山东大学 软件 学院

数据结构 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202000300125 | 姓名：贾星宇 | | 班级：2020级5班 |
| 实验题目：排序算法 | | | |
| 实验学时：4 | | 实验日期：2021年10月26日星期二 | |
| 实验目的：  1、 掌握有序链表的基本操作：插入、删除、查找。  2、 掌握链表遍历器的使用方法。 | | | |
| 硬件环境：  lntel(R) Core(TM) i5-10210U CPU @ 1.60G Hz 2 .1 1 G Hz | | | |
| 软件环境：  CLion 2020.3.3 x64 | | | |
| 实验步骤与内容：  掌握三种排序方法的实现思想：插入排序、冒泡排序、基数排序：  1、 最多接受20个不为零的正整数进行排序，如果中间输入0则代表提前结束输入，0之前输入几个数就用几个数参与排序，0不参与排序。  2、 数字选择排序方法，1-Bubble Sort,2-Insert Sort,3-Radix Sort(注意大小写要区分)。  3、 基数排序能够仅仅实现小于10的正整数的排序。如果输入的数据有大于9数据，基数排序不再排序，直接输出一个0后结束程序。  4、 使用所选排序方法的排序，结果输出所用方法以及结果，每个数之间用“，”隔开，中间不要有空格。  5、 输入输出请严格按要求的格式实现，不能少任何一行文字。 | | | |
| 结论分析与体会：   1. 冒泡排序的核心思想是从头开始对length个元素依次进行比较，把大的向后移，一次循环后最大的元素放到了最后面。之后再对length-1个元素依次比较，大的元素放后面，最后第二大的元素放到了倒数第二的位置…… 2. 插入排序的思路是如果一开始只有一个元素，则将之放到数组里；假设前n个元素已经按顺序排好，则将第n+1个元素放入时，从后往前依次比较，如果比它大，则将大的元素后移，把应该插入的元素放到未移之前的位置，直到比较到比它小的元素，停止交换。 3. 箱子排序的思路是：比如要求有0~9 10个元素，则建立10个箱子，每读取一个元素，则将这个元素放到对应箱子编号的箱子里，最后再将这十个箱子连接起来。 4. 如果是基数排序，则是把该数拆分成不同的数位，然后每个数位由高到低箱子排序，然后排序次数由数位决定。 | | | |

**本次实验主要代码：**

void givRes(int arr[],int len){

for (int i = 0; i < len; ++i) {

if (i==(len-1))cout << arr[i]<<endl;

else cout<<arr[i]<<",";

}

}

void bubbleSort(int arr[],int len){

for (int i = len-1; i > 0; --i) {

for (int j = 0; j < i; ++j) {

if (arr[j]>arr[j+1])swap(arr[j],arr[j+1]);

}

}

givRes(arr,len);

}

void insert(int arrays[],int len,int insertEle){

int i;

for (i = len-1; i >=0 && insertEle<arrays[i]; i--) {

arrays[i+1] = arrays[i];

}

arrays[i+1] = insertEle;

}

void insertSort(int arr[],int len){

for (int i = 1; i <len; ++i) {

insert(arr,i,arr[i]);

}

givRes(arr,len);

}

struct node{

int ele;

node \* next = NULL;

node(int ele,node \*next){

this->ele = ele;

this->next = next;

}

};

class chain{

public:

node \*firstNode;

chain(){

firstNode = NULL;

}

void addFromHead(int ele){

firstNode = new node(ele,firstNode);

}

};

void giveResByBox(chain cha){

node \*fNode = cha.firstNode;

node \*pre;

while (fNode->next!=NULL){

cout << fNode->ele <<",";

fNode = fNode->next;

}

cout << fNode->ele<<endl;

}

node \* getTail(chain cha){

if (cha.firstNode==NULL)return cha.firstNode;

node \*result = cha.firstNode;

while (result->next !=NULL)result = result->next;

return result;

}

void radixSort(int arr[],int len){

for (int i = 0; i < len; ++i) {

if (arr[i]>9){

cout << 0 << endl;

return;

}

}

chain boxs[10];

chain res;

for (int i = 0; i < len; ++i) {

boxs[arr[i]].addFromHead(arr[i]);//顺序：addFromTail

}

for (int i = 0; i < 10; ++i) {

if (boxs[i].firstNode == NULL) continue;

else if (getTail(res)==NULL)res.firstNode = boxs[i].firstNode;

else {

getTail(res)->next = boxs[i].firstNode;

}

}

giveResByBox(res);

}

int main() {

int arrays[20];

int enter,number = 0;

cout << "Input" << endl;

cin >> enter;

for (int i = 0; enter!=0&&i<20; ++i) {

arrays[i] = enter;

number++;

if (i<19)cin >> enter;

}

cout <<"1-Bubble Sort,2-Insert Sort,3-Radix Sort"<<endl;

int choice;

cin >> choice;

cout << "Output" <<endl;

if (choice==1){

cout << "Bubble Sort" <<endl;

bubbleSort(arrays,number);

}

else if (choice==2){

cout << "Insert Sort" <<endl;

insertSort(arrays,number);

}

else if (choice==3){

cout << "Radix Sort" <<endl;

radixSort(arrays,number);

}

cout << "End0" << endl;

return 0;

}