

《数据结构》实验报告

班级：101011901

学号：191027

姓名：杨乃宸

实验一 顺序表**的基本操作**

实验环境：Visual C++

实验目的：

1、掌握顺序线性表的定义；

2、掌握顺序线性表的基本操作，如建立、查找、插入和删除等。

实验内容：

1. 输入一组整形元素序列，建立顺序表L；
2. 逐个显示该顺序表中；
3. 在顺序表L中第i个数据元素之前插入一个元素e。 插入前表长n=L->last+1。
4. 删除顺序表L中第i个元素；

实验提示：

顺序表的定义：

typedef struct

{

ElemType elem[MAXSIZE]; /\*线性表占用的数组空间\*/

int last; /\*记录线性表中最后一个元素在数组elem[ ]中的位置

（下标值），空表置为-1\*/

}SeqList;

1. 程序源代码

#include <stdio.h>

#include <stdlib.h>

#include <malloc.h>

// 调头

#define OK 1

#define ERROR 0

#define TRUE 1

#define FALSE 0

#define ElemType int

#define MAXSIZE 100

// 宏定义

typedef struct {

ElemType elem[MAXSIZE] ;

int a ;

int last ;

} SqList ; SqList \* L ;

// 定义

int InsList ( SqList \*L , int i , ElemType e ) {

int k ;

if ( ( i < 1 ) or ( i > L -> last + 2 ) ) {

printf ( " 位置不对 " ) ;

return ( ERROR ) ;

}

if ( L -> last >= MAXSIZE - 1 ) {

printf ( " 满了 " ) ;

return ( ERROR ) ;

}

L -> last ++ ;

for ( k = L -> last ; k > i ; k -- ) {

L -> elem [ k - 1 ] = L -> elem [ k - 2 ] ;

L -> elem [ i - 1 ] = e ;

}

return ( OK ) ;

}

// 插入

int DelList ( SqList \*L , int i ) {

int k ;

if ( ( i < 1 ) or ( i > L -> last + 1 ) ) {

printf ( " 删不了 " ) ;

return ( ERROR ) ;

}

for ( k=i ; i <= L -> last ; k++ )

L -> elem [ k - 1 ] = L -> elem [ k ] ;

L -> last -- ;

return ( OK ) ;

}

//删除

int main () {

SqList l ;

SqList \* ip ;

int i , in , co , pr ;

printf ( " ① 定义\n 表长:\n " ) ;

scanf ( "%d" , &l.last ) ;

printf ( " 内容:\n " ) ;

for (i = 1 ; i <= l.last ; i++ ) {

scanf ( "%d" , &l.elem[i] ) ;

}

printf ( "\n ② 显示\n 表:\n " ) ;

for ( i = 1 ; i <= l.last ; i++ ) {

printf ( "%d " , l.elem[i] ) ;

}

printf ( "\n\n ③ 插入\n 位置:\n " ) ;

scanf ( "%d" , &in ) ;

printf ( " 内容:\n " ) ;

scanf ( "%d" , &co ) ;

InsList ( ip , in , co ) ;

printf ( " 表:\n " ) ;

for ( i = 1 ; i <= l.last ; i++ ) {

printf ( "%d " , l.elem[i] ) ;

}

printf ( "\n ④ 删除\n 位置:\n " ) ;

scanf ( "%d" , &pr ) ;

DelList ( L , pr ) ;

