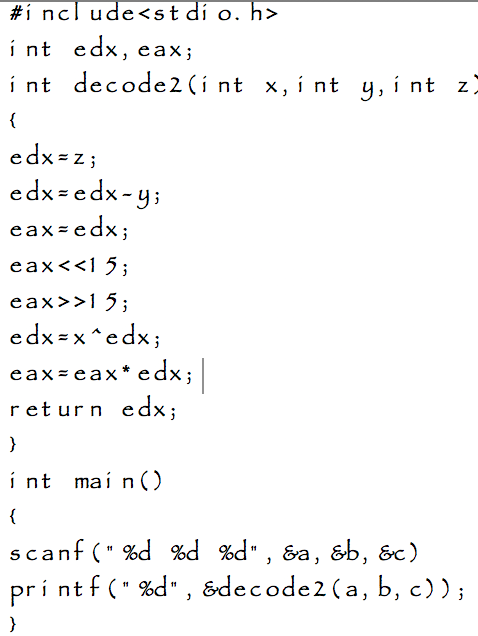
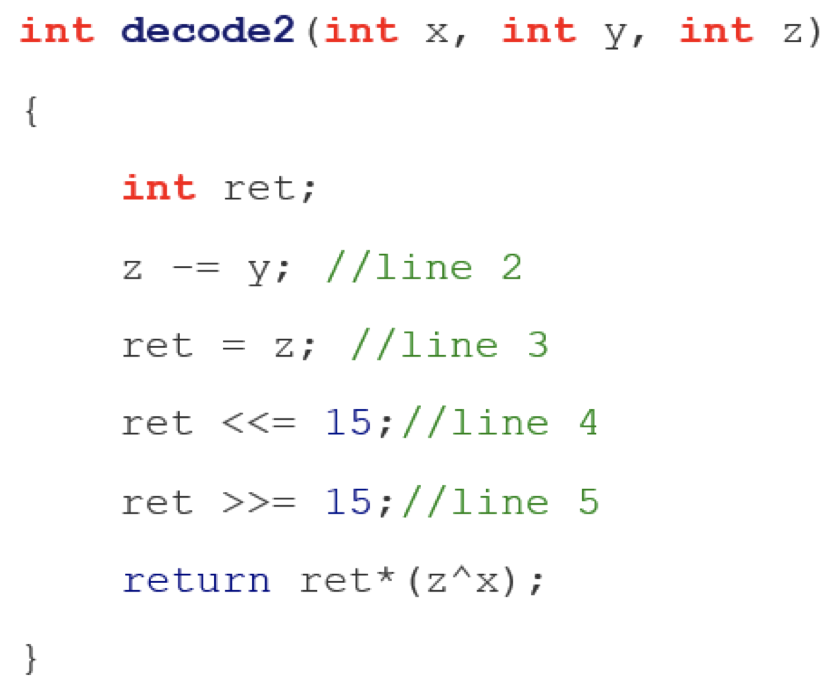


