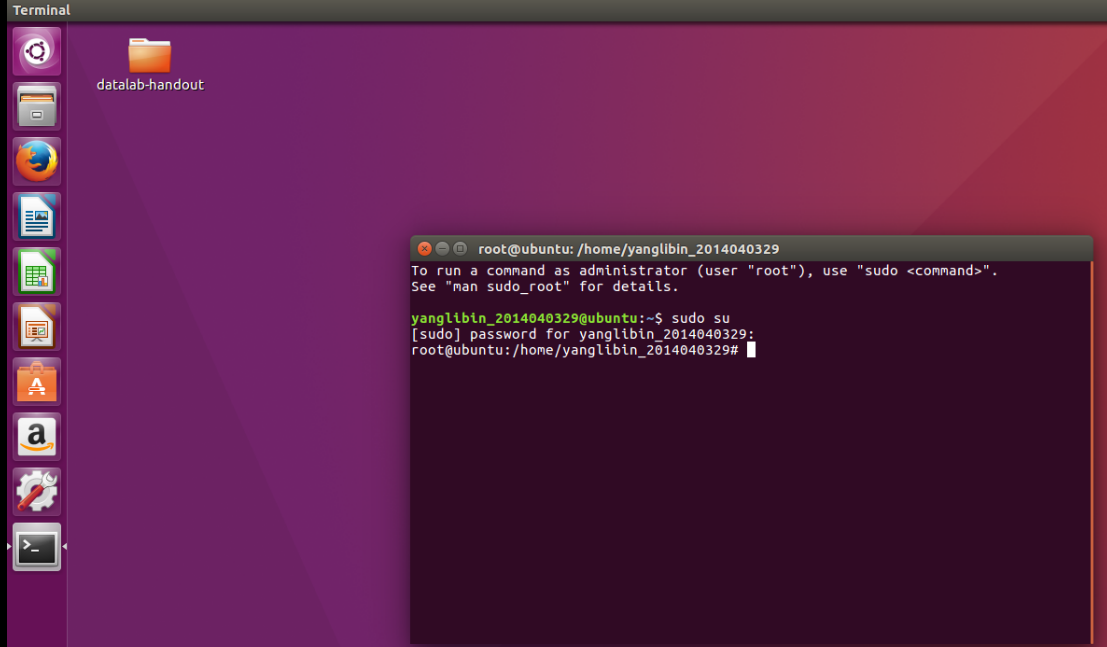
**教务处制**

**一、 实验目标：**

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

**二、实验环境：**

1. 计算机（Intel CPU）
2. Ubuntu Linux操作系统



**三、实验内容与步骤**

1、根据bits.c中的要求补全以下的函数：

int bitXor(int x, int y);

int tmin(void);

int isTmax(int x);

nt allOddBits(int x);

int negate(int x);

int isAsciiDigit(int x);

int conditional(int x, int y, int z);

int isLessOrEqual(int x, int y);

int logicalNeg(int x);

int howManyBits(int x);

unsigned float\_twice(unsigned uf);

unsigned float\_i2f(int x);

int float\_f2i(unsigned uf);

2、在Linux下测试以上函数是否正确，指令如下：

\*编译：./dlc bits.c

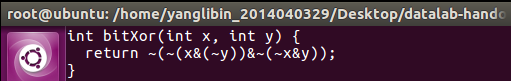
\*测试：make btest

./btest

**四、实验结果**

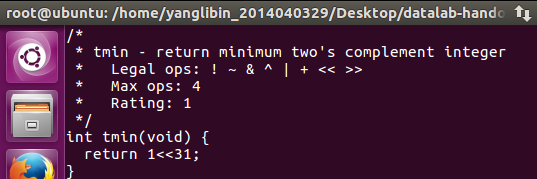
**1.** **int bitXor(int x, int y);**

由离散数学逻辑命题逻辑可得，异或如下：

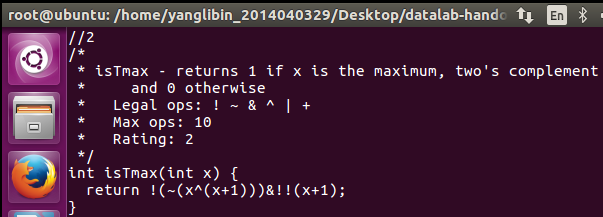


**2.** **int tmin(void);**

最小值为0x8000 0000，由1左移31即可得到：

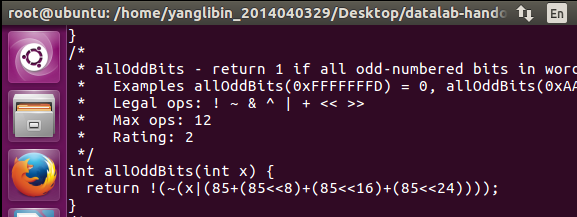


1. **int isTmax(int x);**

最大值为0x7fff ffff，加一会变为0x10000000，而此数加上本身后会变为0，本身加本身为0的数只有0和0x1000 0000，故而再将0xffffffff排除即可