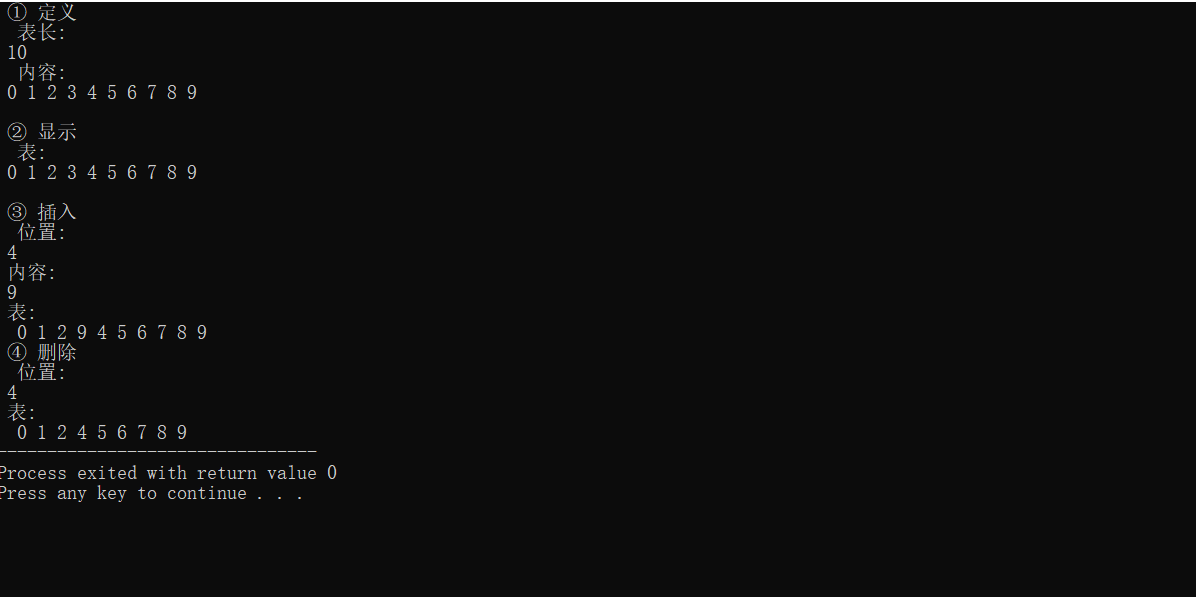
for ( i = 1 ; i <= l.last ; i++ ) {

printf ( "%d " , l.elem[i] ) ;

}

}

1. 运行结果界面



三、遇到的问题总结



《数据结构》实验报告

班级： 101011901

学号： 191027

姓名： 杨乃宸

实验二 链表**的基本操作**

实验环境：Visual C++

实验目的：

1、掌握链表相关的定义；

2、掌握链表的基本操作（创建，给链表中添加元素，查找元素，删除元素等）。

实验内容：

1. 通过键盘输入表中元素值，利用尾插法建单链表,并返回该单链表头指针L；
2. 将链表L中的元素就地逆置；（说明：逆置就是使得表中内容由原来的(a1,a2,…,ai-1,ai,ai+1, …,an）变为（an,an-1,…,ai+1,ai，ai-1, …,a1)。就地逆置就是不需要额外申请结点空间，只需要利用原有的表中的节点空间）

实验提示：

#define ElemType char

typedef struct Node /\*结点类型定义\*/

{

ElemType data;

struct Node \* next;

}Node, \*LinkList; /\* LinkList为结构指针类型\*/

1. 程序源代码

#include <stdio.h>

#include <stdlib.h>

#define ElemType char

#include <malloc.h>

// 宏一下

typedef struct Node { // 定结点类型

ElemType data ;

struct Node \*next ;

}Node , \*LinkList ; // LinkList\_结构指针类型

// 抽象一个带链表数据和地址的结点

void Illustrate ( LinkList L )

{

Node \*p,\*q;

p = L -> next ;

L -> next = NULL ;

while ( p != NULL )

{

q = p -> next ;

p -> next = L -> next ;

L -> next = p ;

p = q ;

}

}

// 结点后移的方法

LinkList Create\_list() {

LinkList L;

Node \*r, \*s;

char c;

int motto =1; // 初始化

L=(Node \* )malloc(sizeof(Node));

L->next=NULL; // 给链表头一个位置

r=L; // 一开始只有头

while(motto)

{

c=getchar();

if(c!=' ')

{

s=(Node\*)malloc(sizeof(Node)); // 给新结点空间

s->data=c;

r->next=s;

r=s;