作答：

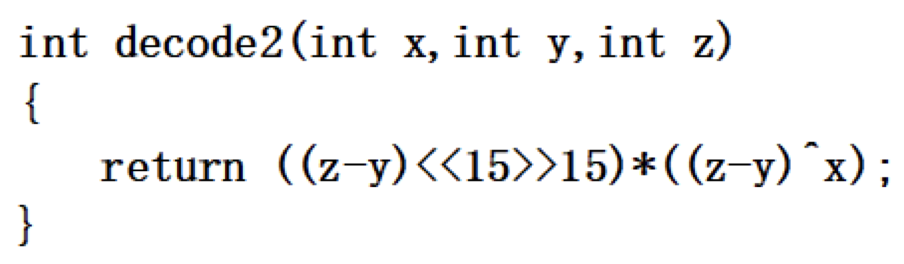


不过，虽然这样写是完全把汇编代码搬运到c程序里面，可是如果要让程序反汇编是上面汇编代码这样的话，就不能这么写。

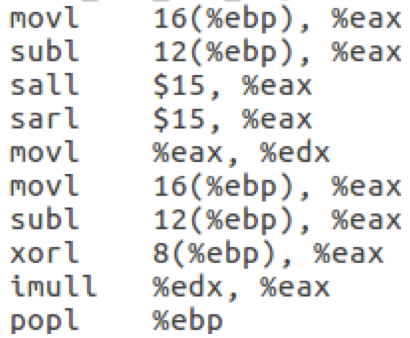


这么写直接反汇编也和上面的代码不是一样的，而只有在-O1优化之后才是一样的

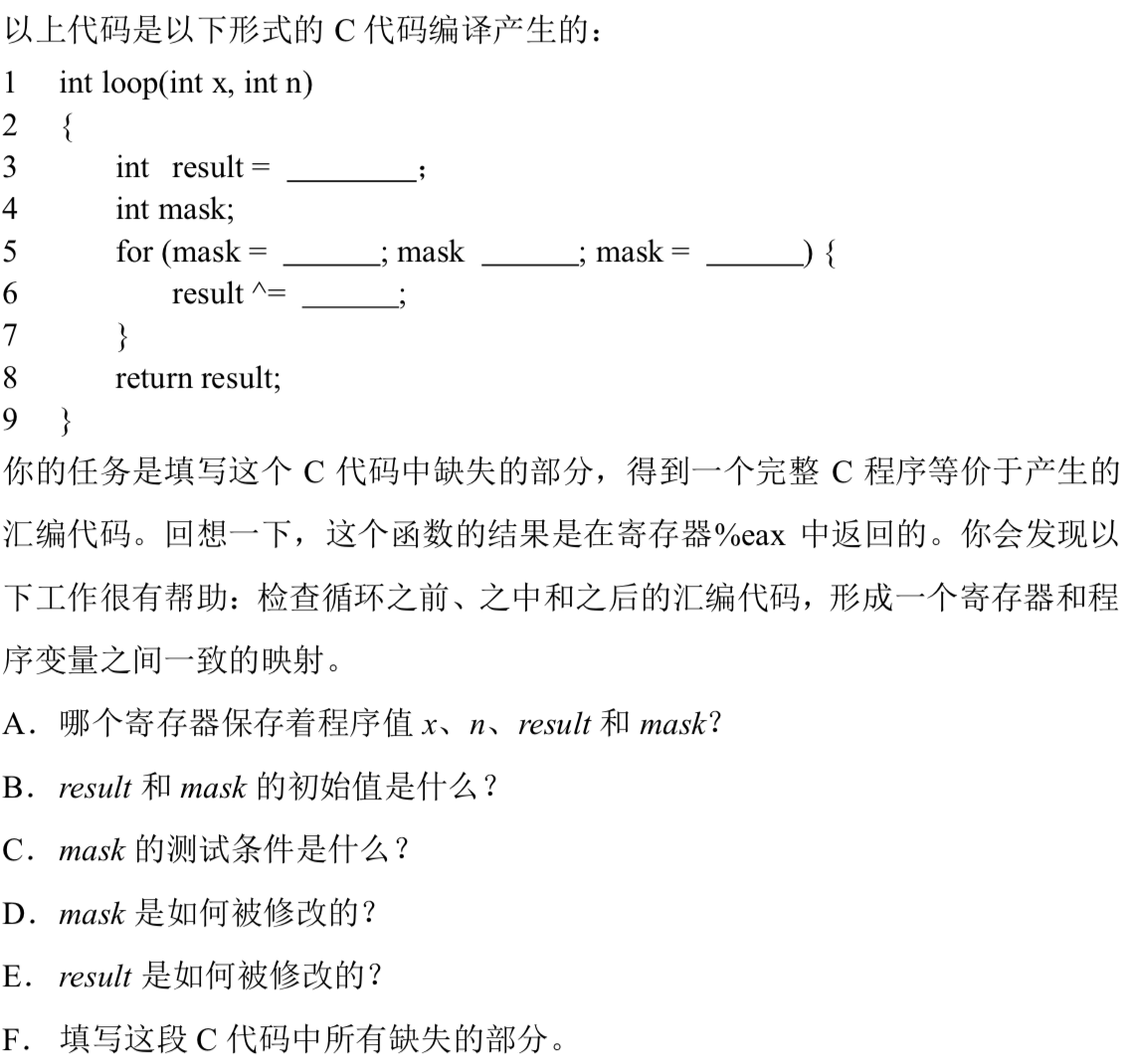
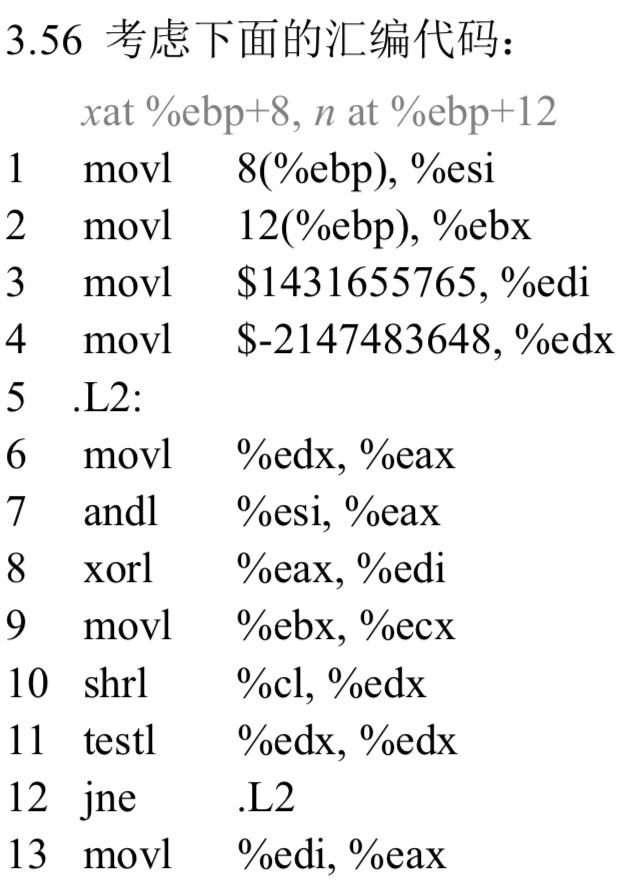
所以可能是定义变量的原因，要是把所有的运算都直接写在返回值上面



