**一．实验目的**

掌握顺序表构造、查找、插入和合并等算法

**二．实验内容**

动态创建一个有5个数据元素的顺序表；

从键盘上输入一个新元素和插入位置，插入顺序表；

从键盘上输入一个删除位置，从顺序表中删除这个位置上的元素；

根据需要输出顺序表中的内容。

**三．实验步骤和实验结果**

**1.**

void create(struct sqlist &L)

{

int n;

scanf("%d",&n);

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

{

scanf("%d",&L.data[i]);

L.length++;

}

}

2.

void insert(struct sqlist &L,int n)

{

if(n<1||n>L.length+1)

{

printf("插入位置出错");

}

else

{

int d;

scanf("%d",&d);

for(int j=L.length-1;j>=n-1;j--)

{

L.data[j+1]=L.data [j];

}

L.data [n-1]=d;

L.length++;

Show(L);

}

}

**3.**

void de(struct sqlist &L,int n)

{

if(n<1||n>L.length+1)

{

printf("删除位置出错");

}

else

{

for(int j=L.length-1;j>=n-1;j--)

{

L.data[j-1]=L.data [j];

}

L.length--;

show（L）;

}

}

4.

void show(struct sqlist &L)

{

for(int i=0;i<L.length ;i++)

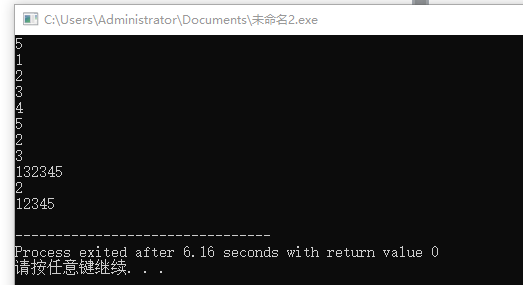
{

printf("%d",L.data [i]);

}

printf("\n");

}



（图为创建一个5个元素的顺序表，在2号位插入数据为3的新元素，后删除2号位元素。）

**以下为全部代码：**

#include<stdio.h>

#include<stdlib.h>

#define maxsize 100

typedef struct sqlist{

int \*data;

int length;

};

void create(struct sqlist &L)

{

int n;

scanf("%d",&n);

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

{

scanf("%d",&L.data[i]);

L.length++;

}

}

void show(struct sqlist &L)

{

for(int i=0;i<L.length ;i++)

{

printf("%d",L.data [i]);

}

printf("\n");

}

void insert(struct sqlist &L,int n)

{

if(n<1||n>L.length+1)

{

printf("插入位置出错");

}

else

{

int d;

scanf("%d",&d);

for(int j=L.length-1;j>=n-1;j--)

{

L.data[j+1]=L.data [j];

}

L.data [n-1]=d;

L.length++;

show(L);

}

}

void de(struct sqlist &L,int n)

{

if(n<1||n>L.length+1)

{

printf("删除位置出错");

}

else

{

for(int j=n;j<=L.length-1;j++)

{

L.data[j-1]=L.data [j];

}

L.length--;

show(L);

}

}

int main()

{

struct sqlist L;

int arr[maxsize];

L.data =arr;

L.length=0;

create(L);

int x,y;

scanf("%d",&x);

insert(L,x);

scanf("%d",&y);

de(L,y);

return 0;

}

四．分析与讨论

1.顺序表在建立是要注意指针的指向

2.在插入操作时，插入后，要从最后一个位置开始，后一个赋值前一个，避免数据被覆盖掉。

3.在删除操作时，删除后，要从删除后最后一个位置开始，前一个给后一个赋值。

4.插入和删除操作相似，但不完全相同，在覆盖时，要注意时前一个覆盖后一个还是后一个覆盖前一个。