# Table of Contents

[《数据结构》上机实验内容和要求](#p3)

[实验一 基于线性表的图书信息管理](#p4)

[实验二 基于栈的算术表达式求值](#p13)

[实验三 基于字符串模式匹配算法的病毒](#p17)

[感染检测问题](#p17)

[实验四 基于哈夫曼树的数据压缩算法](#p21)

[实验五 基于二叉树的表达式求值算法](#p25)

[实验六 基于Dijsktra算法的最短路径求解](#p28)

[实验七 基于广度优先搜索的六度空间](#p31)

[理论验证](#p31)

[实验八 排序算法的实现与分析](#p36)

《数据结构》实验指导书

计算机专业实验中心

2023 年 9 月

目 录

[《数据结构》上机实验内容和要求 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1](#p3)

[实验一 基于线性表的图书信息管理 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2](#p4)

[实验二 基于栈的算术表达式求值 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11](#p13)

[实验三 基于字符串模式匹配算法的病毒感染检测问题 . . . . . . . . . . . . . . . . . . . . 15](#p17)

[实验四 基于哈夫曼树的数据压缩算法 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19](#p21)

[实验五 基于二叉树的表达式求值算法 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23](#p25)

[实验六 基于 DIJSKTRA 算法的最短路径求解 . . . . . . . . . . . . . . . . . . . . . . . . . . . 26](#p28)

[实验七 基于广度优先搜索的六度空间理论验证 . . . . . . . . . . . . . . . . . . . . . . . . . . 29](#p30)

[实验八 排序算法的实现与分析 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34](#p36)

2

《数据结构》上机实验内容和要求

通过上机实验加深对课程内容的理解，提高程序设计、开发及调试能力。本

实验指导书适用于 16 学时《数据结构》实验课，实验项目具体内容如下：

序号

实验名称

每组人数

实验学时

实验类别

1

实验一 基于线性表的图书信息管理

1

2

设计

2

实验二 基于栈的算术表达式求值

1

2

设计

实验三 基于字符串模式匹配算法的病毒感

3

1

2

设计

染检测问题

4

实验四 基于哈夫曼树的数据压缩算法

1

2

设计

5

实验五 基于二叉树的表达式求值算法

1

2

设计

6

实验六 基于 Dijsktra 算法的最短路径求解

1

2

设计

实验七 基于广度优先搜索的六度空间理论

7

1

2

设计

验证

8

实验八 排序算法的实现与分析

1

2

设计

实验报告要求：

请按照评分标准和报告模板要求，提交实验报告电子版文件。

1

实验一 基于线性表的图书信息管理

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、实验内容

选用顺序表或链表实现下述线性表的的基本操作。

1. 基于顺序（链式）存储结构的图书信息表的创建和输出

问题描述

定义一个包含图书信息(书号、书名、价格)的顺序表（链表），读入相应的

图书数据来完成图书信息表的创建。然后，统计图书表中的图书个数，同时逐行

输出每本图书的信息。

输入要求

输入 n+1 行，其中前 n 行是 n 本图书的信息(书号、书名、价格)，每本图书

信息占一行，书号、书名、价格用空格分隔，价格之后没有空格。最后第 n+1

行是输入结束标志:000(空格分隔的三个 0)。其中，书号和书名为字符串类型，

价格为浮点数类型。

输出要求

总计 n+1 行，第 1 行是所创建的图书信息表中的图书个数，后 n 行是 n 本图

书的信息(书号、书名、价格)，每本图书信息占一行，书号、书名、价格用空格

分隔。其中，价格输出保留两位小数。

输入样例

9787302257646 程序设计基础 25.00

9787302164340 程序设计基础(第 2 版) 20.00

2

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

0 0 0

输出样例

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. 基于顺序（链式）存储结构的图书信息表的修改

问题描述

首先，定义一个包含图书信息(书号、书名、价格)的顺序表（链表），读入

相应的图书数据完成图信息表的创建。然后，计算所有图书的平均价格，将所有

低于平均价格的图书价格提高 20%，所有高于或等于平均价格的图书价格提高

10%。最后，逐行输出价格修改后的图书信息。

输入要求

输入 n+1 行，前 n 行是 n 本图书的信息(书号、书名、价格)，每本图书信息

占一行，书号、书名、价格用空格分隔，价格之后没有空格。最后，第 n+1 行是

输入结束标志:000(空格分隔的三个 0)。其中书号和书名为字符串类型，价格为

浮点数类型。

输出要求

3

总计 n+1 行,第 1 行是修改前所有图书的平均价格,后 n 行是价格修改后 n 本

图书的信息(书号、书名、价格)、每本图书信息占一行，书号、书名、价格用空

格分隔。其中，价格输出保留两位小数。

输入样例

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

0 0 0

输出样例

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 单片机原理与应用技术 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

