数据结构大作业报告

课程号： 学号： 姓名：

|  |
| --- |
| 1. 作业题目   对一个学生成绩表实现添加、修改、删除和排序和查找的功能  要求： （1）分析需求，写出用到的数据结构；  （2）对排序（如冒泡法，其他快速算法）和查找（如顺序查找，二分查找）写出算法描述；  （3）排序按分数由高到低，查找按姓名； |
| 1. 实验内容 2. 由于学生成绩的统计涉及较多的数据，故选择的数据结构为链表。   2.选用带头结点的双链表存储学生信息。头节点存储链表的基本信息，如长度，尾指针等，且头节点省去了排序、删除等操作对第一个结点的特殊操作；双链表访问前一个结点较方便可以简化排序操作。    3.链表中每个结构体包含五个成员，分别为左右指针、学生姓名、学生学号、学生成绩。  代码如下：  typedef struct node  {  struct node \*lnext; //在头节点时当作尾结点指针  char student\_number[20];  char name[10];  int scores; //在头节点当作链表长度  struct node \*rnext;  }node;  其中，链表的头节点的左指针存储链表尾结点的地址，头节点的学生成绩存储链表长度。  4.主程序通过调用7个函数，分别实现链表的创建，排序，查找，添加，删除，修改，遍历操作。函数如下：  void main()  {  char index=1;  linklist head;  while(index)  {  printf("请输入要进行的操作 1.创建 2.排序 3.根据姓名查找 4.添加 5.删除 6.修改 7.遍历 0.退出\n");  scanf("%d",&index);  if(index==0) break;  switch(index)  {  case 1: head=creat(); break;  case 2: sort(head); break;  case 3: output(head); break;  case 4: add(head); break;  case 5: shanchu(head); break;  case 6: revamped(head); break;  case 7: bianli(head); break;  }  }  }  5.排序算法采用冒泡排序法，从尾指针开始依次将最大的数交换至前面，若无交换则排序完成。具体代码在void sort(linklist head)中；  查找算法采用顺序查找依次遍历每个结点直到找到符合条件的结点。具体代码在linklist output(linklist head)函数中。 |
| 1. 程序   //1900200920  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  typedef struct node  {  struct node \*lnext; //在头节点时当作尾结点指针  char student\_number[20];  char name[10];  int scores; //在头节点当作链表长度  struct node \*rnext;  }node;  typedef node \*linklist;  linklist creat();  void add(linklist );  void bianli(linklist );  void shanchu(linklist);  void sort(linklist);  void output(linklist );  void revamped(linklist );  linklist find(linklist, char\* );  void main()  {  char index=1;  linklist head;  while(index)  {  printf("请输入要进行的操作:1.创建 2.排序 3.根据姓名查找 4.添加 5.删除 6.修改 7.遍历 0.退出\n");  scanf("%d",&index);  if(index==0) break;  switch(index)  {  case 1: head=creat(); break;  case 2: sort(head); break;  case 3: output(head); break;  case 4: add(head); break;  case 5: shanchu(head); break;  case 6: revamped(head); break;  case 7: bianli(head); break;  }  }  }  linklist creat() //创建双链表  {  linklist head;  int i,size;  head = (linklist)malloc(sizeof(node));  head->lnext = head; //头节点的左指针指向左右一个节点  head->rnext = NULL;  head->scores = 0;  strcpy(head->name, "tou");  strcpy(head->student\_number, "tou"); //对头节点定义  printf("输入学生个数\n");  scanf("%d",&size);  for(i=0; i<size; i++)  add(head);  return head;  }  void add(linklist head) //链表末尾增加一个元素  {  int length = head->scores++;  linklist p;  p = head->lnext;  p->rnext = (linklist)malloc(sizeof(node));  p->rnext->lnext = p;  p = p->rnext; //令最后一结构体的右指针指向自己  p->rnext = NULL;  head->lnext = p;  printf("输入学生的学号:");  scanf("%s",p->student\_number);  printf("输入学生的名字");  scanf("%s",p->name);  printf("输入学生的分数");  scanf("%d",&p->scores);  }  void bianli(linklist head)  {  linklist p;  p = head;  while(p)  {  if(strcmp(p->name, "tou") && strcmp(p->student\_number, "tou"))  printf("学生姓名为：%s 学生学号为：%s 学生成绩为：%d \n",p->name,p->student\_number,p->scores);  else  printf("学生个数为：%d\n",p->scores);  p = p->rnext;  }  }  void shanchu(linklist head) //删除元素  {  char name[10];  linklist p;  printf("输入要删除的学生的姓名：");  scanf("%s",name);    p = find(head, name);  if(p)  {  head->scores--;  p->lnext->rnext = p->rnext;  if(p->rnext)  p->rnext->lnext = p->lnext;  else  head->lnext = p->lnext;  free(p);  }  else  printf("未找到该生信息");  }  void sort(linklist head) //链表排序  {  char flag = 1;  linklist n, p, temp; //n代表排完了n以前的节点  n = head->rnext;  while(n && flag)  {  flag = 0;  p = head->lnext;  while(p!=n && p->rnext!=n)  {  if(p->lnext->scores < p->scores)  {  flag = 1;  temp = p->lnext;  temp->lnext->rnext = p;  if(p == head->lnext)  head->lnext = temp;  else  p->rnext->lnext = temp;  temp->rnext = p->rnext;  p->lnext = temp->lnext;  temp->lnext = p;  p->rnext = temp;  }  else  p = p->lnext;  }  if(p == n) //p!=n说明n未发生交换需要右移  n = n->rnext;  }  printf("已完成排序\n");  }  void output(linklist head) //查找并输出  {  char name[10];  linklist p;  printf("输入要查找的学生的姓名：");  scanf("%s",name);  p = find(head, name);  if(p)  printf("学生姓名为：%s\n学生学号为：%s\n学生成绩为：%d\n",p->name,p->student\_number,p->scores);  else  printf("未找到该生信息\n");  }  void revamped(linklist head) //根据名字修改  {  char name[10], i;  linklist p;  printf("输入要修改信息的学生的姓名：");  scanf("%s",name);  p = find(head, name);  if(p)  {  printf("1.修改姓名 2.修改学号 3.修改分数\n");  scanf("%d", &i);  switch(i)  {  case 1: scanf("%s", p->name); break;  case 2: scanf("%s", p->student\_number); break;  case 3: scanf("%d", &p->scores); break;  }  }  else  printf("未找到该生信息\n");  }  linklist find(linklist head, char \*name) //根据名字查找  {  linklist p;  p = head->rnext;  while(strcmp(p->name, name) && p)  p = p->rnext;  return p;  } |
| 1. 运行结果 |