**南京师范大学中北学院**

**《数据结构》**

**实**

**验**

**报**

**告**

**姓 名: 钟玮**

**学 号: 85213214**

**班 级: 计算机2班**

**日 期: 2024年3月11日**

**单链表的基本操作**

1. **实验目的**

**1、掌握链表的定义；**

**2、掌握单链表的基本操作，如建立、查找、插入、删除及合并等。**

1. **实验内容**

**1、逆位序输入（随机产生）n个元素的值，建立带表头结点的单链线性表L；**

**2、当单链表中第i个元素存在时，其值赋给e并返回；**

**3、在带头结点的单链线性表L的第i个元素之前插入元素e；**

**4、在带头结点的单链线性表L中，删除第i个元素，并由e返回其值；**

**5、已知单链线性表La和Lb的元素按值非递减排列，归并La和Lb得到新的单链线性表Lc，Lc的元素也按值非递减排列；**

**6、遍历单链表中的数据，并打印。**

1. **实验主要代码**

**可以截图并做简要解释，如：**

1. **逆位序输入（随机产生）n个元素的值，建立带表头结点的单链线性表L**

**void CreateList\_L(LinkList &L, int n)**

**{**

**// 逆位序输入（随机产生）n个元素的值，建立带表头结点的单链线性表L**

**LinkList p;**

**int i;**

**srand((unsigned)time(NULL));**

**L = (LinkList)malloc(sizeof(LNode));**

**L->next = NULL;**

**for (i = n; i > 0; --i)**

**{**

**p = (LinkList)malloc(sizeof(LNode));**

**p->data = rand() % 50;**

**p->next = L->next;**

**L->next = p;**

**}**

**}**

1. **当单链表中第i个元素存在时，其值赋给e并返回**

**Status GetElem\_L(LinkList &L, int i, ElemType &e)**

**{**

**LinkList p;**

**p = L->next;**

**int j = 1;**

**while (p && j < i)**

**{**

**p = p->next;**

**++j;**

**}**

**if (!p || j > i)**

**return ERROR;**

**e = p->data;**

**return OK;**

**}**

1. **在带头结点的单链线性表L的第i个元素之前插入元素e**

**Status ListInsert\_L(LinkList &L, int i, ElemType e)**

**{**

**LinkList p, s;**

**p = L;**

**int j = 0;**

**while (p && j < i - 1)**

**{**

**p = p->next;**

**++j;**

**}**

**if (!p || j > i - 1)**

**return ERROR;**

**s = (LinkList)malloc(sizeof(LNode));**

**s->data = e;**

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

**p->next = s;**

**return OK;**

**}**

1. **在带头结点的单链线性表L中，删除第i个元素，并由e返回其值**

**Status ListDelete\_L(LinkList &L, int i, ElemType &e)**

**{**

**LinkList p, q;**

**p = L;**

**int j = 0;**

**while (p->next && j < i - 1)**

**{**

**p = p->next;**

**++j;**

**}**

**if (!(p->next) || j > i - 1)**

**return ERROR;**

**q = p->next;**

**p->next = q->next;**

**e = q->data;**

**free(q);**

**return OK;**

**}**

**5、已知单链线性表La和Lb的元素按值非递减排列，归并La和Lb得到新的单链线性表Lc，Lc的元素也按值非递减排列**

**void MergeList\_L(LinkList &La, LinkList &Lb, LinkList &Lc)**

**{**

**LinkList pa, pb, pc;**

**pa = La->next;**

**pb = Lb->next;**

**Lc = pc = La;**

**while (pa && pb)**

**{**

**if (pa->data <= pb->data)**

**{**

**pc->next = pa;**

**pc = pa;**

**pa = pa->next;**

**}**

**else**

**{**

**pc->next = pb;**

**pc = pb;**

**pb = pb->next;**

**}**

**}**

**pc->next = pa ? pa : pb;**

**free(Lb);**

**}**

**6、遍历单链表中的数据，并打印**

**void ListTraverse\_L(LinkList &L)**

**{**

**LinkList p;**

**p = L->next;**

**printf("当前单链表元素为：\n");**

**while (p)**

**{**

**printf("%d ", p->data);**

**p = p->next;**

**}**

**printf("\n");**

**}**

1. **main函数**

**int main()**

**{**

**LinkList la, lb, lc, ld;**

**ElemType e1, e2, e3;**

**int loc1, loc2, loc3;**

**//\***

**// 单链表la的基本操作**

**CreateList\_L(la, 10);**

**ListTraverse\_L(la);**

**printf("请输入需要查找的元素位置：\n");**

**scanf("%d", &loc1);**

**GetElem\_L(la, loc1, e1);**

**printf("链表中第%d个位置上的元素是：%d \n", loc1, e1);**

**printf("请输入需要插入的位置及元素值：\n");**

**scanf("%d %d", &loc2, &e2);**

**ListInsert\_L(la, loc2, e2);**

**ListTraverse\_L(la);**

**printf("请输入需要删除的位置：\n");**

**scanf("%d", &loc3);**

**ListDelete\_L(la, loc3, e3);**

**printf("删除的元素为：%d\n", e3);**

**ListTraverse\_L(la);**

**printf("\n");**

**// 合并lb,lc至ld**

**CreateList\_L(lb, 0);**

**CreateList\_L(lc, 0);**

**int i, j;**

**for (i = 1; i <= 6; i++)**

**{**

**ListInsert\_L(lb, i, i);**

**}**

**printf("待合并Lb:\n");**

**ListTraverse\_L(lb);**

**for (j = 1; j <= 4; j++)**

**{**

**ListInsert\_L(lc, j, j);**

**}**

**printf("待合并Lc:\n");**

**ListTraverse\_L(lc);**

**MergeList\_L(lb, lc, ld);**

**printf("合并后的单链表：\n");**

**ListTraverse\_L(ld);**

**}**

1. **实验结果**

**实验运行结果的截图，如main函数运行结果截图：**

****