3. 基于顺序（链式）存储结构的图书信息表的最贵图书的查找

问题描述

定义一个包含图书信息(书号、书名、价格)的顺序表（链表），读入相应的

图书数据来完成图书信息表的创建。然后，查找价格最高的图书，输出相应图书

的信息。

输入要求

4

总计输入 n+1 行，其中，第一行是图书数目 n，后 n 行是 n 本图书的信息(书

号、书名、价格)，每本图书信息占一行，书号、书名、价格用空格分隔，价格

之后没有空格。其中，书号和书名为字符串类型，价格为浮点数类型。

输出要求

总计输出 m+1 行，其中，第一行是最贵图书的数目(价格最高的图书可能有

多本).后 m 行是最贵图书的信息，每本图书信息占一行，书号、书名、价格用空

格分隔。其中，价格输出保留两位小数。

输入样例

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

输出样例

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

4. 基于顺序（链式）存储结构的图书信息表的新图书的入库

问题描述

首先，定义一个包含图书信息(书号、书名、价格)的顺序表（链表），读入

相应的图书数据来完成图书信息表的创建。然后，根据指定的待入库的新图书的

位置和信息，将新图书插入到图书表中指定的位置上。最后，输出新图书入库后

所有图书的信息。

输入要求

5

总计 n+3 行。首先，输入 n+1 行，其中，第一行是图书数目 n，后 n 行是 n 本图书的信息(书号、书名、价格)，每本图书信息占一行，书号、书名、价格用

空格分隔，价格之后没有空格。其中，书号和书名为字符串类型，价格为浮点数

类型。之后输入第 n+2 行，内容仅为一个整数，代表待入库的新图书的位置序号。

最后，输入第 n+3 行，内容为新图书的信息，书号、书名、价格用空格分隔。

输出要求

若插入成功：

输出新图书入库后所有图书的信息(书号、书名、价格)，总计 n+1 行，每行

是一本图书的信息，书号、书名、价格用空格分隔。其中，价格输出保留两位小

数。

若插入失败:

只输出以下提示:抱歉，入库位置非法!

输入样例

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

输出样例

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

6

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

5. 基于顺序（链式）存储结构的图书信息表的旧图书的出库

问题描述

定义一个包含图书信息(书号、书名、价格)的顺序表（链表），读入相应的

图书数据来完成图书信息表的创建。然后根据指定的待出库的旧图书的位置，将

该图书从图书表中删除。最后输出该图书出库后的所有图书的信息。

输入要求

总计 n+2 行。首先，输入 n+1 行，其中，第一行是图书数目 n，后 n 行是 n 本图书的信息(书号、书名、价格)，每本图书信息占一行，书号、书名、价格用

空格分隔，价格之后没有空格。其中书号和书名为字符串类型，价格为浮点数类

型。然后，输入第 n+2 行，内容仅为一个整数，代表待删除的旧图书的位置序

号。

输出要求

若删除成功：

输出旧图书出库后所有图书的信息(书号、书名、价格)，总计 n-1 行，每行

是一本图书的信息，书号、书名、价格用空格分隔。其中，价格输出保留两位小

数。

若删除失败：

只输出以下提示:抱歉，出库位置非法!

输入样例

6

9787302257646 程序设计基础 25.00

9787302164340 程序设计基础(第 2 版) 20.00

7

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

输出样例

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

6. 基于顺序（链式）存储结构的图书信息表的图书去重（选做）

问题描述

出版社出版的任何一本图书的书号(ISBN)都是唯一的，即图书表中不允许包

含书号重复的图书。定义一个包含图书信息(书号、书名、价格)的顺序表（链表），

读入相应的图书数据来完成图书信息表的创建(书号可能重复)。然后进行图书的

去重，即删除书号重复的图书(只保留第一本)。最后输出去重后所有图书的信息。

输入要求

总计输入 n+1 行，其中，第一行是图书数目 n，后 n 行是 n 本图书的信息(书

号、书名、价格)，每本图书信息占一行，书号、书名、价格用空格分隔，价格

之后没有空格(书号可能重复)。其中书号和书名为字符串类型，价格为浮点数类

型。

输出要求

总计输出 m+1 行(m≤n)，其中，第一行是去重后的图书数目。后 m 行是去重

后图书的信息（书号、书名、价格)，每本图书信息占一行，书号、书名、价格

用空格分隔。其中，价格输出保留两位小数。

8

输入样例

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

输出样例

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

三、实验提示

不同的存储结构对应的算法实现不同，下面分别给出图书信息表的顺序存储

结构和链式存储结构的定义。

根据图书所包含的基本信息，可以对图书信息的结构体描述如下:

#define MAXSIZE 10000 //图书表可能达到的最大长度

typedef struct //图书信息定义

{

char no[20]; //图书 ISBN

char name[50]; //图书名字

float price; //图书价格

}Book;

