实验一　线性表

班级: 计算机 17-1 班 姓名: 冯子旋　学号：　07

一、实验目的

1．掌握线性表的一种实现方法；

2．掌握利用线性表解决实际问题的方法；

3．通过上机实践加强利用数据结构解决实际应用应用问题的能力。

二、实验题目与要求

实验题目：单链表的实现及其应用

实验要求：

（1）实现单链表的类型定义及顺序表的初始化、插入、删除、取元素、输出等操作。

（2）以采用单链表表示集合，设计程序实现集合的创建、并集及交集等运算。

三、实验内容

1．带头结点的单链表的实现

完成单链表的类型定义及其基本操作函数代码，并将其保存为“LinkList.h”。

//单链表类型及其基本操作函数的定义

```
typedef struct LNode{
    LElemType data;
    struct LNode *next;
}LNode, *LinkList;
Status InitLinkList(LinkList &L){
```

//初始创建空的带头结点的单链表 L

```
    L=(LinkList)malloc(sizeof(LNode));
    if(!L) return OVERFLOW;
    L->next=NULL;
    return OK;
}

int listLength(LinkList L){//求单链表 L 的长度
        LNode *p=L;
        int j=0;
        while(p->next){
      p=p->next;   j++;
    }
    return j;   }

Status listIsEmpty(LinkList L) {
```

//判断单链表 L 是否为空；空返回 TRUE，否则返回 FALSE

```
    if(!L)
        return TRUE;
    else return FALSE;
}
Status getElem(LinkList L,int i,LElemType &e){
```

//取单链表 L 的第 i 个元素

```
    LNode *p=L;
    int j=0;
    while(j<i && p->next){
        p=p->next;
        j++;
    }
    if(j==i){
    e=p->data;
    return OK;
    }
    else return ERROR;
}
Status equal(LElemType, LElemType);
```

//元素相等关系的比较函数声明

```
LinkList locateElem(LinkList L,LElemType e){
    //在单链表 L 中查找数据域值为 e 的结点；
    //找到，返回结点的地址；找不到，则返回 NULL
    LNode *p=L->next;
    while(p && !equal(p->data,e))
        p=p->next;
    if(p) return p;
    else return NULL;
}
Status listInsert(LinkList &L,int i, LElemType e){
```

//在单链表 L 的第 i 个位置插入元素 e

```
    int j=0;
    while(j<i-1 && p->next){
        p=p->next;
        j++;
    }
```

```
    if(j == i-1){
        q=(LNode *)malloc(sizeof(LNode));
        if(!q) return OVERFLOW;
        q->data=e;
        q->next=p->next;
        p->next=q;
        return OK;
    }
    else return ERROR;
}
Status listDelete(LinkList &L,int i,LElemType &e){
//删除单链表的第 i 个元素并由 e 返回其值
    LNode *p=L,*q;
    int j=0;
    while(j<i-1 && p->next)
    {
        p=p->next;
        j++;
    }
    if(j==i-1 && p->next){
        q=p->next;
        p->next=q->next;
        e=q->data;
        free(q);
    }
    else return ERROR;
    }
void listTraverse(LinkList L,void (*visit)(LElemType e))
{  //遍历操作
    LinkList p;
    p=L->next;
    while(p){
        visit(p->data);    p=p->next;
    }
}
```

2．单链表应用程序设计

用单链表表示集合，实现集合的创建、并集及交集等运算。设计完成空缺代码,并上机调试。

```
#include <stdio.h>
#include <stdlib.h>
#include "Status.h"
typedef int LElemType; //定义元素类型为 int 类型
#include "LinkList.h"
typedef LinkList mySetType; //定义集合类型
Status equal(LElemType e1,LElemType e2){
    //元素相等关系的比较函数
    if(e1==e2) return TRUE;
    else return FALSE;
}
void visitElem(LElemType e){ //元素的访问函数
    printf("%d ",e);
}
void creatSet(mySetType &A, int n){
    //通过输入的 n 个元素创建集合 A
    int i,e;
    InitLinkList(A);
    printf("\t 输入%d 个整数：",n);
    for(i=0; i<n; i++){
        scanf("%d", &e);
        listInsert(A,i+1,e) ;    //头端插入元素
    }
}
void setUnion(mySetType A, mySetType B,
                                    mySetType &C){
   //集合的并集运算，实现 C=A∪B
    int i,k,len,e;
    InitLinkList(C); k=0; //初始化空集合 C；k 为 C 长度
    len=listLength(A);
    for(i=1; i<=len; i++){
    //将集合 A 中所有元素在集合 C 的头端插入
        getElem(A,i,e);    listInsert(C, 1 , e);
     }
     Len=listLength(B) ; //求集合 B 的长度
    for(i=1;i<=len;i++){
        //将 B 中不属于 A 的元素，在 C 头端插入
        getElem(B,i,e);
        if(!locateElem(A, e))    //若元素 e 不在 A 中
            listInsert(C, 1, e); //在 C 的头端插入 e
    }
}
void setIntersection(mySetType &A,mySetType B) {    //
集合交集运算，实现 A=A∩B
    int i,e,len;
    len=listLength(A);    i=1;
    while(i<=len){
```

```
        getElem(A, i, e);       //取 A 的第 i 个元素给 e
        if(  !locateElem(B,e)) {//若 e 不属于 B
            listDelete(A,i,e) ; //删除 A 的第 i 个元素
            len--; //当前 A 的长度已减 1；下一位序仍为 i
        }
        else i++;   //下一元素的位序为
    }
}
void outputSet(mySetType A){
```

//集合的输出操作

```
    printf("{ ");
    listTraverse(A,visitElem);
    printf("}\n");
}

int main(){     //用于测试的主函数

     mySetType A,B,C;
     int n;
     printf("创建   种类 1 :\n");
     printf("\t 数量: ");
     scanf("%d", &n);
     creatSet(A,n);
     printf("  种类 1=");
     outputSet(A);
     printf("创建   种类 2 :\n");
     printf("\t 数量: ");
     scanf("%d", &n);
     creatSet(B,n);
     printf("  种类 2=");
     outputSet(B);
     InitLinkList(C);
     setUnion(A,B,C);
     printf("\n  全部种类  =  种类 1  ∪种类 2 =");
     outputSet(C);
     setIntersection(A,B);
     printf("  共有种类  =  种类 1  ∩种类 2=");
     outputSet(A);
     return 0;

}
```

3．程序运行结果

```
创建   种类1 :
       数量: 3
        输入3 个整数：1 2 3
 种类1={ 1  2  3 }
创建   种类2 :
       数量: 4
        输入4 个整数：2 3 4 5
 种类2={ 2  3  4  5 }

 全部种类 = 种类1 ∪种类2 ={ 5  4  3  2  1 }
 共有种类 = 种类1 ∩种类2={ 2  3 }

--------------------------------
Process exited after 8.853 seconds with return value 0
请按任意键继续. . .
```

成绩：_____