**中央财经大学**

**《数据结构与算法》实验报告**

**实验项目名称**  线性表的基本操作

**实 验 日 期**  2023-03-11

**姓 名**  陈博驹

**班 级**  金融工程21-2

**学 号**  2021310277

**成 绩**

|  |
| --- |
| **实验概述：** |
| **【实验目的及要求】**   1. 熟悉VC++ 开发及调试环境。 2. 掌握顺序表的定义和插入、删除、查找等基本操作的实现 3. 熟悉顺序表的简单应用，体会顺序表的优缺点 4. 掌握单链表的定义和插入、删除、查找等基本操作的实现 5. 熟悉单链表的简单应用，体会单链表的优缺点   **【实验环境】**  写出程序编写时所使用的编译器、操作系统等。  **VSC**  **Win10** |
| **实验内容：** |
| **【实验内容与实验过程及分析】**  列出实验需要做的各个题目，并针对每个题目：  1. 写出关键代码（要求选出该题目最核心的代码并加注释）；  2. 描述调试的过程、调试过程中所出现的现象及解决方法，给出题目最终运行结果；  3. 对该题目进行简短总结，写出该题目所涉及的知识点。   * **创建一个顺序表对象，并完成下列操作，把每次操作的结果显示到屏幕。**   **（1）保存下面一组数据 D={3,10,14,15,40}，**  **（2）分别查找第1、3、5、7位元素并返回**  **（3）分别在第1、3、6、7位插入新元素2023**  **（4）查找元素3、15、2013，如果存在返回第一次出现的位置，不存在给出提示并返回0.**  **（5）清空表**  **（6）删除表**  [实验运行结果]  （1）保存下面一组数据 D={3,10,14,15,40}  （2）分别查找第1、3、5、7位元素并返回          （3）分别在第1、3、6、7位插入新元素2023        （4）查找元素3、15、2013，如果存在返回第一次出现的位置，不存在给出提示并返回0.      （5）清空表    （6）删除表    [实验总结(题目所涉及的知识点)]   * 顺序表的初始化 * 顺序表类型的定义 * 顺序表的创建，从屏幕上依次读取数据生成顺序表 * 输出顺序表 * 顺序表的插入，以及插入的几种情况 * 顺序表的删除，以及删除的几种情况 * 按位查询顺序表中的元素 * 清空顺序表 * 检查顺序表是否是空表 * 求得顺序表的长度 * 销毁顺序表 * 顺序表的原地逆置 * 升序的顺序表中插入元素后仍保持升序 * 寻找中间元素的位序 * **创建一个单链表对象，并完成下列操作，把每次操作的结果显示到屏幕。**   **（1）保存下面一组数据 D={3,10,14,15,40}，**  **（2）分别查找第1、3、5、7位元素并返回**  **（3）分别在第1、3、6、7位插入新元素2023**  **（4）查找元素3、15、2013，如果存在返回第一次出现的位置，不存在给出提示并返回0.**  **（5）清空表**  **（6）删除表**  [实验运行结果]  （1）保存下面一组数据 D={3,10,14,15,40}    （2）分别查找第1、3、5、7位元素并返回          （3）分别在第1、3、6、7位插入新元素2023            （4）查找元素3、15、2013，如果存在返回第一次出现的位置，不存在给出提示并返回0        （5）清空表    （6）删除表    [实验总结(题目所涉及的知识点)]   * 单链表结点的类定义 * 单链表的初始化 * 输出带头结点的单链表 * 判断单链表是否为空 * 获取单链表的长度 * 按位查找 * 按值查找 * 单链表的销毁 * 单链表的清空 * 元素的插入及其多种情况 * 删除指定结点及其多种情况 * 尾插法建立单链表 * 头插法建立单链表 * 单链表的原地逆置 * 升序单链表插入元素后继续保持升序 * 中间位置结点的查询   **3、实现顺序表的原地转置，并分析算法的时间复杂度。**  **注：原地转置即空间复杂度为O(1)。**  [实验代码（不写主函数）]  Status ListReverse(Sqlist &L)  {  int i,j,n,t;  n=L.length;  if(n==0||n==1)  {  return OK;  }  i=0,j=n-1;  while(i<j){  t=L.data[i];  L.data[i] = L.data[j];  L.data[j] = t;  i++;  j--;  }  } [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 先声明四个整型变量i,j,n,t * 令n等于顺序表的长度，如果n等于0或者1那么顺序表就不需要逆置 * 令i=0，j=n-1，即i代表第一个元素，j代表最后一个元素 * 然后用一个while循环，当i＜j的时候，用临时的整型变量t存储data[i]的值，再把data[j]的值赋给data[i]，然后把t的值赋给data[j]，i自增，j自减，就完成了一次循环 * 时间复杂度：由于有循环结构，时间复杂度为O(n/2)即O(n) * 空间复杂度：有4个辅助变量故空间复杂度为O(1)   **4. 递增有序的顺序表，实现插入新元素e后仍有序，并分析时间和空间复杂度。