3.1．请写出如下递归程序的输出结果。

```
#include <stdio.h>
#include <stdlib.h>
int Fb(int x);
void main( )
{ int k=40, val;
   val= Fb(k);
  printf(" %d , %d 。" , val, k );
}

int Fb(int x)
{ int y;
  if (x <= 5 )   y = x / 5;     /* x/5 表示 x 整除 5*/
  else    y = Fb(x - 26) + x/3;
  printf("%d , ", x );
  printf("%d ; ", y );
  return ( y );
}
```

（1）主程序执行【int k=40, val; val= Fb(k);】后，调用递归函数 **Fb(40)**，子程序 **Fb(40)**入栈顺序为：

入栈顺序	堆栈		
	x	y	
3	**-12**	**Fb(-12)=-12/5=-2**	**←栈顶 Top**
2	**14**	**Fb(14)=Fb(14-26)+14/3= Fb(-12)+4**	
1	**40**	**Fb(40)=Fb(40-26)+40/3= Fb(14)+13**	**←栈底 Bottom**

即：

栈顶 Top→	**x=12，y=Fb(-12)=-12/5=-2**
	x=14，y=Fb(14)=Fb(14-26)+14/3= Fb(-12)+4
栈底 Bottom→	**x=40，y=Fb(40)=Fb(40-26)+40/3= Fb(14)+13**

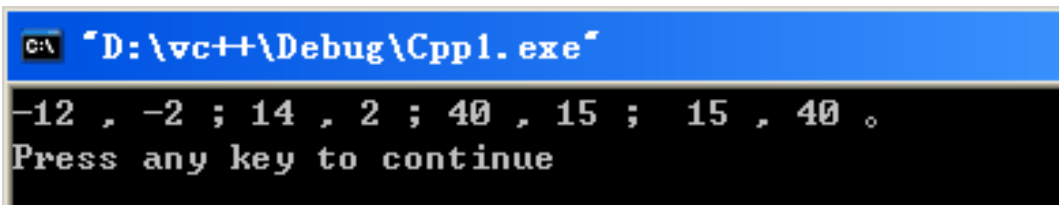
（2）子程序 **Fb( )**出栈顺序：

出栈次序	当前栈顶元素		输出数据	
	x	y	printf("%d , ", x);	printf("%d ; ", y);
1	**-12**	**Fb(-12)=-12/5=-2**	**-12，**	**-2；**
2	**14**	**Fb(14)= Fb(-12)+4=-2+4=2**	**14，**	**2；**
3	**40**	**Fb(40)= Fb(14)+13=2+13=15**	**40，**	**15；**

（3）主程序【**val= Fb(k);**】执行后【**printf(" %d , %d 。" , val, k);**】的输出结果：**15，40**。

因此，该程序执行结果为：**-12，-2；14，2；40，15；**15，40。

程序运行结果截图如下：



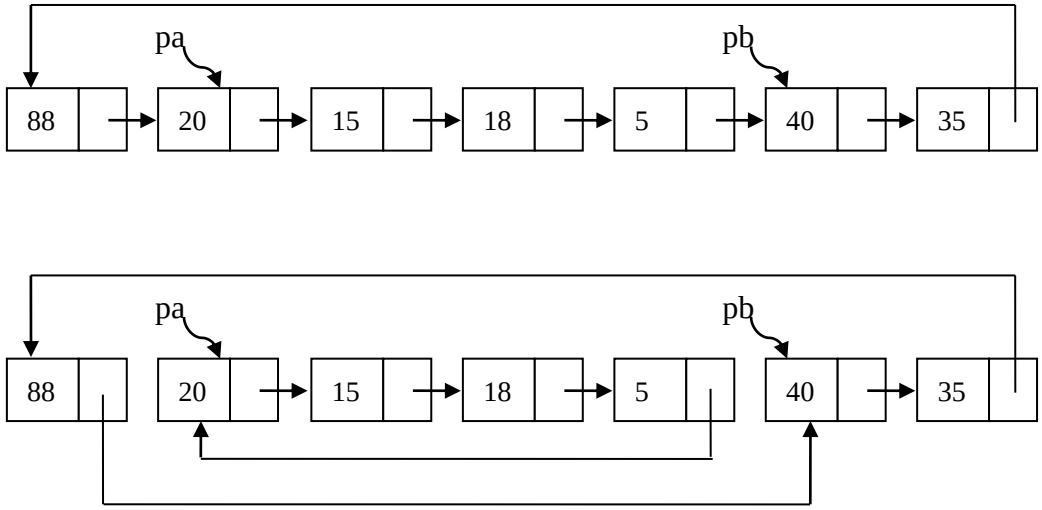
3.2. 简述下列算法的功能

```
void Split(Lnode *s , Lnode *q )
{
   Lnode *p;
   p=s;
   while ( p->next !=  q ) p = p->next;
   p->next = s;
}

void AtoBB(Lnode *pa , Lnode *pb )
{
   //pa和pb分别指向单循环链表( 结点数 > 1 )中的两个结点.
   split(pa,pb);
   split(pb,pa);
}
```

解：

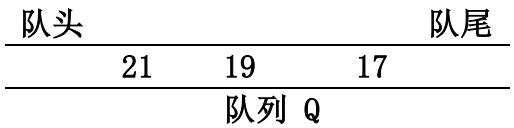
该算法的功能为：将原单循环链表从 pa 和 pb 所指向的结点断裂，即原来指向 pb 结点的链，转向指 pa，而原来指向 pa 结点的链，转向指 pb。这样就形成两个独立的单循环链表。



3.3．试读下列栈和队列操作的算法，请给出调用并执行 testit(3)后的所有输出结果。

```
//Stack为栈，Queue为队列
//InitStack(S)为构造一个空栈，InitQueue(Q)为构造一个空队列。
//EnQueue( Q , nb )表示入队列操作；DeQueue(Q)表示出队列操作。
void testit( int m )
{
   Stack S;
   Queue Q;
   int  na , nb , nm ;
   InitStack(S);   InitQueue(Q);
   na=12;   nb=21;   nm=m;
   while( na < nb )
   {
      Push( S ,na ); EnQueue( Q , nb );
      na++; nb -=2;  nm++;
   }
   printf("nm=%d,na=%d,nb=%d\n",nm,na,nb);
   while ( nm > 0 )
   {
      nm -= 2;
      na=POP(S);
      nb=DeQueue(Q) + na;
      printf("Out:%d\n",nb);
   }
}
```

解：



入栈/入队：

nm	na	nb
3	12	21
4	13	19
5	14	17
6	15	15

输出：nm=6, na=15, nb=15

出栈/出队：

nm	na	nb	输出
6 – 2 = 4	14	21 + 14 = 35	35
4 – 2 = 2	13	19 + 13 = 32	32
2 – 2 = 0	12	17 + 12 = 29	29

输出：Out:35

Out:32

Out:29

所以，输出结果为：

nm=6, na=15, nb=15

Out:35

Out:32

Out:29