基于上述图书信息的描述,下面分别给出图书信息表的顺序存储结构和链式

存储结构的定义。图书信息表的顺序存储结构如下:

9

typedef struct

{

Book \*elem; //存储空间的基地址

int length; //图书表中当前图书个数

)sqlist; //图书表的顺序存储结构类型为 SqList

链式存储结构的定义如下:

typedef struct LNode

{

Book data; //结点的数据域

struct LNode \*next; /结点的指针域

}LNode,\*LinkList; //LinkList 为指向结构体 LNode 的指针类型

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

10

实验二 基于栈的算术表达式求值

一、实验目的

1.掌握栈的基本操作算法的实现，包括栈初始化、进栈、出栈、取栈顶元素

等。

2.掌握利用栈实现中缀表达式求值的算法。

二、实验内容

问题描述

输入一个中缀算术表达式，求解表达式的值。运算符包括“+”、“-”、“\*”、“/”、

“(”、“)”、“=”，参加运算的数为 double 类型且为正数。

要求:直接使用中缀算术表达式进行计算，不能转换为后缀或前缀表达式再进

行计算，只考虑二元运算即可。

输入要求

多组数据，每组数据一行，对应一个算术表达式，每个表达式均以“=”结尾。

当表达式只有一个“= ”时，输入结束。参加运算的数为 double 类型。

输出要求

对于每组数据输出 1 行，为表达式的运算结果。输出保留两位小数。

输入样例

2+2=

20\*(4.5-3)=

=

输出样例

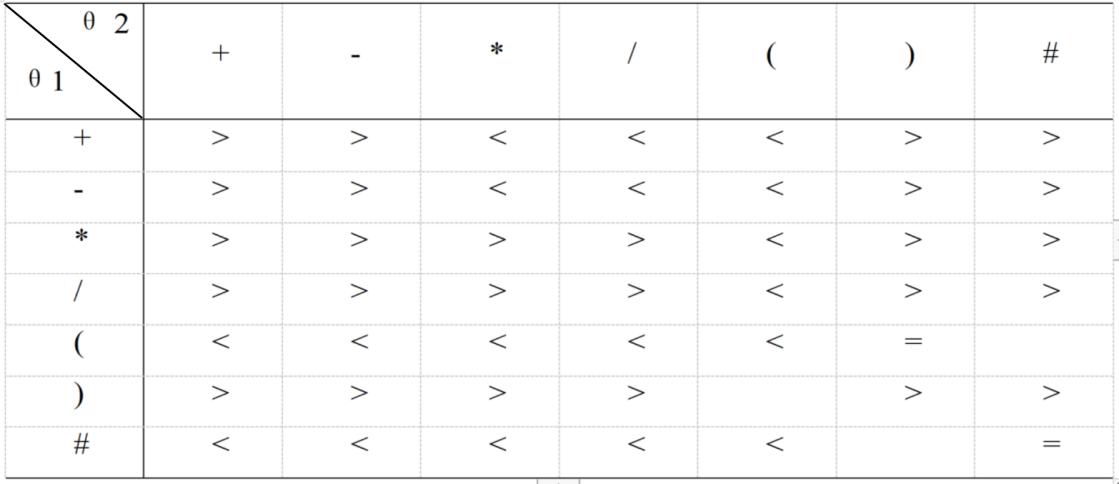
4.00

30.00

三、实验提示

此实验内容即为主教材算法 3.22 的扩展内容，算法 3.22 只考虑个位数的运

11



算，不具备拼数功能。拼数功能可以手工编写，也可以借助 C 语言的库函数 atof() 函数来完成，其功能是将字符串转换为双精度浮点数(double)。

算符间的优先关系如下表所示（表来源：严蔚敏《数据结构》）：

表 2.1 算符间的优先关系

表中需要注意的是θ1 为运算符栈 OPTR 的栈顶元素 top，θ2 为从表达式中

读取的操作符 c，此优先级表可以用二维数组实现。

算法步骤：

为了实现用栈计算算数表达式的值，需设置两个工作栈：用于存储运算符的

栈 OPTR，以及用于存储操作数及中间结果的栈 OPND。

算法基本思想如下：

（1）首先将操作数栈 OPND 设为空栈，而将'#'作为运算符栈 OPTR 的栈底

元素，这样的目的是判断表达式是否求值完毕。

（2）依次读入表达式的每个字，表达式须以'#'结，读入字符若是操作数则入

栈 OPND，读入字符若是运算符，则将此运算符 c 与 OPTR 的栈顶元素 top 比较

优先级后执行相应的操作，具体操作如下：

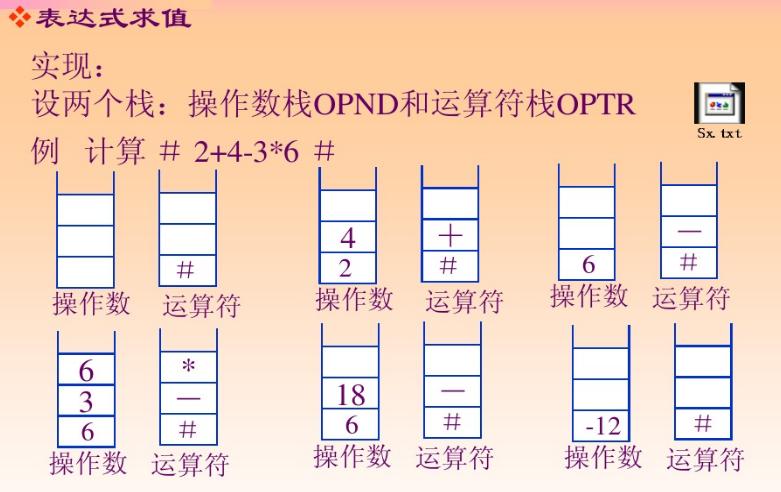
(i)若 top 的优先级小于 c，即 top<c，则将 c 直接入栈 OPTR，并读入下一字

符赋值给 c；

(ii)若 top 的优先级等于 c，即 top=c，则弹出 OPTR 的栈顶元素，并读入下一

字符赋值给 c，这一步目的是进行括号操作;

12



（iii）若 top 优先级高于 c，即 top>c，则表明可以计算，此时弹出 OPND 的

栈顶两个元素，并且弹出 OPTR 栈顶的的运算符，计算后将结果放入栈 OPND

中。直至 OPTR 的栈顶元素和当前读入的字符均为'#'，此时求值结束。

比较算符优先关系代码示例：

1. int getIndex(char theta)

//获取 theta 所对应的索引

2. {

3.

int index = 0;

4.

switch (theta)

5.

{

6.

case '+':

7.

index = 0;

8.

break;

9.

case '-':

10.

