**新疆政法学院**

**数据结构与算法**

**实验报告**

班 级： 计算机科学与技术4班

学 号： 2124030160

姓 名： 孙久猛

指导老师： 张家琪

学 期： 2022-2023第一学期

|  |  |  |  |
| --- | --- | --- | --- |
| **实验题目** | 实验二：练习链式存储中按正序、逆序查找数据 | | |
| **实验时间** | **10:00** | **实验地点** | **东训0114** |
| **实验成绩** |  | **实验性质** | **□验证性 √设计性 □综合性** |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | |
| 一、实验目的  1.理解链式存储数据间的组织方式；  2.理解正序查找链表数据的算法思想；  3.理解逆序查找链表数据的算法思想；  4、通过对比正序、逆序查找链表数据的思路不同，加深对链式存储的理解。  5、针对逆序查找，思考还有哪几种方法可以解决？强化学生对解决相同问题不同方法的思考。 | | | |
| **二、实验项目内容（实验题目）**  （一）求链表的第n个元素  1、建立链表，并输入相应数据；  2、按正序查找，输入要查找第n个元素的位数；  3、输出链表中第n个元素的值。  （二）求链表的倒数第m个元素  1、建立链表，并输入相应数据；  2、按逆序查找，输入要查找倒数第m个元素的位数；  3、输出链表中倒数第m个元素的值。  **三、源程序（实验步骤/实验过程/算法）**  //Deom\_LinkList  #include<stdio.h>  #include <stdlib.h> //为了排除()malloc()的异常  #include<math.h>  #define TRUE 1  #define FALSE 0  #define OK 1  #define ERROR 0 //定义宏变量  typedef int Status;  typedef int ElemType;  typedef struct DuLNode  {  ElemType data;  struct DuLNode \*prior,\*next;  }DuLNode,\*DuLinkList;  Status InitList(DuLinkList \*L)  { /\* 产生空的双向循环链表L \*/  \*L=(DuLinkList)malloc(sizeof(DuLNode));  if(\*L)  {  (\*L)->next=(\*L)->prior=\*L;  return OK;  }  else  return OVERFLOW;  }  int ListLength(DuLinkList L)  { /\* 初始条件：L已存在。操作结果：返回L中数据元素个数 \*/  int i=0;  DuLinkList p=L->next; /\* p指向第一个结点 \*/  while(p!=L) /\* p没到表头 \*/  {  i++;  p=p->next;  }  return i;  }  DuLinkList GetElemP(DuLinkList L,int i)  {  int j;  DuLinkList p=L;  for(j=1;j<=i;j++)  p = p->next;  return p;  }  Status ListInsert(DuLinkList L,int i,ElemType e)  {  DuLinkList p,s;  if(i<1||i>ListLength(L)+1)  return ERROR;  p=GetElemP(L,i-1);  if(!p)  return ERROR;  s=(DuLinkList)malloc(sizeof(DuLNode));  if(!s)  return OVERFLOW;  s->data=e; /\* 在第i-1个元素之后插入 \*/  s->prior=p;  s->next=p->next;  p->next->prior=s;  p->next=s;  return OK;  }  DuLinkList FindElemP( DuLinkList L, int m )  {  DuLinkList p1,p2;  int Counter;  p1 = p2 = L;  for(Counter=1 ; Counter<=m; Counter++)  p1 = p1->next;  while(p1!=L){  p1 = p1->next;  p2 = p2->next;  }  return p2;  }  void ListTraverse(DuLinkList L,void(\*visit)(ElemType))  { /\* 由双链循环线性表L的头结点出发,正序对每个数据元素调用函数visit() \*/  DuLinkList p=L->next; /\* p指向头结点 \*/  while(p!=L)  {  visit(p->data);  p=p->next;  }  printf("\n");  }  void vd(ElemType c) /\* ListTraverse()调用的函数(类型一致) \*/  {  printf("%d ",c);  }  void main()  {  DuLinkList L1, L2;  int i,n,m;  ElemType e;  InitList(&L1);//初始化L1  for(i=1;i<=10;i++)  ListInsert(L1,i,i); //在第i个结点之前插入i,正序查找备用  printf("初始化链表为：1,2,3,4,5,6,7,8,9,10\n");  printf("---- 求链表第n个元素 ----\n");  printf("输入需要查找的元素位数：");  scanf("%d",&n);  L1=GetElemP(L1,n);  if(L1)  printf("链表的第%d个元素值为%d\n",n,L1->data);  else  printf("不存在第%d个元素\n",n);  printf("---- 求链表倒数第n个元素 ----\n");  InitList(&L2);//初始化L2  for(i=1;i<=10;i++)  ListInsert(L2,i,i); // 在第i个结点之前插入i，倒序查找备用  printf("输入需要查找的倒数第m个元素的位数：");  scanf("%d",&m);  L2 = FindElemP(L2, m); //调用函数并赋值  if(L2) //判断是否存在L2  printf("链表的第%d个元素值为%d\n",m,L2->data);  else  printf("不存在第%d个元素\n",m);  return; //跳出  }  **四、运行结果** | | | |

日 期： 2022年10月7日