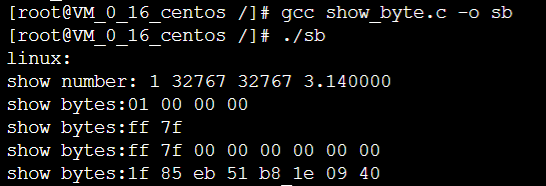
# 2.55



linux 64位采用小端存储，所以先输出低位，然后再输出高位。

# 2.59

#include<stdio.h>

int main()

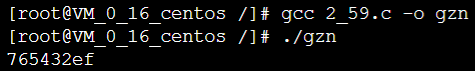
{

int x=0x89ABCDEF,y=0x76543210;

printf("%02x\n",(x&0x000000FF)|(y&0xFFFFFF00));

return 0;

}



# 2.63

#include<stdio.h>

unsigned srl(unsigned x,int k)

{

unsigned xsra = (int) x >> k;

return (xsra&(~(-1<<(8\*sizeof(int)-k))));

}

int sra(int x,int k)

{

int xsrl = (unsigned) x >> k;

int flag=(xsrl&(1<<(8\*sizeof(int)-k-1)));

if (flag) return (xsrl|(-1<<(8\*sizeof(int)-k))); else return xsrl;

}

int main()

{

int x1,k;

for(;;)

{

scanf("%x %d",&x1,&k);

unsigned x2;

x2=(unsigned)x1;

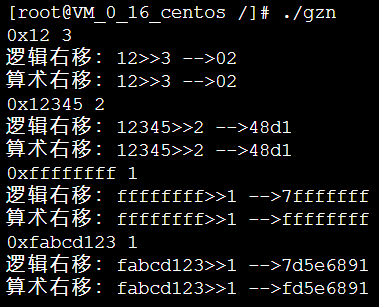
printf("閫昏緫鍙崇Щ锛?02x>>%d ->%02x\n",x2,k,srl(x2,k));

printf("绠楁湳鍙崇Щ锛?02x>>%d ->%02x\n",x1,k,sra(x1,k));

}

return 0;

}



# 2.67

A：错误：1<<32

B：只要排除大于32位情况就可以，既然不能移动32那么移动31次就好了，我们把1自己先左移一位变成2不就好了，改成2<<31即可这种方法返回结果是正确但是同样会警告忽略即可。

C：加一个判断16位情况，代码如下：

int bad\_int\_size\_is\_32()

{

int bf\_msb = 0x2 << 15;

int st\_msb = bf\_msb << 15;

int by\_msb = st\_msb<<1;

return bf\_msb && st\_msb && !by\_msb;

}

# 2.71

#include <stdio.h>

#include <stdlib.h>

int Merge(unsigned a,int b)

{

int x=a<<((3-b)<<3);

b=sizeof(int)<<3;

return x>>b;

}

int main()

{

unsigned a;

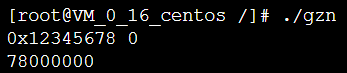
int b;

scanf("%x %d",&a,&b);

printf("%x\n",Merge(a,b));

return 0;

}



将要提取的字节移到最高位，再用有符号数的算术右移来实现，无符号转有符号。

# 2.75

#include <stdio.h>

#include <stdlib.h>

int add\_out(unsigned x, unsigned y){return x + y >= x;}

unsigned uhp(unsigned x, unsigned y)

{

int w = sizeof(int)<<3, i;

unsigned high = 0;

unsigned sum = (y&0x1)?x:0;

for(i=1; i<w; i++)

{

if( (y>>i) & 0x1 )

{

high += x>>(w-i);

if(!add\_out(sum, x<<i)) high++;

sum += (x<<i);

}

}

return high;

}

int main()

{

unsigned a,b;

scanf("%x%x",&a,&b);

printf("%x\n",uhp(a,b));

return 0;

}

# 2.79

#include <stdio.h>

#include <stdlib.h>

int mul3div4(int x)

{

return (x>>2)+(x>>1)+((((x<<1)&0x3)+(x&0x3))>>2);

}

int main()

{

int x;

scanf("%d",&x);

printf("%d\n",mul3div4(x));

return 0;

}

求3\*x/4;原式子可化为(x<<1+x)>>2,可以转换为x>>2+x>>1

# 2.83

A： 1/3中Y=5，k=4，1/3=5/(2^4-1)，y=Y/(2^k-1)

B： （1）101，Y=5，k=3，y=5/7；

（2）0110，Y=6，k=4，y=2/5；

（3）010011，Y=19，k=6，y=19/31；

# 2.87

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **描述** | **hex** | **M** | **E** | **V** | **D** |
| -0 | 0x8000 | 0 | -15 |  |  |
| 最小的>2的值 | 0x4001 | 1025/1024 | 1 | 1025\*2^(-9) | 2.00195315 |
| 512 | 0x6000 | 1 | 9 |  |  |
| 最大非规格数 | 0x03ff | 1023/1024 | -14 | 1023\*2^(-24) | 0.000060975516 |
| 负无穷大 | 0xfc00 |  |  |  |  |
| 0x3BB0 | 0x7376 | 955/512 | 13 | 955\*2^(4) | 15280 |

# 2.91

A 11.0010 0100 0011 1111 0110 11

B 根据提示对照公式x=Y/(2^k-1),得到k=3,Y=001，22/7=3+1，故表示为11.001001001...

C 第9位

# 2.95

float\_bits float\_half(float\_bits f)

{

unsigned sign=f>>31;

unsigned exp=(f>>23) & 0xff;

unsigned frac=f&0x7fffff;

if( exp == 0xff )

return f;

else if(exp==0)//非规格

return (sign<<31) | ((frac>>1)+(1 & (frac&3==3)));

else if(exp==1)//规格

return (sign<<31) | ((exp-1)<<23) | ( (frac>>1)&0x400000);

else

return (sign<<31) | ((exp-1)<<23) | frac;

}