index = 1;

11.

break;

12.

case '\*':

13.

index = 2;

14.

break;

15.

case '/':

16.

index = 3;

17.

break;

18.

case '(':

19.

index = 4;

13

20.

break;

21.

case ')':

22.

index = 5;

23.

break;

24.

case '#':

25.

index = 6;

26.

default:break;

27.

}

28.

return index;

29. }

30.

31. char getPriority(char theta1, char theta2) 32. //获取 theta1 与 theta2 之间的优先级

33. {

34.

const char priority[][7] =

35. //算符间的优先级关系

36.

{

37.

{ '>','>','<','<','<','>','>' }, 38.

{ '>','>','<','<','<','>','>' }, 39.

{ '>','>','>','>','<','>','>' }, 40.

{ '>','>','>','>','<','>','>' }, 41.

{ '<','<','<','<','<','=','0' }, 42.

{ '>','>','>','>','0','>','>' }, 43.

{ '<','<','<','<','<','0','=' }, 44.

};

45.

46.

int index1 = getIndex(theta1);

47.

int index2 = getIndex(theta2);

48.

return priority[index1][index2]; 49. }

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

14

实验三 基于字符串模式匹配算法的病毒

感染检测问题

一、实验目的

1、掌握字符串的顺序存储表示方法。

2、掌握字符串模式匹配算法 BF 算法或 KMP 算法的实现。

二、实验内容

问题描述

医学研究者最近发现了某些新病毒，通过对这些病毒的分析，得知它们的

DNA 序列都是环状的。现在研究者已收集了大量的病毒 DNA 和人的 DNA 数据，

想快速检测出这些人是否感染了相应的病毒。为了方便研究，研究者将人的 DNA

和病毒 DNA 均表示成由一些字母组成的字符串序列，然后检测某种病毒 DNA 序列是否在患者的 DNA 序列中出现过，如果出现过，则此人感染了该病毒，否

则没有感染。例如，假设病毒的 DNA 序列为 baa，患者 1 的 DNA 序列为 aaabbba 则感染;患者 2 的 DNA 序列为 babbba，则未感染。

注意，人的 DNA 序列是线性的，而病毒的 DNA 序列是环状的。

输入要求

多组数据，每组数据有 1 行，为序列 A 和 B，A 对应病毒的 DNA 序列，B

对应人的 DNA 序列。A 和 B 都为“0”时输入结束。

输出要求

对于每组数据输出 1 行，若患者感染了病毒输出“YES”，否则输出“NO”。输

入样例

abbab abbabaab

baa cacdveabacsd

abe def

0 0

输出样例

15

YES

YES

NO

三、实验提示

此实验内容即要求实现主教材的案例 4.1，具体实现可参考算法 4.5，算法 4.5

是利用 BF 算法来实现字符串的模式匹配过程的，效率较低，可以利用 KMP 算

法完成模式匹配以提高算法的效率。同学们可以模仿算法 4.5，利用 KMP 算法

来完成病毒感染检测的方案。

算法 4.5 病毒感染检测

【算法步骤】

①从文件中读取待检测的任务数 num。

②根据 mm 个数依次检测每对病毒 DNA 和人的 DNA 是否匹配，循环 m 次，

执行以下操作：

·从文件中分别读取一对病毒 DNA 列和人的 DNA 列;

