|  |  |  |
| --- | --- | --- |
| 郭沫若校名  **《数据结构》**  **实验报告**  新校徽 | | |
|  | 题 目: | 实验一 线性表（顺序表部分） |
|  | 姓 名: | 王晓悦 |
|  | 学 院: | 信息与网络工程学院 |
|  | 专 业: | 物联网工程 |
|  | 班 级: | 171 |
|  | 学 号: | 2704170119 |
|  | 指导教师: | 李德胜 职称:教授 |
| 2018 年 10 月 10 日 | | |

一、实验要求

本次实验的重点包括以下内容：

1、熟练掌握顺序表的存储结构

2、熟练掌握顺序表的基本操作

二、程序代码

#include< iostream >

using namespace std;

const int maxsize=1000;

template < class T >

class Seqlist{

public:

Seqlist():length(0) {}

Seqlist(const T a[],int n);

int Getlength();

void Printlist();

void Insert(int i,T x);

T Delete(int i);

T Get(int i);

int Locate(T x);

private:

T date[maxsize];

int length;

};

template < class T >

Seqlist< T >::Seqlist(const T a[],int n){

if(n>maxsize)

throw "error";

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

date[i]=a[i];

}

length=n;

}

template < class T >

int Seqlist< T >::Getlength(){

return length;

}

template < class T >

void Seqlist< T >::Printlist(){

cout<<"遍历线性表的各个数据元素"<<endl;

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

cout<<date[i]<<' ';

}

cout<<endl;

}

template < class T >

void Seqlist< T >::Insert(int i,T x){

if(length>maxsize||i<1||i>length+1)

throw "error";

for(int j=length;j>=i;j--){

date[j]=date[j-1];

}

date[i-1]=x;

length++;

}

template < class T >

T Seqlist< T >::Delete(int i){

if(length==0||i<1||i>length)

throw "error";

T x=date[i-1];

for(int j=i;j<length;j++){

date[j-1]=date[j];

}

length--;

return x;

}

template < class T >

T Seqlist< T >::Get(int i){

if(i<1||i>length)

throw "error";

return date[i-1];

}

template < class T >

int Seqlist< T >::Locate(T x){

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

if(date[i]==x)

return i+1;

}

return 0;

}

int main(){

int a[1000],n;

cout<<"输入元素个数(<1000)"<<endl;

cin>>n;

cout<<"输入元素"<<endl;

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

cin>>a[i];

Seqlist < int > list1(a,n);

list1.Printlist();

cout<<"向位置2插入元素0后";

list1.Insert(2,0);

list1.Printlist();

int x=list1.Delete(4);

cout<<"删除元素："<<x<<endl;

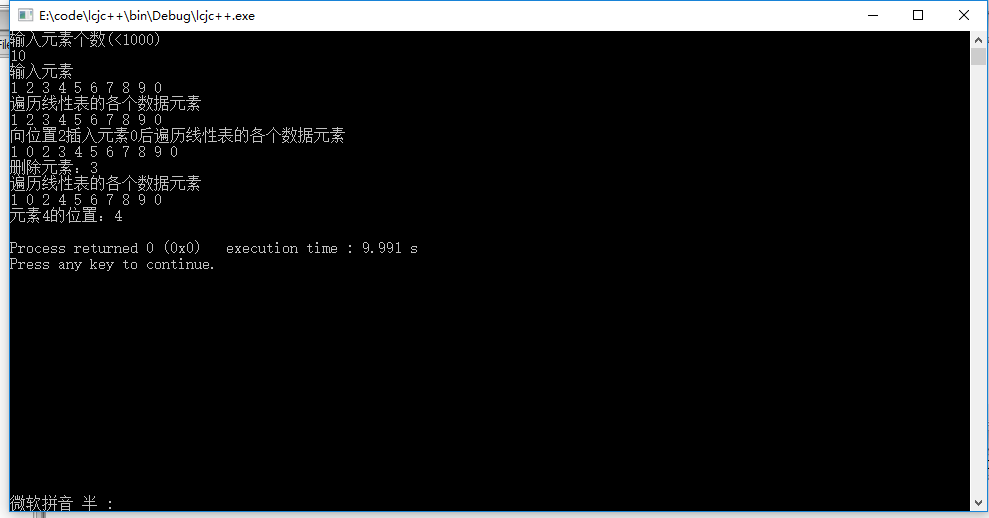
list1.Printlist();

int p=list1.Locate(4);

cout<<"元素4的位置："<<p<<endl;

}

三、运行结果



四、总结

顺序表的存储结构利用物理位置上的相邻关系来表示数据元素之间的逻辑关系，顺序表的优点如下：

1.不需要伟表示元素之间的逻辑关系而增加额外的存储空间；

2.可以方便地随机访问顺序表中的任意位置的元素；

顺序表的缺点如下：

1.插入和删除操作需要移动大量的数据元素，效率较低，在等概率条件下，两种操作平均需要移动顺序表中约一半的数据；

2.顺序表难以选择合适的存储容量，顺序表要求占用连续的存储空间，存储分配只能预先进行，因此属于静态存储分配方式，若开始时分配的空间过小，则插入操作很容易引起顺序表的溢出，若分配的空间过大，则可能造成一部分的空间长期闲置，不能充分利用。