**链表建立与排序参考资料**

# 2.程序中的新技术

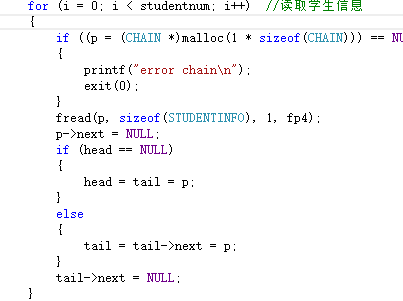
## 2.1.链表

### 2.1.1.对链表的理解

链表可看作一种特殊的结构体，它在包含储存普通信息的成员的基础上增加了指向下一块储存成员信息的成员。各块空间首尾连接构成了链表。使用链表结构可以克服数组链表需要预先知道数据大小的缺点，链表结构可以充分利用计算机内存空间，实现灵活的内存动态管理。但是链表失去了数组随机读取的优点，同时链表由于增加了一个成员，空间开销比较大。

### 2.1.2.创建链表

链表的定义如图，在包含普通学生信息的基础上增加CHAIN型的指针，用于指向下一块空间。创建链表前首先定义CHAIN型的\*head，\*tail，\*p。并指向NULL，防止出现野指针。

创建链表时，进行循环，循环次数由学生个数决定。

循环内首先为\*p分配CHAIN型空间，随后将二进制文件中的学生信息读入\*p中。此时注意读入信息的大小为STUDENTINFO型并不是CHAIN型，因为最后一个成员是用来储存下一块空间的位置。 图2-1

随后，如果\*head==NULL，即这是第一个学生信息，则令head、tail均指向p所代表的空间。如果\*head！==NULL，则令tail->next指向p所代表的空间，并将tail指向下一块空间即tail=tail->next。

循环结束后链表便创建完成，head指向链表的头部，tail指向链表的尾部。

### 2.1.3遍历链表

在进行成绩查询时，需要对链表进行遍历，遍历的方法是从head开始读取一个信息后将head的指向更新到下一位置，即head=head->next。

### 2.1.4.链表排序

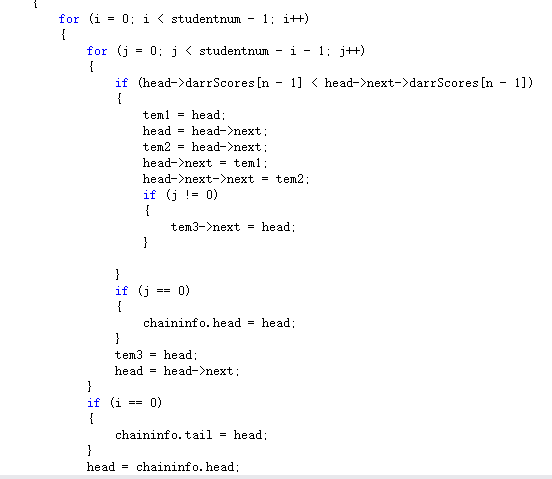
链表的排序不同于数组的排序，每次交换需要涉及4个节点的处理。使用冒泡法进行排序的程序如图

图2-2

参考函数：

void chain(int m)

{

FILE \*fp4, \*fp5;

int i, j, studentnum, classno, n;

double lb, up;

char classname[20];

CHAIN \*head, \*tail, \*p, \*tem1, \*tem2, \*tem3;

CHAININFO chaininfo;

chaininfo.head = chaininfo.tail = NULL;

head = tail = p = NULL;

if ((fp4 = fopen("16281243\_张雪欢\_Lab3.dat", "r")) == NULL) //打开二进制文件

{

printf("error \n");

exit(0);

}

fread(&classno, sizeof(int), 1, fp4); //读取基础信息

fread(classname, sizeof(char), 20, fp4);

fread(&studentnum, sizeof(int), 1, fp4);

fseek(fp4, 516, 1);

for (i = 0; i < studentnum; i++) //读取学生信息

{

if ((p = (CHAIN \*)malloc(1 \* sizeof(CHAIN))) == NULL)