反汇编：



所以我推测O1减少存取内存的次数，节省空间



作答：

int loop(int x,int n)

{

int result = 1431655765;

int mask;

for(mask = -2147483648;mask！= 0 ;mask = mask>>n）

{

result^ = mask & x;

}

return result;

}

%esi x

%ebx n

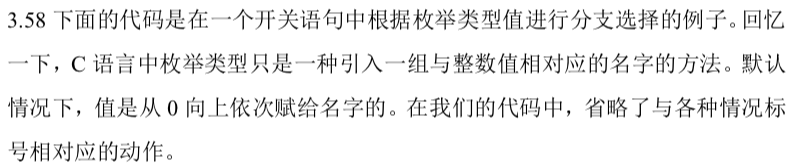
%edi result=1431655765

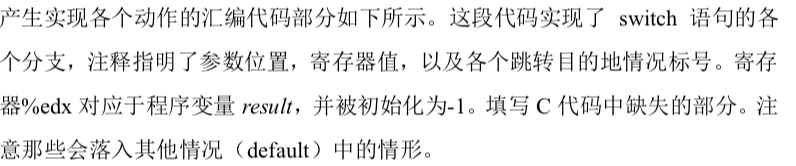
%edx mask=-2147483648

mask测试条件：testl %edx,%edx

mask修改：mask = mask>>n

result修改：result^ = mask & x





作答：

typedef enum{MODE\_A,MODE\_B,MODE\_C,MODE\_D,MODE\_E} mode\_t;

int switch3(int \*p1,int \*p2,mode\_t action)

{

int result =0;

switch(action)

{

case MODE\_A:

result = \*p1;

