**答案**

**思考题：**

1. 至少需要8个处理器

1/(0.02+0.98/n) > 7 hence n > 7;

1. M = 3；N = 16
2. A：false，x = -2147483648

B: false, x = 46342, x\*x = -2147386332

C: True， 因为对于int，x前面加负号就是把符号位换一下，要么大于0，要么小于等于0

D: false, x = -2147483648, -x = -2147483648

E: true, 因为这是在位级层面比较，位级层面x+y与ux+uy完全相同

F：true, 因为~y = -y-1,所以原式：-x\*y-x+uy\*ux == -x,由于这个运算也是位级层面，故而由E知道等式成立

G：true，因为在位级层面乘2的幂次方就代表左移多少位，即使会有溢出。

H：true, 如果符号位为0，则移位后值不变，如果符号位为1，则移位后值变小

4.1)Y 因为添加符号只是将符号位相反，并不影响大小的比较

N if x = min and y = 0

2)Y 左移四位相当于乘16，故((x+y)<<4)+y-x=16x+16y+y-x=17y+15x

3)Y ~x=-x-1;~y=y-1;因此~y+~x+1=-x-y-1=~(x+y)

4)Y –(y-x)=x-y;而ux,uy在位级层面上和x,y相等

5)N 当x的符号位为1时，x=ux

6)Y 左移时如果末尾为1，那么右移回来末尾为0，x变小，如果末尾为0，那么x的值不变

7)N float在表示很大的数时不准确，例如x=2147483647

8)N 因为x+y可能会溢出,但是double表示的数范围比int大因此dx+dy不会溢出；例如：x=2147483647,y=2147483647;

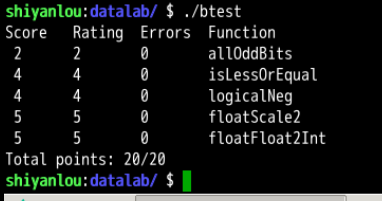
9)Y double可以精确表示int，故交换顺序不影响结果

10)Y double可以精确表示int，故交换顺序不影响结果

N 32\*32 = 64, but double\_max = 52;

**实践题：**

运行结果如下



五个函数的源代码如下：

int allOddBits(int x) { //get

int y = 0xAA + (0xAA << 8);

y = y + (y << 16);

return !((y & x) ^ y);

}

// x <= y

int isLessOrEqual(int x, int y) {

int s = y + (~x + 1); //y-x

int m = 1 << 31; //10000...000

int sx = !!(x & m); //>0 1

int sy = !!(y & m); //<0 0

int c = !!(sx ^ sy); //异号为1

int f = s & m;

return (c & !sy) | (!c & !f);

}

int logicalNeg(int x) {

return (((x | (~x + 1)) >> 31) + 1);

}

unsigned floatScale2(unsigned uf) { //get

int exp = (uf >> 23) & 0xff;

if (exp == 255) return uf; //infinity and NaN

int sign = uf & (1 << 31); //10000...00 or 0000...00

if (exp == 0) return uf << 1 | sign;

exp++;

if (exp == 255) return 0x7f800000 | sign;

return (uf & 0x807fffff) | (exp << 23);

}

int floatFloat2Int(unsigned uf) { //get

int exp = (uf >> 23) & 0xff;

int sign = uf & (1 << 31);

if (exp == 255) return 0x80000000;

if (exp == 0) return 0;

int k = exp - 127;

int f = uf & 0x00ffffff;

if (k < 0) return 0;

if (k > 31) return 0x80000000;

if (k < 23)

f >>= (23 - k);

else

f <<= (k - 23);

if (sign)

return ~f + 1;

else

return f;

}