#include<iostream>

#include"max.h"

#include"string"

#include"math.h"

#include"vector"

#include"map"

#include <unordered\_set>

#include <algorithm>

#include <cstdlib>

#include <sstream>

using namespace std;

//下面的main函数请勿修改

#include <stdio.h>

#include <stdlib.h>

struct node

{//链表结点类型，包含一个存放整型数据的 data 成员，和指向前驱和后继结点的指针

    int data ;

    struct node \*llink, \*rlink ;

};

struct Hnode

{

    //双向链表头结点,有两个指针成员，分别指向双向链表的第一个结点和最后一个结点

    struct node \*head, \*tail;

};

//第二关代码

struct Hnode \*createDlist()

{//函数功能：创建一个带头结点的双向链表，tail指向尾结点；head指针指向第一个结点，返回值是指向头结点的指针

  struct Hnode \*list=(struct Hnode \*)malloc(sizeof(struct node));

  struct  node \* head=(struct node \*) malloc(sizeof(struct node));

  list->head=head;

  list->tail=head;

  head->data=0;

  head->llink=NULL;

  head->rlink=NULL;

  return list;

}

void insertDlist(struct Hnode \*list, int insData)

{

 //在双向链表的表头插入数据

 struct node \* head=list->tail;

 struct node \* p=(struct node \*) malloc(sizeof(struct node));

 p->data=insData;

 p->rlink=NULL;

 p->llink=NULL;

 if(head->llink==NULL&&head->rlink==NULL){//特殊情况

            p->llink=head;

            head->rlink=p;

            p->rlink=head;

            head->llink=p;

            list->head=p;

            return ;

 }

 struct node \*nex=head->rlink;

 p->rlink=nex;

 p->llink=head;

 nex->llink=p;

 head->rlink=p;

 return ;

}

int deleteData(struct Hnode  \*list, int delData)

{

    //在双向链表中删除值为delData的第一个结点，若删除成功返回1，否则返回0

    int k=0;

    struct node \*head=list->tail;

    struct node \*pre=head;

    struct node \*cur=head->rlink;

    while(cur!=head){

        if(cur->data==delData){

            k=1;

            break;

        }

        pre=cur;

        cur=cur->rlink;

    }

    if(k==1){

  struct node \*nex=cur->rlink;

    pre->rlink = nex;

    nex->llink=pre;

    free(cur);

    }

  return k;

}

void printDlist(struct Hnode \*list)

{

 //输出双向循环链表中各数据元素的值，每输出一个数据元素换行

    struct node \*head=list->tail;

    struct node \* cur = head->rlink;

    if(list->tail==list->head)  return;

     while(head!=cur){

        printf("%d\n",cur->data);

        cur=cur->rlink;

     }

}

int destroyDlist(struct Hnode \*list)

{//释放双向链表占用的存储空间，释放所有结点,返回释放的结点数，不含list本身

    struct node \*head=list->tail;

    struct node \* ta=list->head;

    ta->rlink =NULL;

   head->llink=NULL;

   list->tail=head;

   list->head=head;

    struct node \*pre=head->rlink;

    int sum=0;

    while(pre!=NULL){

        struct node \*tem=pre;

        pre=pre->rlink;

        free(tem);

        sum++;

    }

    return sum;

}

int main(void){

struct Hnode \*list=createDlist();

int n=0;

scanf("%d",&n);

for(int i=0;i<n;i++){

    int insData=0;

    scanf("%d",&insData);

    insertDlist(list,  insData);

}

printDlist(list);

int del=0;

scanf("%d",&del);

int ans =deleteData(list,  del);

printf("ans :%d\n",ans);

int sum=destroyDlist(list);

printf("sum=%d\n",sum);

printDlist(list);

}