计组第二章家庭作业

软件1401-宗玉芬

201426010124

2.58

Int is\_little\_endain()

{

int x=1;

return \*(char \*) &x;

}

解析：题目要求小端法返回1大端法返回0.那我们就定义一个整数为1。

0x100 0x101 0x102 0x103 0x100 0x101 0x102 0x103

01 00 00 00 00 00 00 01

小端法 大端法

所以，我们只要取出整数1的首地址中的值即可，即\*(char \*) &x。，\*(char \*)&a取a所在地址的第0个字节的值作为一个char单字节返回

2.71

A:它错在得到的结果虽然是无符号的，但并非是由原数扩展而来的无符号数。例如数为0x10101081抽取字节为0时扩展出的无符号应该是11111111111111111111111110000001，而原代码得到的数为0000000000000000000000000010000001

B：int xbyte(unsigned word,int bytenum){

int temp=word<<((3-bytenum)<<3);

return temp>>24;

}

解析：此函数是将一个无符号的word抽取出指定字节并扩展为32位int。首先将指定字节数左移到最高字节，再右移到最低字节。

2.84

A：浮点数5.0等于二进制数101.0.即1.01×2^2.根据浮点数表示法V=(-1)s×M×2E知尾数M=1.01，小数f=0.01=0.25，E=2。所以指数部分e=E+Bias=2+(2^(k-1)-1)= 2^(k-1)+1

所以位表示的三个部分s-e-f: 0-10….01-010000..0

B:能够被准确描述的最大奇整数，那么其M=1.111..1，故f部分全为1，E应该为n。当然，这个假设在2^(k-1)>=n的情况下才能成立。这时，s=0,e=n+2^(k-1)-1,f=11...1

C：最小的正规格化数为0-0….1- 00..0则e=1,f=0,M=f+1=1,则E=1-Bias.所以V=2^(1-bias)即2^(2-2^(k-1))，所以其倒数值V为2^(2^(k-1)-2)，所以M为1.00000，f部分为全0，E=2^(k-1)-2，e=2^(k-1)-2 + bias = 2^k - 3，即为11..101。位表示为0-11..101-00..0

2.88

A:false.因为float只能表示24位，当X为最大值时，float无法准确表示，会有信息丢失。而double是可以准确表示32位数的，所以两边不会总是相等

B：false。右边可能会发生溢出而左边不会导致结果可能不相等。例如当x 和y都为最大值的时候右边溢出左右值不相等

C：true 因为double可以精确表示所有正负2^53以内的所有整数。所以三个数相加可以精确表示。

D：false 两个2^32以内的数相乘可能大于2^53，最大为2^64，然而double只能精确表示2^53以内的数，如此一来可能发生错误导致两边的数可能不相等

E：false，0/0.0为NaN，(非0)/0.0为正负inf。同号inf相减为NaN，异号inf相减也为被减数的inf

2.92

float\_bits float\_negate(float\_bits f) {

if((((f>>23)&0xff) ^0xff)||(!(f&((1<<23)-1))))

f^=(1<<31);

return f;

}

解析：题目要求对于浮点数，如果f是NaN则返回f，否则返回-f.首先明白对于32位的单精度浮点数，exp占8位，frac部分占23位.对于NaN数，exp部分全为1，frac部分不等于0.

if((((f>>23)&0xff) ^0xff)||(!(f&((1<<23)-1))))是f不为NaN时为真。其中((f>>23)&0xff) ^0xff是先将f右移23位得到frac部分，经过与操作和异或操作判断是否否全为1；(!(f&((1<<23)-1)))是将1左移23位得到10…0,-1得到0111..1,与操作f判断frac部分。f^=(1<<31)是得到-f