#include<stdio.h>

#include<stdlib.h>

void Merge(int \*a,int \*b,int \*c,int nA,int nB,int& nCout )//此处a b c为指针类型的数组名，传值方法不是“地址传递”，不能改变实参的值

//&nCout为“引用传递”，即引用实参的值，这种方式可以真正改变实参的值

{

int i=0;//初始化数组a的下标

int j=0;//初始化数组b的下标

int k=0;//初始化数组c的下标

while(i < nA && j < nB)//当数组都没有结束比较

{

if(a[i] < b[j])//若a的元素小于b的元素

{

c[k++] = a[i];

++i;

nCout++;//将数组a的元素加到c中

}

else if(a[i] == b[j])//若a的元素等于b的元素

{

c[k++] = a[i];

++i;

++j;

nCout++;//将a或b的元素加入c中，a b的下标都要加1,c中元素个数也要加1

}

else//否则将b的元素加入c中

{

c[k++] = b[j];

++j;

nCout++;

}

}

if(i==nA)//若a的元素首先遍历完毕

{

for(int m=j;m<nB;++m)

{

c[k++] =b[m];

nCout++;

}//将b剩下的元素加入c中,注意！a b本身就为非降序，若不为非降序，这里达不到排序的效果

}

else//若b首先遍历完毕，则将a 剩下的元素加入c中

{

for(int m=i;m<nA;++m)

{

c[k++]=a[m];

nCout++;

}

}

}

int main()

{

int\* a=(int\*)malloc(sizeof(int)\*6);//为a b分配动态空间，有多少元素就分配多少

int\* b=(int\*)malloc(sizeof(int)\*3);

a[0]=2;

a[1]=3;

a[2]=4;

a[3]=6;

a[4]=8;//非降序

a[5]=9;

b[0]=7;

b[1]=9;

b[2]=10;

int nOut = 0;//初始化输出个数

int\* output=(int\*)malloc(12\*sizeof(int));//为最后输出的函数分配动态空间

Merge(a,b,output,6,3,nOut);//此处的六个参数为实参，可以为常量，变量或者表达式，目的是将实参的值,注意此处的a b并没有随形参的变化改变！只有调用的时候写Merge(&a,&b)才会改变a b的值

//传递给形参，类型和数量必须与形参相同

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

{

printf("%d,",output[i]);

}//打印最后输出的数组

printf("\n");

free (a);//释放空间

a=NULL;

free (b);

b=NULL;

free (output);

output=NULL;

printf("\n");

system("pause");

return 0;

}