3.34

A.x

B.代码补充如下：

```
int rfun(unsigned x) {
    if (x==0)
        return 0;
    unsigned nx = x>>1,
    int rv = rfun(nx);
    return (x&0x1)+rv;
}
```

C.计算x中各位的和。它递归的计算了除了最低位之外的所有其他位的和，然后加上最低位得到结果

3.56

A.x、n、result和mask分别保存在 %esi %ebx %edi %edx中

B.result的初始值为1431655765 十六进制补码形式为0x55555555
  mask的初始值为-2147483648 十六进制补码形式为0x80000000

C.mask是否等于0 即：mask != 0

D.mask被通过逻辑逻辑右移n位而被修改，因为在调用n时只调用了寄存器的低八位因此这个n不会是一个很大的数，又因为声明mask时是int变量，而要想逻辑右移，先将其强制转化成unsigned类型，再进行右移n位。
即： mask = ((unsigned) mask) >> n

E.result是通过与x和mask进行位与的结果异或而来
即result^=(x&mask)

F.代码补充如下：

```
int loop(int x,int n)
{
    int result=0x55555555;
    int mask;
    for(mask=0x80000000; mask! =0; mask=((unsigned) mask) >> n){
        result^=(x&mask);
    }
    return result;
}
```

3.59

```
int switch_prob(int x, int n)
{
    int result= x;
    switch(n) {
    /* Fill in code here */
        case 40:
            result<<=3;
```

```
            break;
        case 42:
            result<<=3;
            break;
        case 43:
            result>>=3;
            break;
        case 44:
            result=(result<<3-result)*(result<<3-result)+17;
            break;
        case 45:
            result=result*result+17;
            break;
        default:
            result+=17;
    }
    return result;
}
//%eax中保存的即为result
```

3.66

```
A.7

B.结构a_struct的完整声明如下
typedef struct{
    int idx;
    int x[6];
}a_struct;

//以下为草稿
1 00000000 <test>:
2 0: 55 push %ebp
3 1: 89 e5 mov %esp,%ebp
4 3: 53 push %ebx
5 4: Sb 46 08 mov Ox8(%ebp),%eax  //eax=i
6 7: Sb 4d Oc mov Oxc(%ebp),%ecx //ecx=&bp
7 a: 6b dB 1c imul $Ox1c,%eax,%ebx //ebx=28*i
8 d: 8d 14 c5 00 00 00 00 lea OxO(,%eax,8) ,%edx //edx=8*i
9 14: 29 c2 sub %eax, edx//edx=7*i
10 16: 03 54 19 04 add Ox4(%ecx,%ebx,1),%edx
//edx=add(ecx+ebx+4)+7*i=add(&bp+28*i+4)+7*i
11 1a: Sb 81 ca oo oo oo mov Oxc8(%ecx),%eax //eax=add(&bp+200)=bp->right bp-
>left=add(&bp+4)
12 20: 03 01 add (%ecx),%eax                  //所以200-4=28*CNT
13 22: 89 44 91 08 mov %eax,Ox8(%ecx,%edx,4)
14 26: 5b pop %ebx
15 27: 5d pop %ebp
16 28: c3 ret
```