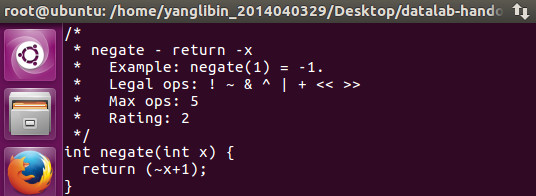
**4.** i**nt allOddBits(int x);**

只有所有奇数位为1的数（二进制），与0x5555 5555进行&运算才会得到0，故而需要得到0x5555 5555即可，将0x5分别左移4、8、16、24得到4个数，然后将这四个数和0x5相加即可得到0x5555 5555



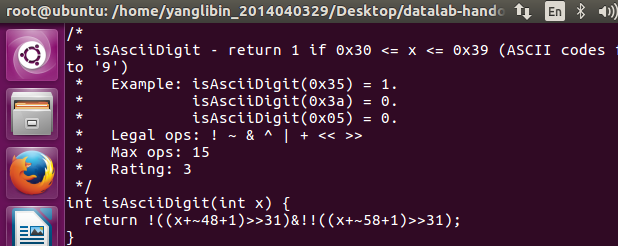
**5**.**int negate(int x);**

取反加1即可



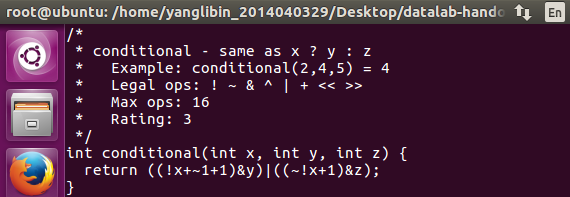
**6.** **int isAsciiDigit(int x);**

x需要>=’0’且<=’9’，将x与临界点作差，然后判断符号位的为0还是1即可



**7.** **int conditional(int x, int y, int z);**

首先使用t=!x，当x为0时返回1，当x不为0时，返回0，根据题意得到( \_ &y)|( \_ &z)，首先空1，当x不为0，即t=0时，需要t转换为0xffffffff（-1），当t=1时，需要将t转换为0x0（0），，将t-1即可，得到空1为“!x+~1+1”，同理空2为“~!x+1”



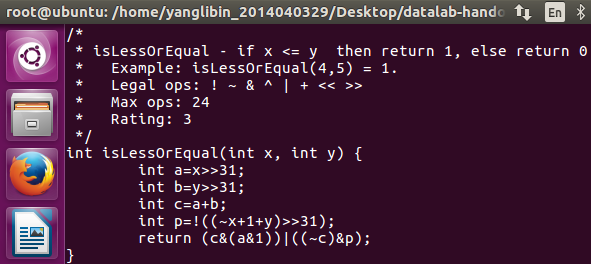
**8.** **int isLessOrEqual(int x, int y);**