}

else

{

motto=0;

r->next=NULL;

}

}

return L;

}

//尾插法返回指针

int main()

{

LinkList l;

Node \*p;

printf(" 打几个数\n");

l = Create\_list();

printf(" 表:\n");

p = l->next;

while(p!=NULL)

{

printf("%c\n",p->data);

p=p->next;

}

Illustrate(l);

printf(" 倒过来的表:\n");

p = l->next;

while(p!=NULL)

{

printf("%c ",p->data);

p=p->next;

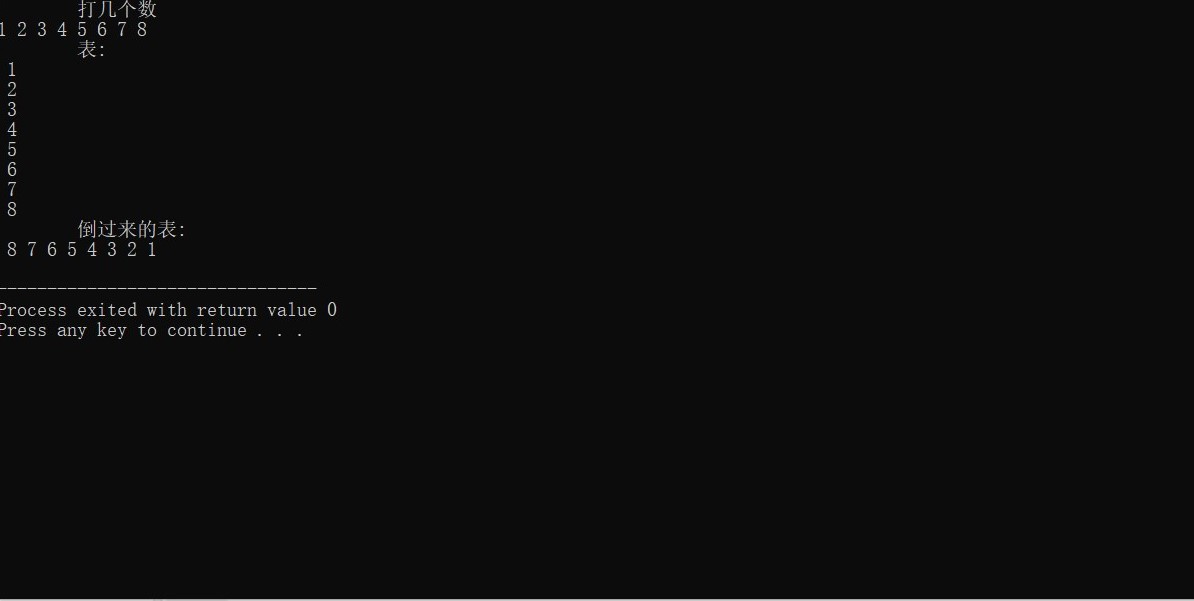
}

printf("\n");

return 0;

}

1. 运行结果界面



三、遇到的问题总结（如果有问题，请总结。如果没问题请写“无”）

无



《数据结构》实验报告

班级： 101011901

学号： 191027

姓名： 杨乃宸

实验三 堆栈和队列

实验环境：Visual C++

实验目的：

1、掌握堆栈和队列相关的定义；

2、掌握堆栈和队列的基本操作（数据元素的入栈、入队，出栈，出队等）。

实验内容：

1. 设从键盘输入一整数的序列：a1, a2, a3，…，an，试编写算法实现：用栈结构存储输入的整数，当ai≠-1时，将ai进栈；当ai=-1时，输出栈顶元素并出栈。算法应对异常情况（入栈满等）给出相应的信息。
2. 实现链式队列入队的算法。（要求：入队的时候请在队尾插入元素。队列满的条件请设置为6）。

实验提示：

Typedef int ElementType;

typedef struct node{ //节点结构

ElementType data;

struct node \*next;

}Qnode,\*PQnode;