·设置一个标志性变量 flag，用来标识是否匹配成功，初始为 0，表示未匹配;

·病毒 DNA 列的长度是 m，将存储病毒 DNA 列的字符串长度扩大为 2m，将

病毒 DNA 序列连续存储两次;

·循环 m 次，重复执行以下操作：

>依次取得每个长度为 m 的病毒 DNA 环状字符串;

>将此字符串作为模式串，将人的 DNA 序列作为主串，调用 BF 算法进行模

式匹配，将匹配结果返回赋值给 flag;

>若 flag 非 0，表示匹配成功，中止循环，表明该人感染了对应的病毒。

·退出循环时，判断 flag 的值，若 flag 非 0，输出““YES”，否则，输出“NO”

【算法描述】

void Virus\_detection()

{//利用 BF 算法实现病毒检测

16

ifstream inFile(“病毒感染检测输人数据.txt"); ofstream outFile(“病毒感染检测输出结果.txt"); inFile>>num; //读取待检测的任务数

while(num--) //依次检测每对病毒 DNA 和人的 DNA 是否匹配

{

infile>>num;

infile>>Virus.ch+l: //读取病毒 DNA 序列，字符串从下标 1 开始存放

inFile>>Person.ch+1: //读取人的 DNA 序列

Vir=Virus.ch; //将病毒 DNA 临时暂存在 Vir 中，以备输出

flag=0; //用来标识是否匹配，初始为 0，匹配后为非 0

m=Virus.length; //病毒 DNA 序列的长度是 m

for(i=m+1,j=1;j<=m;j++)

Virus.ch[i++]=Virus.ch[j]; //将病毒字符串的长度扩大 2 倍

Virus.ch[2\*m+1]="\0'; //添加结束符号

for(i=0;i<m;i++) //依次取得每个长度为 m 的病毒 DNA 环状字符串 temp

{

for(j=1;j<=m;j++) temp.ch[j]=Virus.ch[i+j]; temp.ch[m+1]='\0'; //添加结束符号

flag-Index\_BF(Person,temp;1); //模式匹配

if(flag) break; //匹配即可退出循环

}

//for

If(flag) outfile<<vir+1<<” “<<Person.ch+1<<” “<<”YES”<<endl; else

outfile<<vir+1<<” “<<Person.ch+1<<” “<<”NO”<<endl;

}

//while

}

四、实验任务

17

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

18

实验四 基于哈夫曼树的数据压缩算法

一、实验目的

1.掌握哈夫曼树的构造算法。

2.掌握哈夫曼编码的构造算法。

二、实验内容

问题描述

输入一串字符串，根据给定的字符串中字符出现的频率建立相应的哈夫曼树，

构造哈夫曼编码表，在此基础上可以对压缩文件进行压缩(即编码)，同时可以对

压缩后的二进制编码文件进行解压(即译码)。

输入要求

多组数据，每组数据 1 行，为一个字符串(只考虑 26 个小写字母即可 )。当

输入字符串为“0”时，输入结束

输出要求

每组数据输出 2n+3 行(n 为输入串中字符类别的个数)。第 1 行为统计出来

的字符出现频率(只输出存在的字符，格式为:字符:频度)，每两组字符之间用一

个空格分隔，字符按照 ASCI 码从小到大的顺序排列。第 2 行至第 2n 行为哈夫

曼树的存储结构的终态( 如主教材 139 页表 5.2( b),一行当中的数据用空格分

隔)。第 2n+1 行为每个字符的哈夫曼编码(只输出存在的字符。格式为:字符:编码 )，

每两组字符之间用一个空格分隔，字符按照 ASCI 码从小到大的顺序排列。第

2n+2 行为编码后的字符串，第 2n+3 行为解码后的字符串(与输入的字符串相同)。

输入样例

aaaaaaabbbbbccdddd

aabccc

0

输出样例

19

a:7 b:5 c:2 d:4

1 7 7 0 0

2 5 6 0 0

3 2 5 0 0

4 4 5 0 0

5 6 6 3 4

6 11 7 2 5

7 18 0 1 6

a:0 b:10 c:110 d:111

00000001010101010110110111111111111

aaaaaabbbbbccdddd

a:2 b:1 c:3

1 2 4 0 0

2 1 4 0 0

3 3 5 0 0

4 3 5 2 1

5 6 0 3 4

a:11 b:10 c:0

111110000

aabccc

三、实验提示

此实验内容即要求实现主教材的案例 5.1，具体实现可参考算法 5.10 和算法

5.11。

首先,读入一行字符串,统计每个字符出现的频率;然后,根据字符出现的频率

利用算法 5.10 建立相应的哈夫曼树；最后，根据得到的哈夫曼树利用算法 5.11

求出每个字符的哈夫曼编码，然后译码。

20

1.哈夫曼树的存储表示：

typedef struct{

int weight;

