实验-1-线性表-参考程序

王 新 宇
计算机科学与技术系

```
//实验题第1题第（1）小题
template <class DataType>
DataType DelMinElem(SqList<DataType> &list)
{
    DataType min,tmp;
    int pos;
    //寻找最小值min，并记录其序号pos
    list.GetElem(1, min);     pos = 1;
    for(int i = 2; i <= list.GetLength(); i++)  {
        list.GetElem(i, tmp);
        if(tmp < min)  {
            min = tmp;      pos = i;
        }
    }
    //用最后一个元素替换该最小值，并删除最后一个元素
    //若最小值为最后一个元素，则无需替换
    if(pos != list.GetLength())  {                  //最小值不是最后一个元素
        list.GetElem(list.GetLength(), tmp);
        list.SetElem(pos, tmp);
    }
    list.DeleteElem(list.GetLength(), tmp);   //删除最后一个元素
    return min;
}
```

```
//实验题第1题第（2）小题
template <class DataType>
void DelValue(SqList<DataType> &list, DataType x)
{
    DataType e;
    int pos = 1;
    while (pos <= list.GetLength()) {
        list.GetElem(pos, e);
        if (e == x)    list.DeleteElem(pos, e);
        else    pos++;
    }
}
```

```
//实验题第1题第（3）小题
template <class DataType>
void DelRepElem(SqList<DataType> &list)
{
    DataType e, t;
    int pos = 1;
    while (pos <= list.GetLength() - 1)  {
        list.GetElem(pos, e);
        int i = pos + 1;
        while (i <= list.GetLength())  {
            list.GetElem(i, t);
            if (t == e)    list.DeleteElem(i, t);
            else    i++;
        }
        pos++;
    }
}
```

```
//实验题第2题第（1）小题
template <class DataType>
Status OrdListInsertElem(SqList<DataType> &list, DataType e)
{
    if (list.GetLength() == list.GetMaxSize())        return OVER_FLOW;
    else  if(list.IsEmpty())
        list.InsertElem(1, e);
    else
    {
        DataType tmp;
        int pos = 1;
        while(pos <= list.GetLength())
        {
            list.GetElem(pos, tmp);
            if(e > tmp)    pos++;
            else    break;
        }
        list.InsertElem(pos, e);
    }
    return SUCCESS;
}
```

```
//实验题第2题第（2）小题
template <class DataType>
Status OrdListDeleteElem(SqList<DataType> &list, DataType e)
{
    DataType tmp;
    int pos = 1, count = 0;
    while(pos <= list.GetLength())  {
        list.GetElem(pos, tmp);
        if (tmp > e)    break;
        if (tmp == e) {
            list.DeleteElem(pos, tmp);
            count++;
        }
        else    pos++;
    }
    return count ? SUCCESS : FAILED;
}
```

```
//实验题第2题第（3）小题
template <class DataType>
Status OrdListMerge(SqList<DataType> list1,SqList<DataType> list2,SqList<DataType> &list3)
{
    if(list1.GetLength() + list2.GetLength() > list3.GetMaxSize())    return OVER_FLOW;
    int pos1 = 1,pos2 = 1, pos3 = 1;
    DataType value1,value2;
    while(pos1<=list1.GetLength() && pos2<=list2.GetLength())  {
        list1.GetElem(pos1,value1);      list2.GetElem(pos2,value2);
        if(value1 < value2)
        {   list3.InsertElem(pos3++, value1);    pos1++;    }
        else
        {   list3.InsertElem(pos3++, value2);    pos2++;    }
    }
    while(pos1 <= list1.GetLength())  {
        list1.GetElem(pos1,value1);      list3.InsertElem(pos3++, value1);
        pos1++;
    }
    while(pos2 <= list2.GetLength())  {
        list2.GetElem(pos2,value2);      list3.InsertElem(pos3++, value2);
        pos2++;
    }
    return SUCCESS;
}
```

```
//实验题第2题第（4）小题
template <class DataType>
Status OrdListIntervalDelete(SqList<DataType> &list, DataType s, DataType t)
{
    if(list.IsEmpty() || s>=t)  {
        cout << "Empty list or wrong parameter!" << endl;    return FAILED;
    }
    DataType tmp;
    int pos = 1, count = 0;
    while (pos <= list.GetLength())  {
        list.GetElem(pos, tmp);
        if (tmp > t)    break;
        if (tmp >= s) {
            list.DeleteElem(pos, tmp);
            count++;
        }
        else    pos++;
    }
    return count ? SUCCESS : FAILED;
}
```

```
//实验题第3题第（1）小题
template <class DataType>
Node<DataType> * LinkList<DataType>::LocateAddress(int i) const
{
    if (i < 1 || i > length)   return NULL;
    else
    {
        Node<DataType> *p = head->next;
        int count;
        for (count = 1; count < i; count++)
            p = p->next;
        return p;
    }
}
```

```
//实验题第3题第（2）小题
template <class DataType>
int Frequency(LinkList<DataType> & list, const DataType e)
{
    if(list.IsEmpty())
        return 0;
    int count = 0;
    DataType tmp;
    for(int i=1; i<=list.GetLength(); i++)
    {
        list.GetElem(i, tmp);
        if(tmp == e)
            count++;
    }
    return count;
}
```

```
//实验题第4题第（1）小题
template <class DataType>
void OrdListInsertElem(LinkList<DataType> &list, DataType e)
{
    if(list.IsEmpty())   list.InsertElem(1, e);
    else  {
        int pos = 1;
        DataType tmp;
        while(pos <= list.GetLength())   {
            list.GetElem(pos, tmp);
            if(e > tmp)    pos++;
            else    break;
        }
        list.InsertElem(pos, e);
    }
}
```

```
//实验题第4题第（2）小题
template <class DataType>
Status OrdListDeleteElem(LinkList<DataType> &list, DataType e)
{
    DataType tmp;
    int pos = 1, count = 0;
    while(pos <= list.GetLength())  {
        list.GetElem(pos, tmp);
        if (tmp > e)    break;
        if (tmp == e) {
            list.DeleteElem(pos, tmp);
            count++;
        }
        else    pos++;
    }
    return count ? SUCCESS : FAILED;
}
```