直接用y-x可能会超出int的表示范围，故而：

A、在x与y同号的情况下转换为p=y-x>=0，然后p符号位(p>>31）&1为0则返回1，符号位1则返回0；

B、 异号时，只要x>=0，就要返回0，否则返回1，由(x>>31)&1能达到该效果；

C、 c=a+b可作为x,y同号异号的判断。



**9.** **int logicalNeg(int x);**

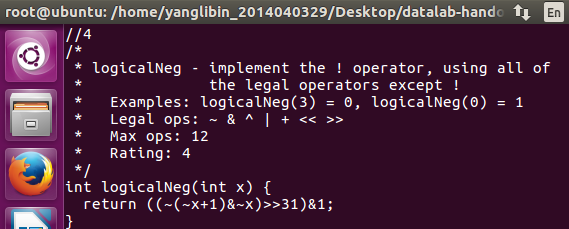
令y=~x+1，考虑x与y的符号位：

A. 当x为0时，两者符号位都为0；

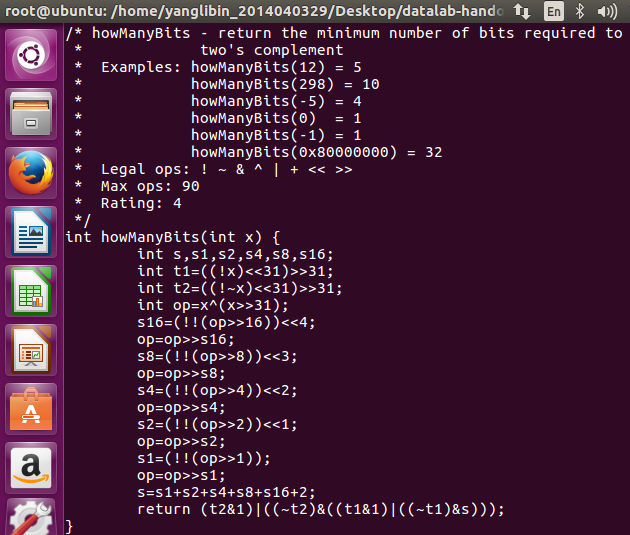
B. 当x=0x8000 0000时，两者符号位都为1；

C. 否则，两者符号位为01或10；

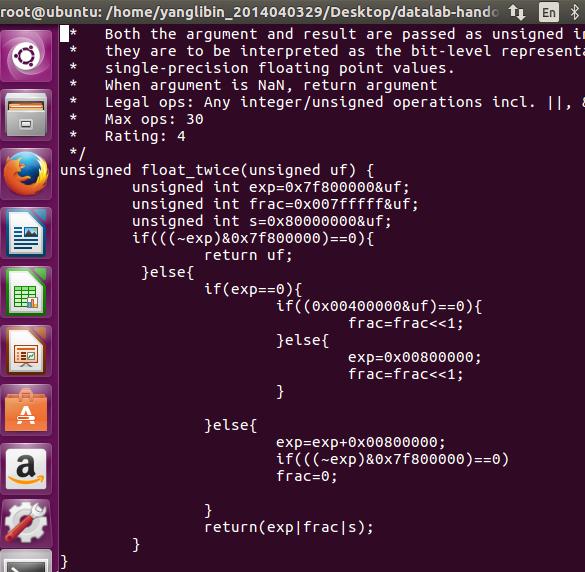
D. 根据离散数学的真值表得出(~x)&(~y).



**10.** **int howManyBits(int x);**



**11. unsigned float\_twice(unsigned uf);**



**12. unsigned float\_i2f(int x);**

A.用（二进制）科学计数法表示int型数时，尾数位数<=23，例如0x00008001，此时将0x8001左移24-16=8位得到frac，而exp则127+16-1；

B.当尾数位数>23时，找到位数最末一位记作x[i]，然后对尾数的舍去分3种情况考虑，初始化c=0：

a) 当x[i-1]=1且x[i-2]、x[i-3]…x[0]都为0（即要偶端舍入情况），且x[i]=1,令c=1（此处frac若是全为1，则会导致frac+c超出范围，这是bug,当还是通过了）；

b) 当x[i-1]=1且x[i-2]、x[i-3]…x[0]不都为0，令c=1（与a存在同样的bug）;

c) 除a、b的情况，c=0；

C.其他特殊情况再考虑一下就好了；



**13. int float\_f2i(unsigned uf);**

exp为0，x=(-1)^s\*0.frac\*2^(-126)；否则x=(-1)^s\*1.frac\*2^(exp-127)分情况考虑：

A.根据float转为int是向0舍入的情况，当exp=0或者exp<127(由exp-127<0得到)；

B.令exp\_sign=((exp>>23)-127)>=0，根据x=(-1)^s\*1.frac\*(exp-127)，

a) exp\_sign=0,x=1.x[22]x[21]…x[0]-->x=1;

b) exp\_sign=1,x=1x[22].x[21]…x[0]-->x=1x[22];