//结点权值

int parent,lchild,rchild;//结点的双亲、左孩子、右孩子

}HTNode,\*HuffmanTree; //动态分配数组存储哈夫曼树

2.统计字符串每个字符出现的次数参考代码：

int count[26]={0};

for(i=0;i<strlen(str);i++){

count[str[i]-'a']++;

}

for(i=0;i<26;i++){

if(count[i]!=0){

printf("%c:%d ",i+'a',count[i]);

｝

2. 构造哈夫曼树，参见教材。其中选择函数参考代码：

void Select(HTNode HT[],int m,int &s1,int &s2){

int i;

int minweight=10000;

for(i=1;i<m;i++){

if(HT[i].weight<minweight&&HT[i].parent==0){

minweight=HT[i].weight;

s1=i;

}

}

minweight=10000;

for(i=0;i<m;i++){

21

if(HT[i].weight<minweight&&HT[i].parent==0&&i!=s1){

minweight=HT[i].weight;

s2=i;

}

}

}

3. 根据哈夫曼树求哈夫曼编码，参见教材。

4. 译码参考代码：

void HuffmanDeCode(HuffmanTree HT,char s[],char a[],int n){

int i=0;

int f=2\*n-1;

while(s[i]!='\0'){

if(s[i]=='0')

f=HT[f].lchild;

else

f=HT[f].rchild;

if(f<=n){

printf("%c",a[f]);

f=2\*n-1;

}

i++;

}

}

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试（译码可选做），完成实验报告的撰写。

22

实验五 基于二叉树的表达式求值算法

一、实验目的

1.掌握二叉树的二叉链表存储表示和二叉树的遍历等基本算法。

2.掌握根据中缀表达式创建表达式树的算法。

3.掌握基于表达式树的表达式求值算法。

二、实验内容

问题描述

输入一个表达式(表达式中的数均为小于 10 的正整数)，利用二叉树来表示该

表达式，创建表达式树，然后利用二叉树的遍历操作求表达式的值。

输入要求

多组数据。每组数据 1 行，为一个表达式，表达式以“=”结尾。当输入只有

一个“=”时，输入结束。

输出要求

每组数据输出 1 行，为表达式的值。

输入样例

2\*(2+5)=

1+2=

输出样例

14

3

三、实验提示

此实验内容即要求实现主教材的案例 5.2，具体实现可参考算法 5.12 和算

法 5.13。

首先，读入表达式，利用算法 5.12 创建一个基于二叉链表表示的表达式树;

然后，利用算法 5.13 对表达式树进行后序遍历，得到表达式的值。

23

1. 二叉树结点定义：

typedef struct Node {

char data;/\*数据域\*/

struct Node \*leftChild;/\*左子树指针\*/

struct Node \*rightChild;/\*右子树指针\*/

}BiTreeNode,\*BiTree;

2. 表达式树的创建算法，参见教材或网址：

https://www.bilibili.com/video/BV1rP411w7VZ/?spm\_id\_from=333.337.search-card.

all.click&vd\_source=608fc0da7c4966c59bc8b67a9e6b1073。

其中，CreateExpTree 参考代码:

BiTreeNode \*\*CreateExpTree(BiTreeNode \*\*T,BiTreeNode \*a,BiTreeNode \*b,char ch)

{

(\*T)=(BiTreeNode\*)malloc(sizeof(BiTreeNode)); (\*T)->data = ch;

(\*T)->leftChild = a;

(\*T)->rightChild = b;

return T;

}

算符优先算法参考实验二。

3. 两个工作栈：

#include<stack>

using namespace std;

stack<BiTreeNode\*>EXPT;

stack<char>OPTR;

入栈示例：OPTR.push('#');

24

出栈示例：OPTR.pop();

取栈顶元素示例：ch=OPTR.top();

4. 后序遍历表述式树求表述式的值，参考教材或网址：

https://www.bilibili.com/video/BV1Le4y1h7ZZ/?spm\_id\_from=333.337.search-card.

all.click&vd\_source=608fc0da7c4966c59bc8b67a9e6b1073。

其中，GetValue 参考代码：

int GetValue(char thetha,int a,int b) {

switch(thetha){

case '+':

return a+b;

break;

case '-':

return a-b;

break;

case '\*':

return a\*b;

break;

case '/':

return int(a/b);

break;

