

# 《数据结构与算法分析》

## 实验报告

|      |              |
|------|--------------|
| 实验名称 | 基于线性表的图书管理系统 |
| 姓名   | 曾庆文          |
| 学号   | 23060218     |
| 院系   | 自动化学院        |
| 专业   | 人工智能         |
| 班级   | 23061011     |
| 实验时间 | 2024.3       |

# 一、实验目的

- 1. 掌握线性表的顺序存储表示和链式存储表示。
- 2. 掌握顺序表和链表的基本操作，包括创建、查找、插入和删除等算法。
- 3. 明确线性表两种不同存储结构的特点及其适用场合，明确它们各自的优缺点。

# 二、实验设备与环境

编辑器：Visual Studio Code  
编译器：gcc

# 三、实验内容与结果

每一项基本操作的输入输出结果如下：

- 1. 基于顺序存储结构的图书信息表的创建和输出

```
请输入数字以选择以下操作：
0.退出系统
1.创建和输出
2.修改图书信息
3.逆序存储
4.查找最贵图书
5.新图书入库
6.旧图书出库
7.图书去重
1
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理及应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00
0 0 0
```

输入

```
6
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理及应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00
```

输出

```
请按任意键继续 . . . |
```

- 2. 基于顺序存储结构的图书信息表的修改

请输入数字以选择以下操作：

- 0.退出系统
- 1.创建和输出
- 2.修改图书信息
- 3.逆序存储
- 4.查找最贵图书
- 5.新图书入库
- 6.旧图书出库
- 7.图书去重

2

9787302257646 程序设计基础 25.00

9787302164340 程序设计基础（第2版） 20.00

9787302219972 单片机技术及应用 32.00

9787302203513 单片机原理及应用技术 26.00

9787810827430 工业计算机控制技术-原理与应用 29.00

9787811234923 汇编语言程序设计教程 21.00

0 0 0

25.50

9787302257646 程序设计基础 30.00

9787302164340 程序设计基础（第2版） 24.00

9787302219972 单片机技术及应用 35.20

9787302203513 单片机原理及应用技术 28.60

9787810827430 工业计算机控制技术-原理与应用 31.90

9787811234923 汇编语言程序设计教程 25.20

请按任意键继续 . . . |

输入

输出

### 3. 基于顺序存储结构的图书信息表的逆序存储

请输入数字以选择以下操作：

- 0.退出系统
- 1.创建和输出
- 2.修改图书信息
- 3.逆序存储
- 4.查找最贵图书
- 5.新图书入库
- 6.旧图书出库
- 7.图书去重

3

6

9787302257646 程序设计基础 25.00

9787302164340 程序设计基础（第2版） 20.00

9787302219972 单片机技术及应用 32.00

9787302203513 单片机原理及应用技术 26.00

9787810827430 工业计算机控制技术-原理与应用 29.00

9787811234923 汇编语言程序设计教程 21.00

9787811234923 汇编语言程序设计教程 21.00

9787810827430 工业计算机控制技术-原理与应用 29.00

9787302203513 单片机原理及应用技术 26.00

9787302219972 单片机技术及应用 32.00

9787302164340 程序设计基础（第2版） 20.00

9787302257646 程序设计基础 25.00

请按任意键继续 . . . |

输入

输出

### 4. 基于顺序存储结构的图书信息表的最贵图书的查找

```
请输入数字以选择以下操作：
0.退出系统
1.创建和输出
2.修改图书信息
3.逆序存储
4.查找最贵图书
5.新图书入库
6.旧图书出库
7.图书去重
4
6
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理与应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811230710 C# 程序设计易懂易会教程 32.00
2
9787302219972 单片机技术及应用 32.00
9787811230710 C# 程序设计易懂易会教程 32.00
请按任意键继续 . . . |
```

