# 数据元素的文件读写

1. 对动态分配的顺序表，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;

int length;

int listsize;

} SqList;

SqList L={{{'a',1,1.1},{'b',2,2.2},{'c',3,3.3},{'d',4,4.4}}, 4,100};

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, 1, sizeof(L), fp); //其中L.elem可以不保存，

//但L.length和L.listsize必须保存

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

//再释放动态分配的空间

fclose(fp);

//读文件的方法

L.length=0;

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

{

printf("File open erroe\n ");

return 1;

}

fread(&L, 1, sizeof(L),1,fp))； //读入顺序表的结构体

L.elem=malloc(L.listsize\* sizeof(ElemType));

fread(L.elem, L.length,sizeof(ElemType),fp))；//读入顺序表的数据元素

fclose(fp);

return 0;

}

2. 对单链表，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 node{

ElemType data;

struct node \*next;

} NODE,\*LinkList;

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

遍历单链表，逐个把结点的数据写到文件中。

//读文件的方法

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

{

printf("File open erroe\n ");

return 1;

}

tail=head= malloc(sizeof(NODE));

ElemType x;

while(read(&x, 1, sizeof(ElemType),fp)) //逐个读入顺序表的数据元素

{ //尾插法加到链表中

p=malloc(sizeof(NODE));

p->data=x;

tail->next=p;

tail=p;

}

tail->next=NULL;

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;

}