}

}

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

25

实验六 基于 Dijsktra 算法的最短路径求解

一、实验目的

1.掌握图的邻接矩阵表示法，掌握采用邻接矩阵表示法创建图的算法。

2.掌握求解最短路径的 Dijsktra 算法。

二、实验内容

问题描述

一张地图包括 n 个城市，假设城市间有 m 条路径(有向图)，每条路径的长度

已知。给定地图的一个起点城市和终点城市，利用 Dijsktra 算法求出起点到终

点之间的最短路径。

输入要求

多组数据，每组数据有 m+3 行。第一行为两个整数 n 和 m，分别代表城市

个数 n 和路径条数 m。第二行有 n 个字符，代表每个城市的名字。第三行到第

m+2 行每行有两个字符 a 和 b 和一个整数 d，代表从城市 a 到城市 b 有一条距离

为 d 的路。最后一行为两个字符，代表待求最短路径的城市起点和终点。当 n 和

m 都等于 0 时，输入结束。

输出要求

每组数据输出 2 行。第 1 行为一个整数，为从起点到终点之间最短路的长度。

第 2 行为一串字符串，代表该路径。每两个字符之间用空格隔开。

输入样例

3 3

A B C

A B 1

B C 1

C A 3

A C

26

6 8

A B C D E F

A F 100

A E 30

A C 10

B C 5

C D 50

D E 20

E F 60

D F 10

A F

0 0

输出样例

2

A B C

70

A C D F

三、实验提示

此实验内容即为主教材算法 6.10 的扩展内容，算法 6.10 求出源点 v0 到图

中其余所有顶点的最短路径。本实验要求求出一个指定起点到一个指定终点的最

短路径。

为了提高算法的效率，在求解时，可以加以判断，当已求得的终点为指定终

点时，则可以终止求解，按要求输出相应结果。

1. 图的邻接矩阵存储表示：

用邻接矩阵表示法表示图，除了一个用于存储邻接矩阵的二维数组外，还需

要一个一维数组存储顶点信息。

27

#define MVNum 100

#define MaxInt 32767

typedef struct{

char vex[MVNum]; //顶点表

int arcs[MVNum][MVNum];//邻接矩阵，权重为整数

int Vexnum;//顶点数

int arcnum; //边数

}AMGraph;

2. 采用邻接矩阵表示法创建有向图（无向图），参见教材算法 6.1。

3. 运用 Dijsktra 算法求最短路径，参见教材算法 6.10。

4. 输出从源点 V0 到终点 Vi 的路径参考代码：

string s,s1;

while(vi!=-1){

//vi 为顶点 Vi 在顶点表 vex[]中的索引。

s1=G.vex[vi];

s1=s1+" ";

s=s1+s;

vi=Path[vi];

}

cout<<"最短路径为："<<s; 注：一维数组 Path[i]用于记录源点 V0 到终点 Vi 的当前最短路径上 Vi 的直

接前驱顶点序号。初值为：如果从 V0 到 Vi 有弧，则 Path[i]为 V0 在顶点表中的

索引；否则为-1。

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

28

实验七 基于广度优先搜索的六度空间

理论验证

一、实验目的

1.掌握图的邻接矩阵和邻接表表示法，掌握采用邻接矩阵和邻接表表示法创

建图的算法。

2.掌握图的广度优先搜索算法。

3.掌握基于图的广度优先搜索的六度空间理论验证的算法。

二、实验内容

问题描述

“六度空间”理论又称作“六度分隔(Six Degrees of Separation)”理论。这个理论

可以通俗地阐述为:“你和任何一个陌生人之间所间隔的人不会超过六个，也就是

说，最多通过五个人你就能够认识任何一个陌生人。”假如给你一个社交网络图，

请你对每个节点计算符合“六度空间”理论的结点占结点总数的百分比。

输入要求

多组数据，每组数据 m+1 行。第一行有两个数字 n 和 m，代表有 n 个人和

m 组朋友关系。n 个人的编号为 1 到 n。第二行到第 m+1 行每行包括两个数字 a 和 b，代表这两个人互相认识。当 n 和 m 都等于 0 时，输入结束。