输入

输出

## 5. 基于顺序存储结构的图书信息表的新图书的入库

```
请输入数字以选择以下操作：
0.退出系统
1.创建和输出
2.修改图书信息
3.逆序存储
4.查找最贵图书
5.新图书入库
6.旧图书出库
7.图书去重
5
6
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理与应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00
2
9787302265436 计算机导论实验指导 18.00
9787302257646 程序设计基础 25.00
9787302265436 计算机导论实验指导 18.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理与应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00
请按任意键继续 . . . |
```

输入

输出

## 6. 基于顺序存储结构的图书信息表的旧图书的入库

```
请输入数字以选择以下操作：
0.退出系统
1.创建和输出
2.修改图书信息
3.逆序存储
4.查找最贵图书
5.新图书入库
6.旧图书出库
7.图书去重

6
6
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理与应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00
2
9787302257646 程序设计基础 25.00
9787302219972 单片机技术及应用 32.00
9787302203513 单片机原理与应用技术 26.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787811234923 汇编语言程序设计教程 21.00

请按任意键继续 . . . |
```

输入

输出

## 7. 基于顺序存储结构的图书信息表的图书去重

```
请输入数字以选择以下操作：
0.退出系统
1.创建和输出
2.修改图书信息
3.逆序存储
4.查找最贵图书
5.新图书入库
6.旧图书出库
7.图书去重

7
6
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787302257646 程序设计基础 25.00
9787810827430 工业计算机控制技术-原理与应用 29.00
9787302219972 单片机技术及应用 32.00
4
9787302257646 程序设计基础 25.00
9787302164340 程序设计基础（第2版） 20.00
9787302219972 单片机技术及应用 32.00
9787810827430 工业计算机控制技术-原理与应用 29.00

请按任意键继续 . . . |
```

输入

输出

实验全部代码如下：

```
1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #define max 100
5.
6. //图书结构体定义
7. typedef struct Book
8. {
9.     char ISBN[20];
10.    char name[50];
11.    float price;
12. }Book;
13.
14. //八个功能函数
15. void createAndOutput(Book *book);           //创建和输出函数
16. void update(Book *book);                     //修改函数
17. void invertOrder(Book *book);                //逆序存储
18. void seekMostExpensive(Book *book);          //查找最贵图书
19. int compare(const void*a,const void*b);      //seekMostExpensive 函数的排序方式
20. void newBookIn(Book *book);                  //新图书入库
21. void oldBookOut(Book *book);                 //旧图书出库
22. void deleteRepeat(Book *book);              //图书去重
23.
24. //两个全局变量
25. int length=0;                                //记录添加的书本的数量
26. float sumPrice=0.0;                          //入库图书总价
27.
28.
29. int main()
30. {
31.     Book book[max];
32.     int handle;
33.     printf("请输入数字以选择以下操作： \n");
34.     printf("0.退出系统\n");
35.     printf("1.创建和输出\n");
36.     printf("2.修改图书信息\n");
37.     printf("3.逆序存储\n");
38.     printf("4.查找最贵图书\n");
39.     printf("5.新图书入库\n");
```

```
40.     printf("6.旧图书出库\n");
41.     printf("7.图书去重\n");
42.     scanf("%d",&handle);
43.     switch (handle){
44.         case 0:
45.             printf("您已成功退出系统\n");
46.             return 0;
47.         case 1:
48.             createAndOutput(book);
49.             break;
50.         case 2:
51.             update(book);
52.             break;
53.         case 3:
54.             invertOrder(book);
55.             break;
56.         case 4:
57.             seekMostExpensive(book);
58.             break;
59.         case 5:
60.             newBookIn(book);
61.             break;
62.         case 6:
63.             oldBookOut(book);
64.             break;
65.         case 7:
66.             deleteRepeat(book);
67.             break;
68.         default:
69.             printf("您输入的是无效数字,请输入有效数字: \n");
70.             break;
71.     }
72.     return 0;
73. }
74.
75. // 创建和输出函数
76. void createAndOutput(Book *book)
77. {
78.     int bookNumber=0;
79.     Book *temp;//暂时存放输入数据
80.     for(;length<max;length++){
81.         scanf("%s", (temp->ISBN));
82.         scanf("%s", (temp->name));
83.         scanf("%f",&(temp->price));
```

```

84.         getchar();
85.         // 终止循环
86.         if(strcmp(temp->ISBN,"0")==0 && strcmp(temp->name,"
0")==0 && temp->price==0){
87.             break;
88.         }
89.         // 暂存数据赋值给 book
90.         else{
91.             *(book+length)=*temp;
92.             bookNumber++;
93.         }
94.     }
95.     printf("%d\n",bookNumber);
96.     for(int i=0;i<bookNumber;i++){
97.         printf("%s %s %.2f", (book+i)->ISBN, (book+i)->name, (
book+i)->price);
98.         printf("\n");
99.     }
100. }
101.
102. // 修改函数
103. void update(Book *book)
104. {
105.     int bookNumber=0;
106.     Book *temp;// 暂时存放输入数据
107.     for(;length<max;length++){
108.         scanf("%s", (temp->ISBN));
109.         scanf("%s", (temp->name));
110.         scanf("%f",&(temp->price));
111.         // 终止循环
112.         if(*(temp->ISBN)=='0'&&*(temp->name)=='0'&&temp->
price==0.0){
113.             break;
114.         }
115.         // 暂存数据赋值给 book
116.         else{
117.             *(book+length)=*temp;
118.             sumPrice+=temp->price;
119.             bookNumber++;
120.         }
121.     }
122.     float avePrice=sumPrice/bookNumber;
123.     printf("%.2f\n",avePrice);
124.     for(int i=0;i<bookNumber;i++){