{

printf("error chain\n");

exit(0);

}

fread(p, sizeof(STUDENTINFO), 1, fp4);

p->next = NULL;

if (head == NULL)

{

head = tail = p;

}

else

{

tail = tail->next = p;

}

tail->next = NULL;

}

chaininfo.head = head; //将链表信息（头尾节点个数）写入

chaininfo.tail = tail;

chaininfo.number = studentnum;

fclose(fp4);

while (m != 0)

{

if (m == 4)

{

printf("创建链表\n");

}

else if (m == 5)

{

printf("\n输出链表内容\n");

printf("%d %s %d\n", classno, classname, studentnum);

for (i = 0; i < studentnum; i++)

{

printf("%s %s ", head->szID, head->szName);

printf("%.1f %.1f %.1f\n", head->darrScores[0], head->darrScores[1], head->darrScores[2]);

head = head->next;

}

head = chaininfo.head;

printf("\n\n");

}

else if (m == 6)

{

printf("\n现在查询课程分数\n请输入要查询的课程编号：1、2、3\n输入0退出\n");

scanf("%d", &n);

while (n != 0)

{

printf("现在输入低分和高分\n");

scanf("%lf%lf", &lb, &up);

for (i = 0; i < studentnum; i++)

{

if (head->darrScores[n - 1] >= lb&&head->darrScores[n - 1] <= up)

{

printf("%s %s %.1f\n", head->szID, head->szName, head->darrScores[n - 1]);

}

head = head->next;

}

printf("\n现在查询课程分数\n请输入要查询的课程编号：1、2、3\n输入0退出\n");

scanf("%d", &n);

}

head = chaininfo.head;

printf("\n\n");

}

else

{

printf("\n现在对成绩排序\n请输入要排序的课程编号：1、2、3\n输入0退出\n");

scanf("%d", &n);

while (n != 0)

{

for (i = 0; i < studentnum - 1; i++)

{

for (j = 0; j < studentnum - i - 1; j++)

{

if (head->darrScores[n - 1] < head->next->darrScores[n - 1])

{

tem1 = head;

head = head->next;

tem2 = head->next;

head->next = tem1;

head->next->next = tem2;

if (j != 0)

{

tem3->next = head;

}

}

if (j == 0)

{

chaininfo.head = head;

}

tem3 = head;

head = head->next;

}

if (i == 0)

{

chaininfo.tail = head;

}

head = chaininfo.head;

}

printf("排后的顺序\n");

for (i = 0; i < studentnum; i++)

{

printf("%s %s ", head->szID, head->szName);

printf("%.1f %.1f %.1f\n", head->darrScores[0], head->darrScores[1], head->darrScores[2]);

head = head->next;

}

head = chaininfo.head;

if ((fp5 = fopen("chain result.txt", "a+")) == NULL)

{

printf("error chaintxt\n");

exit(1);

}

printf("按照%d排序\n", n);

fprintf(fp5, "%d %s %d\n", classno, classname, studentnum);

for (i = 0; i < studentnum; i++)

{

fprintf(fp5, "%s %s ", head->szID, head->szName);

fprintf(fp5, "%.1f %.1f %.1f\n", head->darrScores[0], head->darrScores[1], head->darrScores[2]);

head = head->next;

}

fprintf(fp5, "按照%d排序\n", n);

printf("\n现在对成绩排序\n请输入要排序的课程编号：1、2、3\n输入0退出\n");

scanf("%d", &n);

head = chaininfo.head;

}

}

printf("请选择进行的操作\n1:读取并以二进制写入\n2:结构体数组读入二进制数据并查找需要的成绩\n");

printf("3:指针数组读入二进制数据并查找需要的成绩\n4:创建链表\n5:输出链表内容\n");

printf("6:对链表查找\n7:对链表进行排序\n0：退出\n\n");

scanf("%d", &m);

printf("\n\n");

}

}