1. 简述线性表和栈的区别？队列和栈这两种数据结构的相同点和差异处？

线性表是最基本、最简单、也是最常用的一种数据结构。在实际应用中，线性表都是以栈、队列、字符串等特殊线性表的形式来使用的。

栈是特殊的线性表，是只允许在一端进行插入和删除的线性表。允许插入和删除的叫栈顶，反之则是栈底。栈的插入称为进栈，删除称为出栈。栈的特性是：后进先出。

队列是限定只能在表的一端进行插入，而在表的另一端进行删除的线性表。允许插入的叫队尾，允许删除的叫队头。队列的特点是：先进先出。

队列和栈的相同点：它们都是线性结构，数据元素之间的关系相同。

队列和栈的差异处：栈必须按"后进先出"的规则进行操作，只允许在表尾一端进行插入和删除；而队列必须按"先进先出"的规则进行操作，允许在表尾一端进行插入，在表头一端进行删除。

1. 试证明，若借助栈输入1，2，…，n得到输出序列为P1,P2,…,Pn（它是输入序列的一个排列），则在输出序列中不可能出现这样的情形：i<k<j，使得Pj<Pk<Pi。

假设n=3，Pi=1，Pk=2，Pj=3，i=1，k=2，j=3。顺序输入时，栈内元素排序为：Pi、Pk、Pj。由于栈是后进先出的规则，所以当该栈输出时，输出序列为：Pj=3、Pk=2、Pi=1。i<k<j，Pj>Pk>Pi。假设不成立。证明成立。

1. 利用栈设计算法，从键盘上输入一批整数，然后按照相反的次序打印出来。

void conversion(int N){

InitStack(S);

while(N){

scanf(“%d”,e);

Push (S,e);

}

while(!StackEmpty(S)){

pop(S,e);

printf(“%d”,e);

}

4.试编写如的递归函数



int g(int m, int n){

if(m<0||n<0)

return ERROR;

if(m=0&&n>=0)

return 0;

if(m>0&&n>=0)

return g(m-1,2\*n)+n;

}

1. 编写一个算法，利用栈的基本运算将指定栈中的内容进行逆转。（利用两个栈）

Status ChangeStack( ){

sqstack \*st,\*st1;

int i=1;

int y,n;  
 initStack(st);

initStack(st1);

printf("请输入数字个数\n");

scanf("%d",&n);

for(i=n; i--;){

scanf("%d",&y);

push(st,y);

}

while(!StackEmpty(st)) push(st1, pop(st));

printStack(st1);

}

1. 两个栈s1和s2要共享存储空间c[1..m0],其中 一个栈底设在c[1]处，另一个栈底设在c[m0]处，分别编写s1和s2的进栈push(i,x)、退栈pop(i)和设置栈空setnull(i)的函数，其中i=1,2。注意仅当整个空间c[1..m0]占满时才发生上溢。

void push(int i, ElemType x){

if (top1=top2-1)

printf(“上溢”);

else if (i==1){

top1++;

c[top1]=x

}

else {

top2--;

c[top2]=x;

}

}

pop(int i){

if (i==1){

if(top1==0)

printf("栈1下溢出！\n");

else {

pop=c[top1];

top1--;

}

else {

if(top==m0+1)

printf("栈2下溢出！\n");

else {

pop=c[top2];

top2++;}

}

setnull(int i){

int i;

if (i==1)

top1=0;

else

top2=m0+1;

}