\*p1 = \*p2;

break;

case MODE\_B:

result = \*p1+\*p2;

\*p2 = result;

break;

case MODE\_C:

result = \*p1；

\*p2 =15;

break;

case MODE\_D:

\*p2 = \*p1;

case MODE\_E:

result = 17;

break;

default:

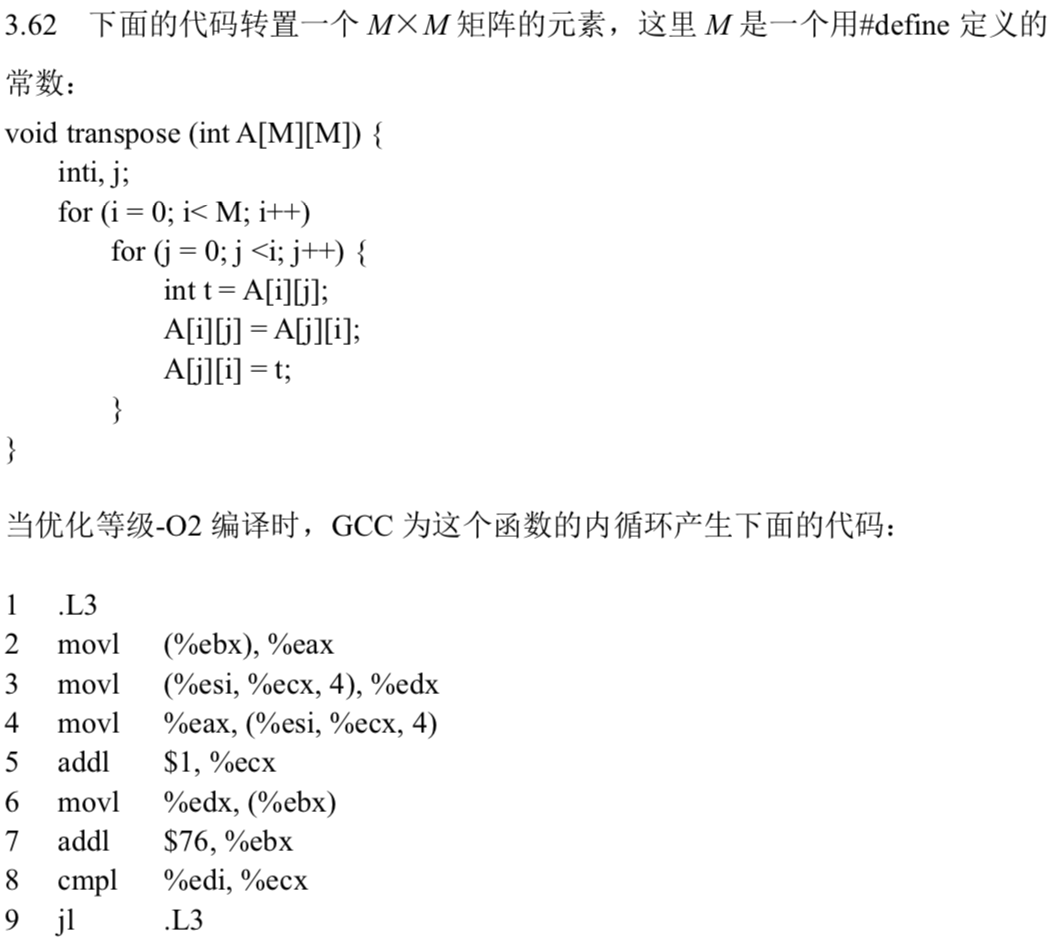
result = -1;