```



```

125.         if((book+i)->price<avePrice){
126.             (book+i)->price*=1.2;
127.             printf("%s %s %.2f", (book+i)->ISBN, (book+i)->
                name, (book+i)->price);
128.             printf("\n");
129.         }
130.     else{
131.         (book+i)->price*=1.1;
132.         printf("%s %s %.2f", (book+i)->ISBN, (book+i)->
                name, (book+i)->price);
133.         printf("\n");
134.     }
135. }
136. }
137.
138. // 逆序存储
139. void invertOrder(Book *book)
140. {
141.     int bookNumber;
142.     scanf("%d", &bookNumber);
143.     length=bookNumber;
144.     for(int i=bookNumber-1; i>=0; i--){
145.         scanf("%s%s%f", (book+i)->ISBN, (book+i)->name, &(bo
            ok+i)->price);
146.     }
147.     for(int i=0; i<bookNumber; i++){
148.         printf("%s %s %.2f\n", (book+i)->ISBN, (book+i)->na
            me, (book+i)->price);
149.     }
150. }
151.
152. // 最贵图书查找
153. void seekMostExpensive(Book *book)
154. {
155.     int bookNumber;
156.     scanf("%d", &bookNumber);
157.     Book orderedBook[bookNumber]; // 有序结构体数组
158.     length=bookNumber;
159.     for(int i=bookNumber-1; i>=0; i--){
160.         scanf("%s%s%f", (book+i)->ISBN, (book+i)->name, &(bo
            ok+i)->price);
161.         *(orderedBook+i)=*(book+i);
162.     }

```

```

163.     qsort(orderedBook,bookNumber,sizeof(Book),compare);//
        对有序结构体数组从大到小排序
164.     int mostExpensiveNum=1;
165.     // 统计最贵图书数量
166.     for(int i=0;i<bookNumber-1;i++){
167.         if((orderedBook+i)->price!=(orderedBook+i+1)->price){
168.             break;
169.         }
170.         mostExpensiveNum++;
171.     }
172.     printf("%d\n",mostExpensiveNum);
173.     for(int i=0;i<mostExpensiveNum;i++){
174.         printf("%s %s %.2f\n",(orderedBook+i)->ISBN,(orderedBook+i)->name,(orderedBook+i)->price);
175.     }
176. }
177.
178. //seekMostExpensive 函数中有序结构体数组的排序方式
179. int compare(const void*a,const void*b)
180. {
181.     return ((Book *)b)->price-((Book *)a)->price;
182. }
183.
184. //新图书入库
185. void newBookIn(Book *book)
186. {
187.     int bookNumber;
188.     scanf("%d",&bookNumber);
189.     length=bookNumber;
190.     for(int i=0;i<bookNumber;i++){
191.         scanf("%s%s%f",(book+i)->ISBN,(book+i)->name,&(book+i)->price);
192.     }
193.     Book newBook;//待入库的新图书
194.     int location;//入库位置
195.     scanf("%d",&location);
196.     scanf("%s%s%f",newBook.ISBN,newBook.name,&(newBook.price));
197.     //判断入库位置是否合法
198.     if((location<1)|| (location>length+1)|| (length==max)){
199.         printf("抱歉, 入库位置非法! \n");
200.         return 0;
201.     }