**  [实验代码（不写主函数）]  Status OrderInsert(Sqlist &L,ElemType e)  {  int i;  for(i=L.length-1;i>=0&&L.data[i]>e;i--){  L.data[i+1]=L.data[i];  L.data[i]=e;}  L.length++;  }  [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 声明一个整型变量i * 写一个for循环，是i等于顺序表的最后一个元素 * i的控制条件是i＞0并且升序表顺序表的元素大于所给的参数e * 最后顺序表的长度增加 * 时间复杂度：最坏情况下的时间度，要把整个顺序表遍历一遍所以时间复杂度为O(n) * 空间复杂度：整个过程中产生了一个辅助变量，所以空间复杂度为O(1)   **5.求出顺序表正中元素的位序，分析算法的时间和空间复杂度。**  [实验代码（不写主函数）]  int Pos\_Mid(Sqlist L,int &i)  {  if((L.length)%2==1)  {  i=L.data[(L.length-1)/2];  return 1;  }  else if((L.length)%2==0)  {  i=L.data[(L.length/2)-1];  return 2;  }  }  [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 对顺序表L的表长模2，若结果为1，则i等于位序为表长减去1后除以2的元素；若结果为0，则i等于表长除以2后再减去1元素。 * 时间复杂度：由于整个过程没有循环结构故时间复杂度为O(1) * 空间复杂度：整个过程没有辅助变量故空间复杂度为O(1)   **6.实现单链表的原地转置，并分析算法的时间和空间复杂度。**  [实验代码（不写主函数）]  Node\* ListReverse(Linklist &L) /\* 思路：逐个反向。从第一个结点开始逐个反向，直到最后一个结点完成。\*/  {  Node \*p;  p = L->next;  L->next = NULL;  while(p != NULL){  Node \*tmp;  tmp = p->next;  p->next = L->next;  L->next = p;  p = tmp;  }  return L;  }  [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 先声明一个Node类型的指针p，对p进行赋值，使得p=L->next,而又使L->指向NULL，之后写一个while循环当p指针不指向空时进入循环，声明一个Node类型指针变量tmp用于临时存储数据，tmp为p->next,p->next赋值为L->next,L->next赋值为p，p赋值为tmp，由此完成了指向的反转。 * 时间复杂度：由于需要进行n次指针方向的转换，故时间复杂度为O(n) * 空间复杂度：由于每次循环都要产生一个tmp指针，所以空间复杂度为O(1)   **7.求出单链表正中元素的位序，分析算法的时间和空间复杂度。**  [实验代码（不写主函数）]  int Pos\_Mid (Linklist L,int &i) /\* 未知长度链表的中间结点定位，计算中间结点位序用i返回,并返回链表结点数量，1为奇数，0为偶数 \*/  {  int l;  l=GetLength(L);  if(l%2==0)  {  cout<<"the linklist has even elements"<<endl;  i=l/2;  return 0;  }  else if(l%2==1)  {  i=(l+1)/2;  cout<<"the linklist has odd elements"<<endl;  return 1;  }  }  [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 声明一个整型变量l，使得l存储单链表的长度，如果l模2余0，返回0，并且把中间点位的位序返回给i，并且说明单链表有偶数个元素，如果l模2余1，返回1，并且把中间点位的位序返回给i，并且说明单链表有奇数个元素。 * 时间复杂度：在该过程中没有循环结构，所以时间复杂度与n无关，时间复杂度为O(n) * 空间复杂度：该过程中一共产生了一个辅助变量，空间复杂度为O(1)   **8.递增有序的单链表，实现插入新元素e后仍有序，并分析时间和空间复杂度。**  [实验代码]  Node \* OrderInsert(Linklist &L,ElemType e) /\* 将e插入升序排列的链表中，并保持顺序关系,链表带有头节点 \*/  {  Node \*per=L;  Node \*cur=L->next;  while(cur->next!=NULL&&cur->data<e)  {  prev=cur;  cur=cur->next;  }  Node \*newNode = new Node;  prev->next = newNode;  newNode->next = cur;  newNode->data = e;  return L;  }  [实验运行结果]    [实验总结(题目所涉及的知识点)]   * 声明一个新结点prev，并把头指针L赋值给它。声明一个新结点cur，并把L->next赋值给它，当cur的指向不为空并且cur的数据域小于e，之后写一个while循环，使得prev和cur分别往后移一位。然后new一个指针结点newNode，使得newNode的数据域来存储e，并且插入单链表。 * 时间复杂度：在最坏情况下，要插入的元素e比链表中所有元素都更大，那么循环要进行n次，故时间复杂度为O(n) * 空间复杂度：该过程中辅助变量有3个，所以空间复杂度为O(1) |
| **附：编译器错误信息** |
|  |