1. 程序源代码

#include <stdio.h>

#include <stdlib.h>

#include<string.h>

#define MAXSIZE 100

enum Status{ERROR,OK};

typedef int ElemType;

typedef struct

{

ElemType \*base;

ElemType \*top;

int stacksize;

}SqStack;

Status InitStack(SqStack &S)

{

S.base=(ElemType \*)malloc(MAXSIZE\*sizeof(ElemType));

if(!S.base)

return ERROR;

S.top=S.base;

S.stacksize=MAXSIZE;

return OK;

}

Status InOutS(SqStack &S)

{

ElemType e;

int n;

if(InitStack(S))

printf("初始化成功\n");

printf("输入元素个数\n");

scanf("%d",&n);

printf("开始入栈");

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

{

scanf("%d",&e);

if(e!=-1)

{

if(S.top-S.base==S.stacksize)

{

printf("栈满\n");

return ERROR;

}

\*S.top++=e;

}

else

{

if(S.top==S.base)

{

printf("栈空\n");

return ERROR;

}

printf("%d ",\*--S.top);

printf("\n");

}

}

printf("全部出栈\n");

while(S.top!=S.base)

{

printf("%d ",\*--S.top);

}

return OK;

}

int main()

{

SqStack S;

InOutS(S);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

#include "rollin.h"

QNode \* initQueue(){

QNode \* queue=(QNode\*)malloc(sizeof(QNode));

queue->next=NULL;

return queue;

}

QNode\* enQueue(QNode \* rear,int data){

QNode \* enElem=(QNode\*)malloc(sizeof(QNode));

enElem->data=data;

enElem->next=NULL;

rear->next=enElem;

rear=enElem;

return rear;

}

QNode\* DeQueue(QNode \* top,QNode \* rear){

if (top->next==NULL) {

printf("\n队列为空");

return rear;

}

QNode \* p=top->next;

printf("%d ",p->data);

top->next=p->next;

if (rear==p) {

rear=top;

}

free(p);

return rear;

}

int main() {

QNode \* queue,\*top,\*rear;

queue=top=rear=initQueue();

rear=enQueue(rear, 1);

rear=enQueue(rear, 2);

rear=enQueue(rear, 3);

rear=enQueue(rear, 4);

rear=enQueue(rear, 5);

rear=enQueue(rear, 6);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

rear=DeQueue(top, rear);

return 0;