```

```

202.     for(int j=length-1;j>=location-1;j--){
203.         *(book+j+1)=*(book+j);
204.     }
205.     *(book+location-1)=newBook;
206.     length++;
207.     for(int i=0;i<length;i++){
208.         printf("%s %s %.2f\n", (book+i)->ISBN, (book+i)->na
           me, (book+i)->price);
209.     }
210. }
211.
212. //旧图书出库
213. void oldBookOut(Book *book)
214. {
215.     int bookNumber;
216.     scanf("%d",&bookNumber);
217.     length=bookNumber;
218.     for(int i=0;i<bookNumber;i++){
219.         scanf("%s%s%f", (book+i)->ISBN, (book+i)->name, &(bo
           ok+i)->price);
220.     }
221.     int location;//出库位置
222.     scanf("%d",&location);
223.     //判断出库位置是否合法
224.     if((location<1)|| (location>length)){
225.         printf("抱歉, 出库位置非法! \n");
226.         return 0;
227.     }
228.     for(int j=location;j<length;j++){
229.         *(book+j-1)=*(book+j);
230.     }
231.     length--;
232.     for(int i=0;i<length;i++){
233.         printf("%s %s %.2f\n", (book+i)->ISBN, (book+i)->na
           me, (book+i)->price);
234.     }
235. }
236.
237. //图书去重
238. void deleteRepeat(Book *book)
239. {
240.     int bookNumber;
241.     scanf("%d",&bookNumber);
242.     length=bookNumber;

```

```

243.     for(int i=0;i<bookNumber;i++){
244.         scanf("%s%s%f", (book+i)->ISBN, (book+i)->name, &(bo
           ok+i)->price);
245.     }
246.     for(int i=0;i<length;i++){
247.         for(int j = 1; j<length ; j++){
248.             if(strcmp((book+i)->ISBN, (book+j)->ISBN)==0&&i!=j){
249.                 //查重
250.                 *(book+j)=*(book+j+1); //把从第j个元素之后的前
           移
251.                 length--;
252.             }
253.         }
254.     }
255.     printf("%d\n", length);
256.     for(int i=0;i<length;i++){
257.         printf("%s %s %.2f\n", (book+i)->ISBN, (book+i)->na
           me, (book+i)->price);
258.     }
259. }

```

## 四、实验总结及体会

1. 模块化设计：一开始设计整套系统时，想把全部功能都写进 main 函数。但是随着项目的深入进行，我逐渐发现这种构架方法有很多缺陷。首先它不利于测试，因为每个部分都是有所关联的，导致测试案例有时很难进行；其次，它不利于后期维护，一开始编写还能记住每个变量的含义，但是过了几天就容易忘记，这时再想给系统添加新功能就要重新看之前的变量含义。因此，模块化设计就很重要。我将每一个功能封装进不同的函数，这样主函数调用时就非常方便。而且添加新功能时只需要在加入一个新函数即可，这极大地提高了我的开发效率。
2. 测试与 debug：我在开发的过程中经常遇到代码跑不通，或者陷入死循环的情况，这时就需要 debug 了。打断点、逐步运行，一行一行检查，才能发现代码的错误所在。否则只是盯着代码而不去调试的话很难发现错误在哪里。