输出要求

每组数据输出 n 行，对每个结点输出与该结点距离不超过 6 的结点数占结点

总数的百分比，精确到小数点后 2 位。每个结节点输出一行，格式为“结点编号:(空

格)百分比%”。

输入样例

10 9

1 2

2 3

3 4

29

4 5

5 6

6 7

7 8

8 9

9 10

10 8

1 2

2 3

3 4

4 5

5 6

6 7

7 8

9 10

0 0

输出样例

1: 70.00%

2: 80.00%

3: 90.00%

4: 100.00%

5: 100.00%

6: 100.00%

7:100.00%

8: 90.00%

9: 80.00%

30

10:70.00%

1: 70.00%

2: 80.00%

3:80.00%

4: 80.00%

5:80.00%

6: 80.00%

7:80.00%

8:70.00%

9: 20.00%

10: 20.00%

三、实验提示

此实验内容即要求实现主教材的案例 6.1，具体实现可参考算法 6.14。

算法 6.14 给出了利用广度优先搜索方法进行验证的方案，实际上也可以利

用求解最短路径的方法(迪杰斯特拉算法或弗洛伊德算法)对六度空间理论进行

理论上的验证。读者可以根据算法 6.14 和最短路算法自行写出相应的验证方法。

下面是用图的邻接矩阵作为存储结构的参考代码：

1. 图的邻接矩阵存储表示：

用邻接矩阵表示法表示图，除了一个用于存储邻接矩阵的二维数组外，还需

要一个一维数组存储顶点信息。

#define MVNum 100

#define MaxInt 32767

typedef struct{

int vex[MVNum]; //顶点表

int arcs[MVNum][MVNum];//邻接矩阵，无向图，无权重，1 表示有边，0

表示无边。

31

int Vexnum;//顶点数

int arcnum; //边数

}AMGraph;

2. 采用邻接矩阵表示法创建无向图，参见教材算法 6.1。

3. 通过 BFS 广度优先搜索方法遍历图 G 来验证六度空间，参见教材算法 6.14.

其中求邻接点参考代码：

int FirstAdjVex(AMGraph G,int u){

int j;

for(j=1;j<=G.vexnum;j++){

if(G.arcs[u][j]==1)

return j;

}

return 0;

}

int NextAdjVex(AMGraph G,int u,int w){

int i,j;

for(j=w+1;j<=G.vexnum;j++){

if(G.arcs[u][j]==1)

return j;

}

return 0;

}

队列的使用：

#include<queue>

using namespace std;

定义一个队列：queue<int>Q;

32

入队：Q.push(start)；

出队： Q.pop();

取队头元素：u=Q.front();

判断队列是否为空：Q.empty();

4.百分比计算的时候包括顶点自己。

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

33

实验八 排序算法的实现与分析

一、实验目的

1.掌握常用的排序方法，并掌握用高级语言实现排序算法的方法；

2.深刻理解排序的定义和各种排序方法的特点，并能加以灵活应用；

3.了解各种方法的排序过程及其时间复杂度的分析方法。

二、实验内容

问题描述

统计成绩：给出 n 个学生的考试成绩表，每条信息由姓名和分数组成，试设

计一个算法：

1.按分数高低次序，打印出每个学生在考试中获得的名次，分数相同的为同

一名次；

2.按名次列出每个学生的姓名与分数。

输入要求

输入 n+1 行，前 n 行是 n 个学生的信息(姓名，成绩)，每个学生信息占一行，

姓名、

成绩用空格分隔，

成绩之后没有空格。

最后，

第 n+1 行是输入结束标志:00(空

格分隔的二个 0)。其中，姓名为字符串类型，成绩为浮点数类型。

输出要求

总计 n 行，每行是一个学生的信息(名次、姓名、成绩)，名次、姓名、成绩

用空格分隔。其中，成绩输出保留两位小数。

输入样例

张三 80.00

李四 96.00

王五 90.00

郑六 78.00

田七 85.00

34

李明 90.00

0 0

输出样例

1 李四 96.00

2 王五 90.00

2 李明 90.00

3 田七 85.00

4 张三 80.00

5 郑六 78.00

三、实验提示

1.定义结构体。

typedef struct student

//学生信息定义

{

char name[8];

//学生姓名

int score;

//学生成绩

};

2.定义结构体数组。

3.编写主程序，对数据进行排序。

4.至少采用一种排序算法实现，如直接插入排序、快速排序等算法等。

四、实验任务

认真阅读与理解实验内容的具体要求，参考教材相关章节，结合实验内容的

要求，编写实验程序并上机调试与测试，完成实验报告的撰写。

35

参考文献

[1]李冬梅, 张琪 编著. 数据结构习题解析与实验指导[M]. 北京: 人民邮电出版

社. 2022.

[2]严蔚敏, 李冬梅, 吴伟民 编著. 数据结构(C 语言版)[M]. 北京: 人民邮电出版

社. 2019.

36

# Document Outline

* [《数据结构》上机实验内容和要求](#p3)
* [实验一 基于线性表的图书信息管理](#p4)
* [实验二 基于栈的算术表达式求值](#p13)
* [实验三 基于字符串模式匹配算法的病毒](#p17)
* [感染检测问题](#p17)
* [实验四 基于哈夫曼树的数据压缩算法](#p21)
* [实验五 基于二叉树的表达式求值算法](#p25)
* [实验六 基于Dijsktra算法的最短路径求解](#p28)
* [实验七 基于广度优先搜索的六度空间](#p31)
* [理论验证](#p31)
* [实验八 排序算法的实现与分析](#p36)