}

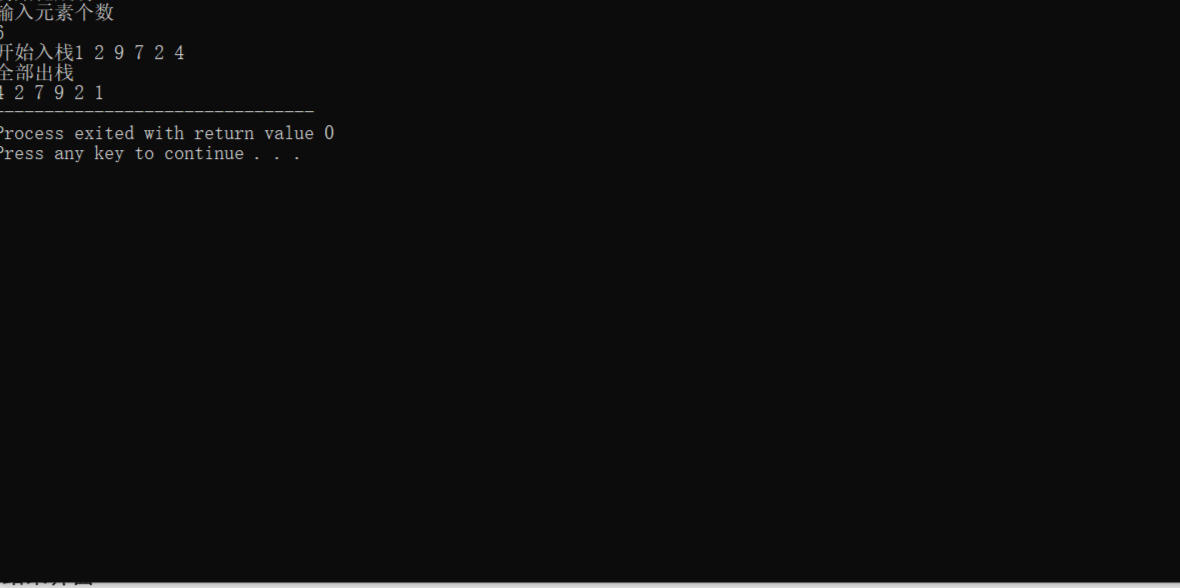
typedef struct QNode{

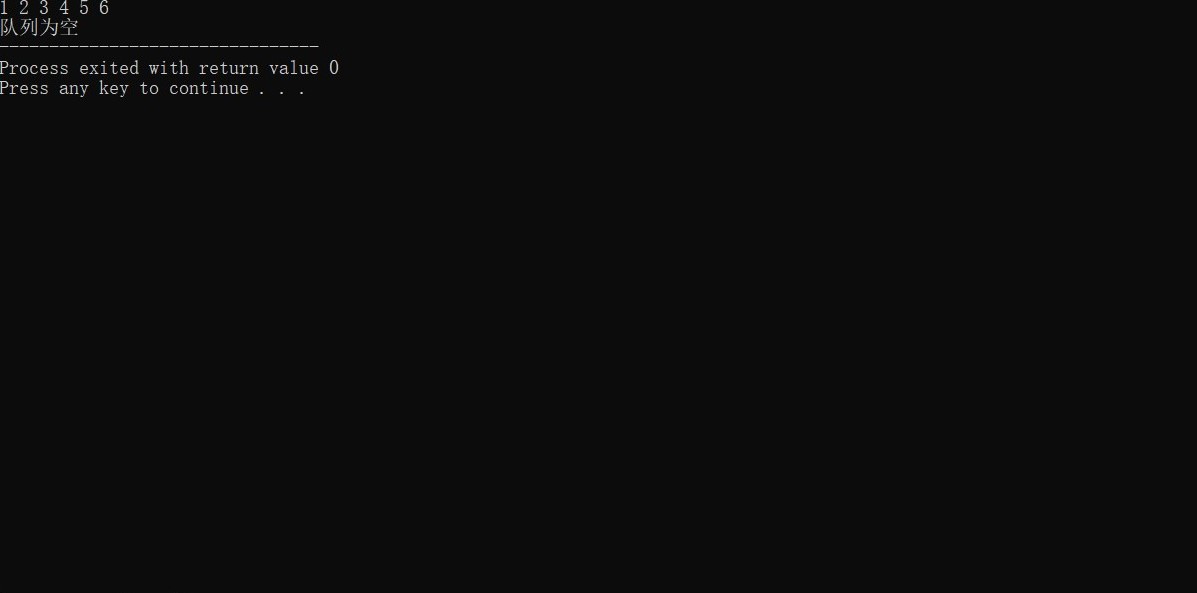
int data;

struct QNode \* next;

}QNode;

1. 运行结果界面





三、遇到的问题总结（如果有问题，请总结。如果没问题请写“无”）

无



《数据结构》实验报告

班级：

学号：

姓名：

实验三 串

实验环境：Visual C++

实验目的：

1、掌握串的相关定义；

2、掌握串的基本操作（串的复制，删除，插入，比较等）。

实验内容：

1. 创建串s。
2. 求从主串s的下标pos起，子串t第一次出现的位置，成功返回位置序号，不成功返回-1

实验提示：

#define MAXLEN 40

typedef struct { /\*串结构定义\*/

char ch[MAXLEN];

int len;

}SString;

1. 程序源代码

#define MAXLEN 40

#include <stdio.h>

typedef struct { /\*串结构定义\*/

char ch[MAXLEN];

int len;