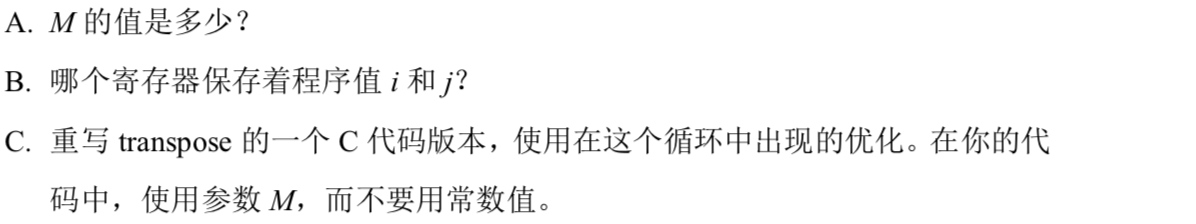
break;

}

return result;

}





作答：

i在edi里面，j在ecx里面

M的值是19

void transpose(int M,int A[M][M])

{

int i,j;

for(int i=0;i<M;i++)

{

int \*a=&A[i][0];

int \*ebx=&A[0][i];

for(j=0;j<I;j++)

{

int t=\*a; //a是那个

**a=e**bx;

\*ebx=t;

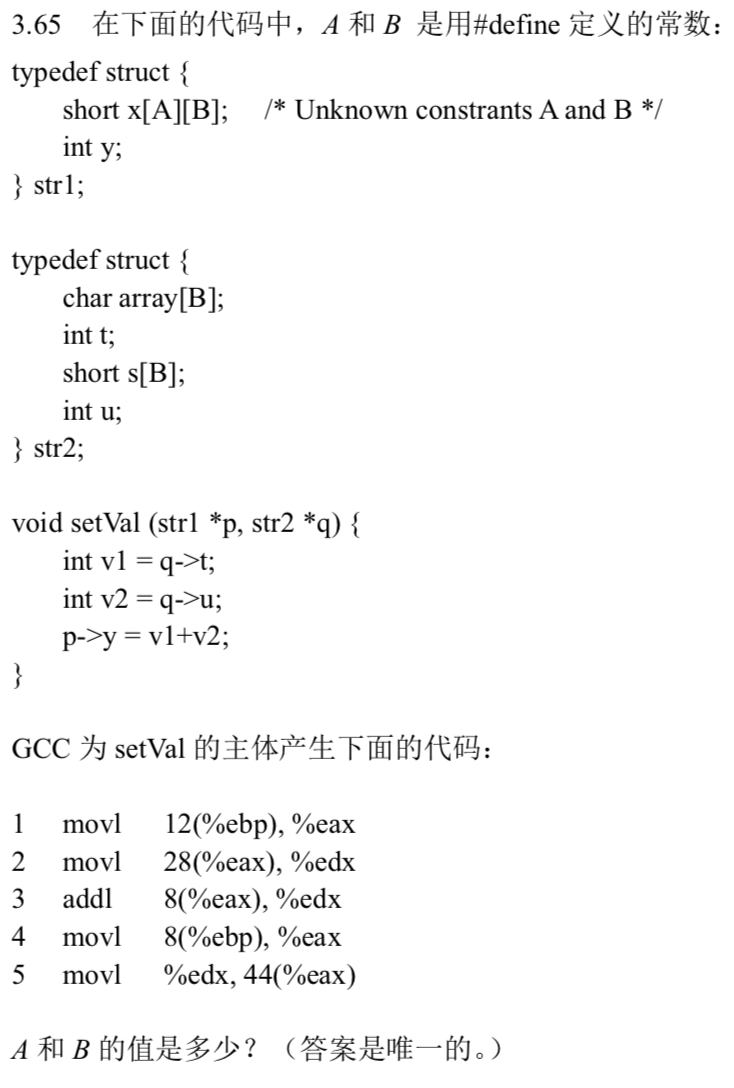
a++;

ebx=ebx+M;

}

}

}



作答：

想了半天，原来这题不是注重于分析汇编代码的逻辑，而是分析这个28、8之间的关系！！！

以前都没咋注意。

首先p->y是v1+v2，对应的就是，然后看t和u，t是8+eax，而u是28+eax，那就说明，他们的差减去t那个int的长度4再除以short的2就差不多是B的数值。那B就是8或7。（栈是四个字节为单位的）

那看A

44/2=21

21=A\*B

那B只有等于7的时候才有解

A=3，B=7