# 数据元素的文件读写

实验中，涉及到的文件读写都是以数据元素为单位的，有关参考代码如下。

#include <stdio.h>

#include <stdlib.h>

typedef struct {

char c;

int d;

float f;

} ElemType;

typedef struct {

ElemType elem[10];

int length;

} SqList;

SqList L={{{'a',1,1.1},{'b',2,2.2},{'c',3,3.3},{'d',4,4.4}}, 4};

int main(int argc, char \*argv[]) {

FILE \*fp;

char filename[30];

int i;

printf("input file name: ");

scanf("%s",filename);

//写文件的方法

if ((fp=fopen(filename,"w"))==NULL)

{

printf("File open erroe\n ");

return 1;

}

fwrite(L.elem,sizeof(ElemType),L.length,fp);

//这里是1次性写入，对于其它物理结构，可通过遍历，逐个访问数据元素

//并写入到文件中

fclose(fp);

//读文件的方法

L.length=0;

if ((fp=fopen(filename,"r"))==NULL)

{

printf("File open erroe\n ");

return 1;

}

while(fread(&L.elem[L.length],sizeof(ElemType),1,fp))

L.length++;

//这里从文件中逐个读取数据元素恢复顺序表，对于不同的物理结构，可通过读//取的数据元素恢复内存中的物理结构。

fclose(fp);

return 0;

}

# 线性表模板实例

#include "iostream"

using namespace std;

typedef float ElemType;

template <typename T>

struct list{

T elem[10];

int length;

};

template <typename T1>

int ListInsert(struct list<T1> &L,int i, T1 x)

{

int j;

if(i<1 || i>L.length+1) return ERROR;

if(i>100) return OVERFLOW;

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

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

L.elem[i-1]=x;

L.length++;

return OK;

}

template <typename T2>

void ListDisplay(struct list<T2> &L)

{

int i;

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

cout<<" "<<L.elem[i];

}

int main()

{

struct list<ElemType> L1;

L1.length=0;

ListInsert(L1,1,(ElemType)10.2);

ListInsert(L1,1,(ElemType)20);

ListDisplay(L1);

return 1;

}