

**数据结构实验报告**

**学 院** 信息工程学院

**专 业** 计算机类（大类招生）

**班 级** 2016240210

**姓 名** 瞿强鑫

**学 号** 2016902094

**指导教师** 史昕

**完成时间** 2017.4.10

**一、实验目的**

1. 掌握线性表链接存储结构；
2. 验证单链表及其基本操作的实现；
3. 进一步理解算法与程序的关系，能够将单链表算法转化为对应程序。

**二、实验设备（环境）及要求**

电脑一台，VC6.0环境下运行

**三、实验内容（步骤）与方法**

本程序通过建立链接表，实现对链接表的求长度、各元素的查找、插入和删除操作.

//建立链表类

**template<class Data>**

**class LinkList**

{

**public:**

LinkList(); //无参构造函数，建立只有头结点的空链表

LinkList(Data a[],int n); //有参构造函数，建立n个元素的单链表

~LinkList(){} //析构函数

int Length(); //求链表长度

Data Get(int i); //查找第i个元素

int Locate(Data x); //查找值为x的位置

void Insert(int i,Data x); //在位置i插入x

Data Delete(int i); //删除第i个元素

void PrintList(); //遍历

**private:**

Node<Data>\*first;

};

//无参构造函数，建立只有头结点的空链表

**template<class Data>**

**LinkList<Data>::LinkList()**

{

Node<Data> \*first=new Node<Data>;

first->next=NULL;

}

//有参构造函数，建立n个元素的单链表

**template<class Data>**

**LinkList<Data>::LinkList(Data a[],int n)**

{

first=new Node<Data>;

Node<Data> \*r=first;

Node<Data> \*s;

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

{

s=new Node<Data>;s->data=a[i];

r->next=s;r=s;

}

r->next=NULL;

}

//求链表长度

**template<class Data>**

**int LinkList<Data>::Length()**

{

int count=0;

Node<Data> \*p=first->next;

while(p!=NULL)

{

p=p->next;

count++;

}

return count;

}

//查找第i个元素

**template<class Data>**

**Data LinkList<Data>::Get(int i)**

{

int count=1;

Node<Data> \*p=first->next;

while(p!=NULL&&count<i)

{

p=p->next;

count++;

}

if(p==NULL) throw"位置";

else return p->data;

}

//查找值为x的位置

**template<class Data>**

**int LinkList<Data>::Locate(Data x)**

{

int count=1;

Node<Data> \*p=first->next;

while(p!=NULL)

{

if(p->data==x) return count;

p=p->next;

count++;

}

return 0;

}

//在位置i插入x

**template<class Data>**

**void LinkList<Data>::Insert(int i,Data x)**

{

int count=0;

Node<Data> \*p=first;

while(p!=NULL&&count<i-1)

{

p=p->next;

count++;

}

if(p==NULL) throw"位置";

else

{

Node<Data> \*s=new Node<Data>;s->data=x;

s->next=p->next;p->next=s;

}

}

//删除第i个元素

**template<class Data>**

**Data LinkList<Data>::Delete(int i)**

{

int count=0,x;

Node<Data> \*p=first;

while(p!=NULL&&count<i-1)

{

p=p->next;

count++;

}

if(p==NULL||p->next==NULL)

throw "位置";

else

{

Node<Data> \*q=p->next;x=q->data;

p->next=q->next;

delete q;

return x;

}

}

//遍历

**template<class Data>**

**void LinkList<Data>::PrintList()**

{

Node<Data> \*p=first->next;

while(p!=NULL)

{

cout<<p->data<<" ";

p=p->next;

}

}

**四、实验结果与分析**

main（）函数如图1：

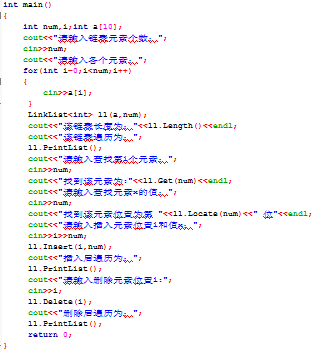


图1

程序开始运行后输入待值到链表，如图2，存入五个元素；



图2

存入后遍历有图3：

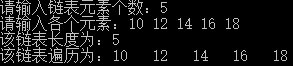


图3

输入查找第3个元素结果如图4，找到该值为14；

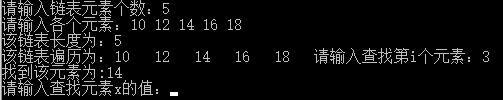


图4

查找值为18的元素，查找成功，如图5；

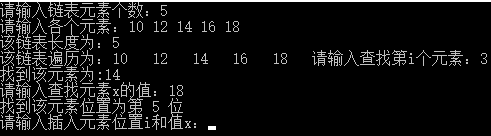


图5

选择在第三个位置插入13的值，插入成功后遍历结果如图6；

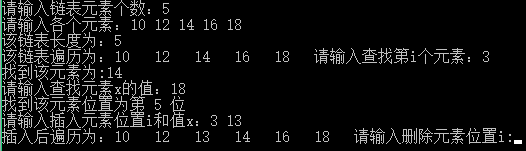


图6

选择删除位置为2的元素，删除操作成功并遍历，结果如图7；

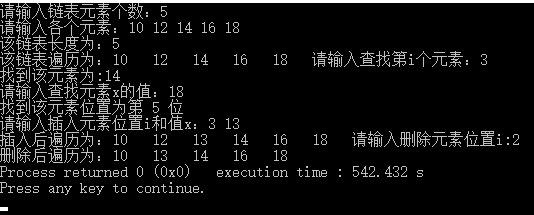


图7

**五、实验心得与体会**

这次程序写完才发现比自己想象的难得多，自己之前对链表的操作仅仅是了解了个皮毛，但好在此次实验过后，自己感觉对链表的认识加深了不少，从课本上的抽象知识到现在脑袋里的具体步骤，从课本上的一个个结点到我建立的struct LinkList类型，包含了数据域和指针域，这并不是说起来那么简单，每个结点都需要声明申请空间，每个数据域得存入数值，且指针域的地址指向得正确把握。

其次，在编程时，我发现自己对模板类的认知完全不够，只知道DataType LinkList<DataType> Get(int i)这样用，而不知道DataType作为未定类型起到的作用，也理解到template <class DataType>对成员函数的作用和必不可缺的前提。

最后说一下使用vc++时遇到的巨大问题吧，我的电脑系统win10的一次较大更新后，vc++无法打开了，我从网上下载了共计不下12次各个版本的vc6.0，可是电脑再打开时还是出错了。没有办法我只好使用codeblocks来编程，使用后我的第一感觉就是方便，可是在几次编译后我认识到了它的不足，它对程序编译小细节可以说是很不严谨，对一些括号的使用正确与否都没有错误提示甚至warning都没有，这对我们这些程序初学者来说是个巨大的灾难，所以我必须解决我电脑不能运行vc6.0的问题，这也是一个警告，希望以后我能够严谨地编程，养成良好的习惯。