}SString;

void createstring(SString \*s) {

int i, j ;

char c;

printf("串长度:");

scanf("%d", &j);

printf("串内容:");

for (i = 0; i < j; i++) {

scanf("%c", &c);

s->ch[i] = c;

}

s->len = j;

}

void output(SString \*s) {

int i;

if (s->len == 0)

printf("空");

else

for (i = 0; i < s->len; i++)

printf("%c", s->ch[i]);

printf("\n");

}

int StrIndex(SString s, int pos, SString t) {

int i, j, start;

if (t.len == 0)

return(0); /\* 空串时是匹配串 \*/

start = pos;

i = start;

j = 0; /\* 从下标pos和头开始 \*/

while (i < s.len && j < t.len)

if (s.ch[i] == t.ch[j])

{

i++;

j++;

}

else

{

start++; /\* 当前对应字符不等时回溯 \*/

i = start;

j = 0;

}

if (j >= t.len)

return(start); /\* 成功返回位置序号 \*/

else

return(-1); /\* 返回空值 \*/

}

#include <stdio.h>

#include <stdlib.h>

#include "SeqString.h"

#include "StringDef.h"

int main(){

SString \*str1;

SString str2;

int pos;

int flag = 0;

str1 = (SString \*)malloc(sizeof(SString));

str1->len = 0;

printf("主串:\n");

createstring(str1);

output(str1);

printf("子串:\n");

createstring(&str2);

output(&str2);

printf("查子串头:");

scanf("%d", &pos);

flag = StrIndex(\*str1, pos, str2);

if (flag == -1)

printf("查不到");

else {

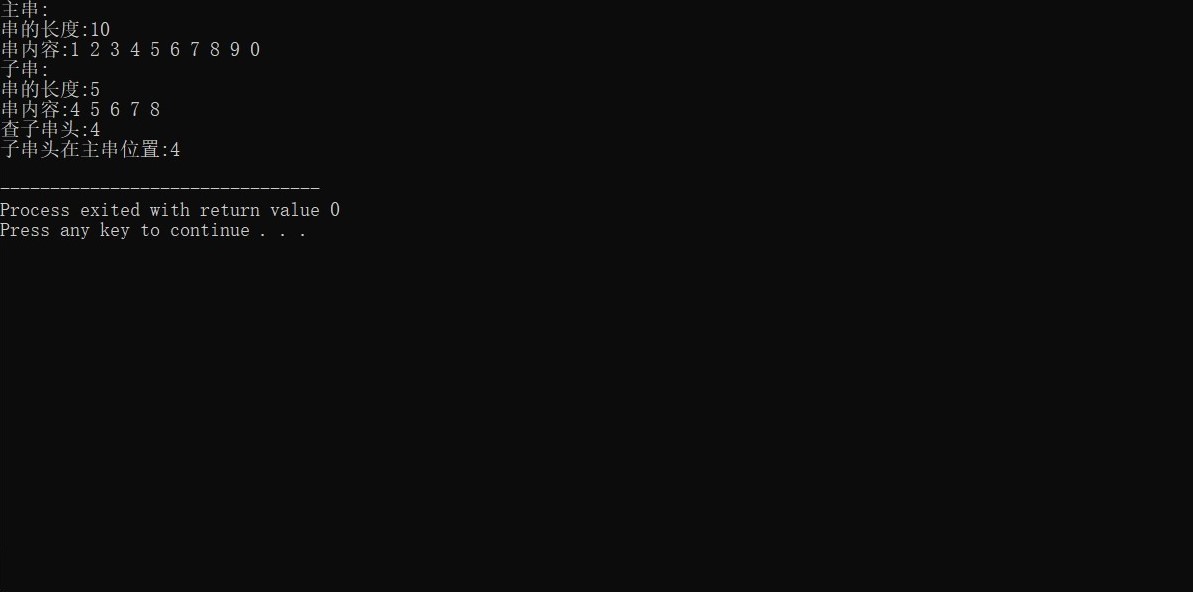
printf("子串头在主串位置:%d\n", flag);

