Homework4

李俊霖

2021/11/09

题目1

- 写一个C表达式，在下列描述的条件下产生1，其他情况产生0，假设X是int类型。代码中不能使用==或！=进行测试。
  - x的任何位都等于1;
  - x的任何位都等于0;
  - x的最低有效字节中的位都等于1;
  - x的最高有效字节中的位都等于1;

Problem 1

- A. `!(~x)`
- B. `!x`
- C. `!~(x | 0x00ffffff)`
- D. `!(x & 0x000000ff)`

题目2

- int为32位，float和double分别是32位和64位IEEE格式
  - Int x =random();
  - Int y = random();
  - Int z = random();
  - Double dx = (double)x;
  - Double dy = (double)y;
  - Double dz = (double)z;
- 对于下面的每个C表达式，判断是否恒为1。如果是请说明原理，如果不是请举出反例。
  - A. (float)x == (float)dx
  - B. dx-dy == (double)(x-y)
  - C. (dx+dy)+dz == dx+(dy+dz)
  - D. (dx*dy)*dz == dx*(dy*dz)
  - E. dx/dx == dz/dz

Problem 2

- A. (float)x == (float)dx

  不成立。比如x为int的最大值时，超过float的精度。
- B. dx-dy == (double)(x-y)

  不成立。当x-y发生溢出时等式不成立，如x=Tmin，y=1，x-y发生下溢。
- C. (dx+dy)+dz == dx+(dy+dz)

  成立。dx,dy,dz是由整型x,y,z转换而来的，相加不会超过double的精度，因此不会产生舍入的误差，满足加法结合律。
- D. (dx * dy) * dz == dx * (dy * dz)

  不成立。double无法精确表示2^{64}以内所有数，三个数相乘可能会产生舍入，因此不满足乘法结合律。例如取x=0xe87d0982,y=0xd2027960,z=0xd119203f，可以验证等式不成立。
- E. dx/dx == dz/dz

  不成立。dx或dz为零时不成立。

题目3

- 编写如下函数，求浮点数f的绝对值|f|。如果f是NaN，那么应该直接返回f（注意NaN不要对f做任何修改）。
- 其中float_bits等价于unsigned，是float数字的二进制形式
  - typedef unsigned float_bits;
- /* Compute |f|. If f is NaN, then return f. */
- float_bits float_absval (float_bits f);

Problem 3

```
float_bits float_absval(float_bits f)
{
    unsigned abs = f&(0x7FFFFFFF);//符号位取0
    if (abs>(0x7F800000))  return f;//判断NAN
    return abs;//否则输出绝对值
}
```

题目4

- 实现如下函数，对于浮点数f，计算2.0*f。如果f是NaN，你的函数应该简单返回f。
- /* Compute 2*f. If f is NaN, return f. */
- float_bits float_twice(float_bits f);

Problem 4

基本思路

- 若浮点数为规格化数的时候，只需要阶码 + 1。
  - 特殊情况：当e的为11111110 ，此时不可以直接+1，它们*2会溢出，因此要提升为无穷。
- 若浮点数为非规格化数，×2就相当于把后面的小数左移一位。

```
float_bits float_twice(float_bits f) {
    unsigned s = f >> 31;//符号位
    unsigned exp = f >> 23 & 0xFF;//阶码
    unsigned frac = f & 0x7FFFFF;//尾数

    int is_NAN_or_infinity = (exp == 0xFF);//判断是否是无穷或者NAN，即阶码为11111111
    if (is_NAN_or_infinity)  return f;

    //非规格化数
    if (exp == 0) frac <<= 1;
    //最大的特殊规格化数，*2溢出，要提升为无穷
    else if (exp == 0xFE){
        exp = 0xFF;
        frac = 0;
    }
    //规格化数
    else exp += 1;

    return sig << 31 | exp << 23 | frac;
}
```