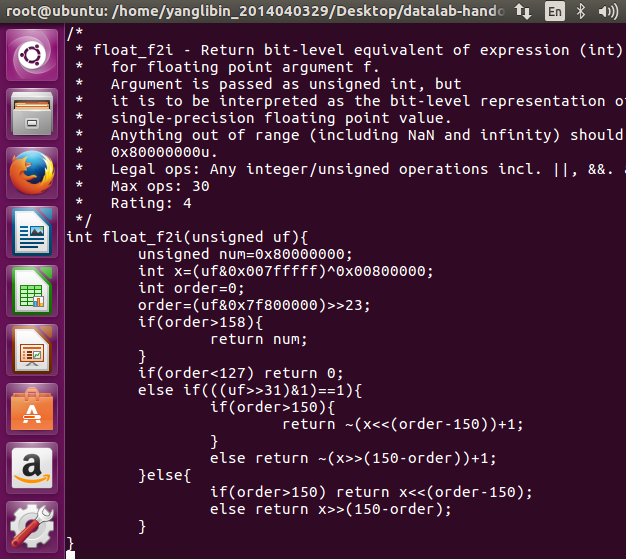
c) exp\_sign=2,x=1x[22]x[21].x[20]…x[0]-->x=1x[22]x[21];

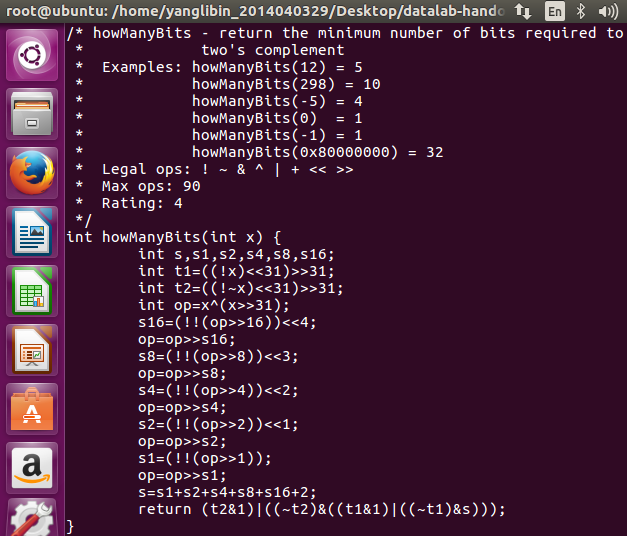
d) exp\_sign=23,x=1x[22]x[21]x[20]…x[0]-->x=1x[22]x[21]…x[0]

e) exp\_sign=30,x=1x[22]x[21]x[20]…x[0]-->x=1x[22]x[21]…x[0]0…0(共31位)

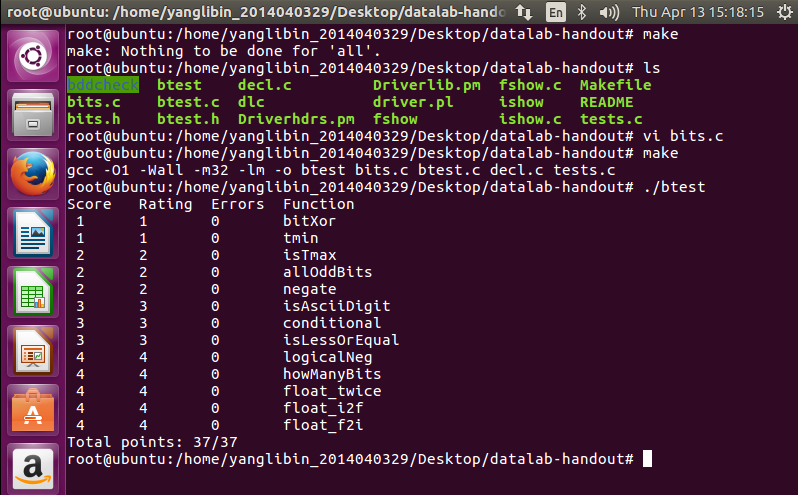
f) exp\_sign=32,x=1x[22]x[21]x[20]…x[0]-->x=1x[22]x[21]…x[0]0…0(共32位)，此时，因为int为有符号，只有0x0000 00000能被表示出来，其他都是NAN;

C.当exp\_sign=((exp>>23)-127)>=33,NAN





**最终截图：**



**五、实验总结与体会**

此次实验主要考查的是对数据的处理，对此需要掌握数据在机器中的表示，运用合理的位运算来实现相应的功能。

通过此次实验，我更好地熟悉和掌握计算机中整数和浮点数的二进制编码表示。尤其是加深对数据二进制编码表示的了解。通过使用有限类型和数量的运算操作实现一组给定功能的函数来提高了我对数据表示方面的C语言编程能力。

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  指导教师签字： 罗秋明    2017年 4月10日 |
| 备注： |