}

return 0 ;

}

1. 运行结果界面



三、遇到的问题总结（如果有问题，请总结。如果没问题请写“无”）

无



《数据结构》实验报告

班级： 101011901

学号： 191027

姓名： 杨乃宸

实验三 树与二叉树

实验环境：Visual C++

实验目的：

1、掌握数与二叉树相关定义；

2、掌握树的基本操作（树的创建，删除，插入，遍历等）。

实验内容：

1. 创建二叉树。
2. 求上记二叉树的叶子节点。

实验提示：

typedef char DataType;

typedef struct Node

{

DataType data;

struct Node \*LChild;

struct Node \*RChild;

}BiTNode, \*BiTree;

1. 程序源代码

#include <stdio.h>

#include <stdlib.h>

typedef char DataType;

typedef struct Node {

DataType data;

struct Node \*LChild;

struct Node \*RChild;

} BiTNode, \*BiTree;

void CreatBinTree(BiTree \*bt) {

char ch;

ch = getchar();

if(ch==' ')

\*bt=NULL;

else {

\*bt=(BiTree)malloc(sizeof(BiTNode));

(\*bt)->data=ch;

CreatBinTree(&((\*bt)->LChild));

CreatBinTree(&((\*bt)->RChild));

}

}

int leaf\_b(BiTree root) {

int LeafCount;

if(root==NULL)

LeafCount=0;

else if((root->LChild==NULL)&&(root->RChild==NULL))

LeafCount=1;

else

LeafCount=leaf\_b(root->LChild)+leaf\_b(root->RChild);

return LeafCount;

}

void InOrder(BiTree root) {

if(root!=NULL) {

InOrder(root->LChild);

printf("%c",root->data);

InOrder(root->RChild);

}

}

int main() {

printf("输下二叉树的先序序列\n空的二叉树打空格\n\n二叉树的中序遍历序列:\n");

BiTree BT;

CreatBinTree(&BT);

InOrder(BT);

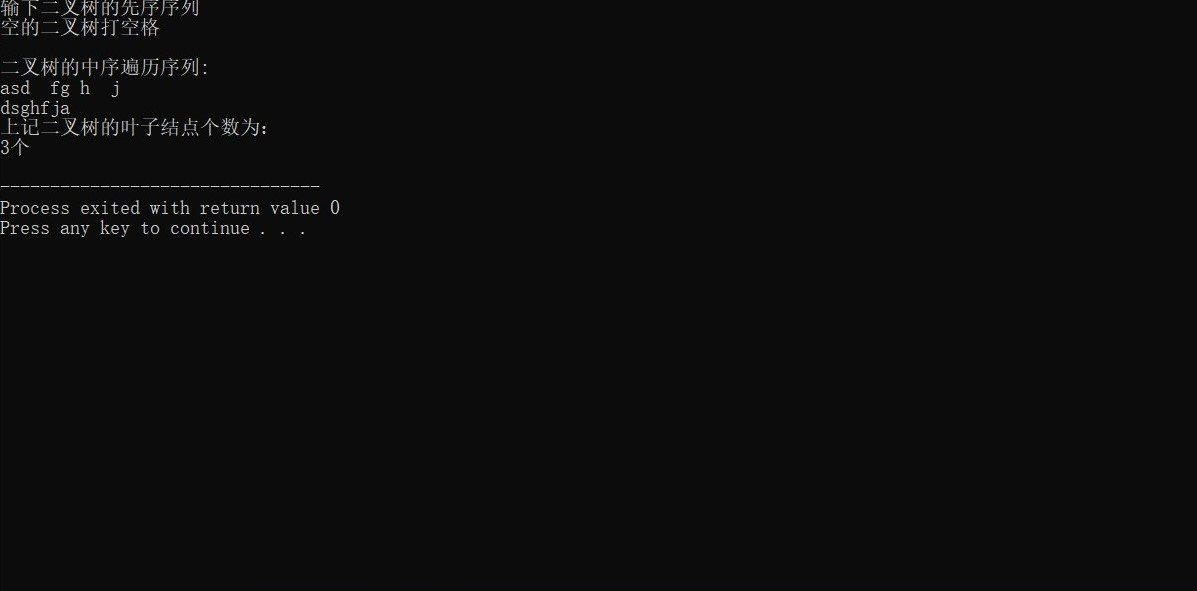
printf("\n");

