山东大学 软件学院 学院

软件测试 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201800301202 | 姓名：李成 | | 班级： 2018级软件4班 |
| 实验题目：线性表操作 | | | |
| 实验学时：4 | | 实验日期2019-11-05 | |
| 实验目的:  1、掌握线性表的基本操作：插入、删除、查找。  2、掌握链表遍历器的使用方法。 | | | |
| 硬件环境：  1、笔记本PC掌握递归的实现思想。 | | | |
| 软件环境：   1. Windows10 2. Eclipse 3. Java | | | |
| 实验步骤与内容：  实验内容：  1、输入 n 个不为零的整数作为节点元素值，遇到 0 代表输入结束（不创 建元素值为 0 的节点），创建有序链表。输出整个链表。  2、输入一个整数，将该数插入到有序表相应位置。输出整个链表。  3、输入一个整数，在链表中进行搜索，输出其在链表中的位置。如果不 存在输出 0。  4、再一次输入一个整数，在链表中进行搜索，输出其在链表中的位置。 如果不存在输出 0。  5、再一次输入 n 个不为零的整数作为节点元素值，遇到 0 代表输入结束 （不创建元素值为 0 的节点），创建有序链表。输出整个链表。  6、使用链表遍历器实现上面两个有序链表的合并，输出合并后的有序链  表。  7、提示：注意单节点链表的测试。  实验步骤：   1. 先创建一个节点类Node类   **class Node{**  **public:**  **int key;**  **Node\* next;**  **Node(int k , Node\* n){**  **key = k;**  **this->next = n;**  **}**  **};**   1. 然后再创建一个链表类，里面有链表的各种方法以及属性。   主要的有插入insert，insert 就是进行比较，直到找到合适的位置，通过  p->next = new Node(k,p->next);  来实现插入操作。  void insert(int k){    if(firstNode == NULL){    this->firstNode = new Node(k,NULL);    }  else{    if(k<firstNode->key){    firstNode = new Node(k,firstNode);  }    else{    p = firstNode;    while(p->next!=NULL&&k>p->next->key)    p = p->next;    p->next = new Node(k,p->next);    }  }    }   1. 然后是链表的输出操作   void output(){    p = firstNode;    while( p != NULL&&p->next!=NULL){    cout<<p->key<<",";    p = p->next;    }    cout<<p->key<<endl;  }   1. 查找方法，find 主要就是通过比较值来定位，看有没有。   int find(int k){    int number=1;    p = firstNode;    while(p!=NULL&&p->key!=k){    p = p->next;    number++;  }    if(p!=NULL) return number;  else return 0 ;    }  5. main方法主要就是为了实现检测如下。  int main(){    LinkedList List ;    int k;    cout<<"input1"<<endl;    cin>>k;    while(k!=0){    List.insert(k);    cin>>k;  }    cout<<"output1"<<endl;  List.output();    cout<<"input2"<<endl;    cin>>k;    if(k!=0)    List.insert(k);    cout<<"output2"<<endl;    List.output();    cout<<"input3"<<endl;    cin>>k;    cout<<"output3"<<endl;    cout<<List.find(k)<<endl;    cout<<"input4"<<endl;    cin>>k;    cout<<"output4"<<endl;    cout<<List.find(k)<<endl;  LinkedList List2 ;    cout<<"input5"<<endl;    cin>>k;    while(k!=0){    List2.insert(k);    cin>>k;  }    cout<<"output5"<<endl;  List2.output();  if(List.first()>List2.first()){    List.combine(List2);    List2.output();    }  else{    List2.combine(List);  List.output();  }    cout<<"End"<<endl;    } | | | |
| 结论分析与体会：  基本实现了链表的操作，排序，查找等等方法。让我学会了在C++中的链表操作。熟悉了量表的各种参数。 | | | |