printf("上记二叉树的叶子结点个数为：\n%d个\n",leaf\_b(BT));

return 0;

}

1. 运行结果界面



三、遇到的问题总结（如果有问题，请总结。如果没问题请写“无”）

无



《数据结构》实验报告

班级： 101011901

学号： 191027

姓名： 杨乃宸

实验三 排序

实验环境：Visual C++

实验目的：

1、掌握冒泡排序的思想；

实验内容：

对记录数组R[9]={23,4,6,56,7,8,42,21,9}做冒泡排序，length为数组的长度

注：数组R也可以自行定义。

实验提示：

1. 程序源代码

#include<stdio.h>

int main() {

int R[9]={23,4,6,56,7,8,42,21,9},i,j,t ;

for(i=1;i<=9;i++)

for(j=0;j<9;j++)

if(R[j]<R[j+1]) {

t=R[j];

R[j]=R[j+1];

R[j+1]=t ;

}

printf("冒泡排序法：");

for(i=0;i<10;i++) {

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

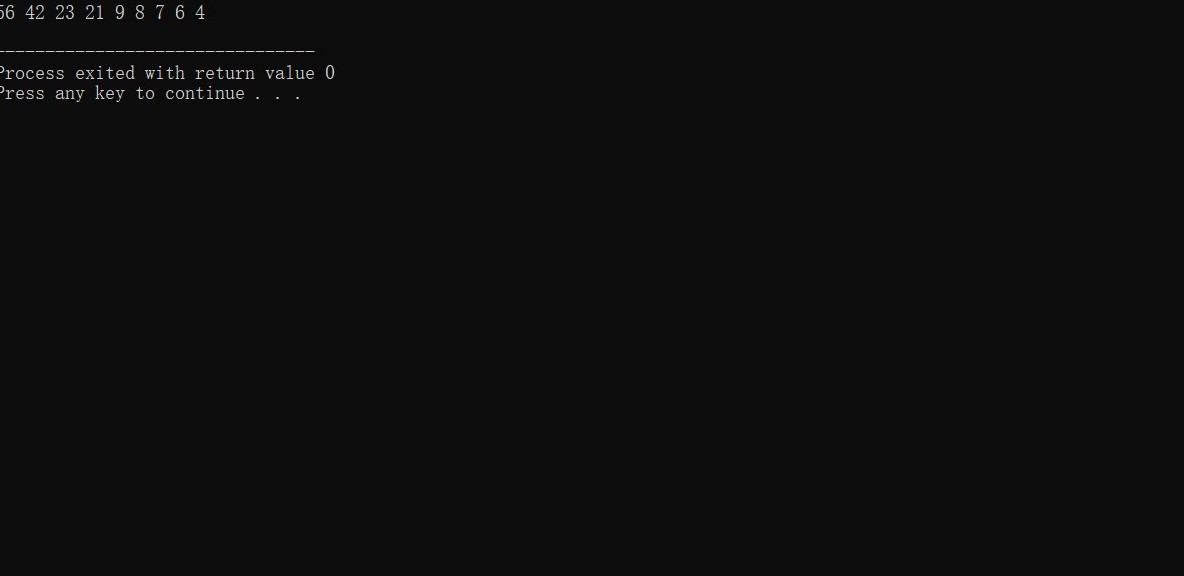
}

printf("\n",R[i]);

return 0;

}

1. 运行结果界面



三、遇到的问题总结（如果有问题，请总结。如果没问题请写“无”）

无