计算机软件基础
课设指导书

信息工程学院

2019.10.30

目　录

ch0　绪论

1.概述

上机实习是对学生的一种全面综合训练，是与课堂听讲、自学和练习相辅相成的必不可少的一个教学环节。通常，实习题中的问题比平时的习题复杂得多，也更接近实际。实习着眼于原理与应用的结合点，使读者学会如何把书上学到的知识用于解决实际问题，培养软件工作所需要的动手能力；另一方面，能使书上的知识变“活”，起到深化理解和灵活掌握教学内容的目的。平时的练习较偏重于如何编写功能单一的“小”算法，而实习题是软件设计的综合训练，包括问题分析、总体结构设计、用户界面设计、程序设计基本技能和技巧，多人合作，以至一整套软件工作规范的训练和科学作风的培养。此外，还有很重要的一点是:机器是比任何教师都严厉的检查者。

为了达到上述目的，本书安排了七个主实习单元，各单元的训练重点在于基本的数据结构，而不强调面面俱到。各实习单元与教科书的各章只具有粗略的对应关系，一个实习题常常涉及几部分教学内容。

每个实习题采取了统一的格式，由问题描述、基本要求、测试数据、实现提示和选做内容五个部分组成。问题描述旨在为读者建立问题提出的背景环境，指明问题“是什么”。基本要求则对问题进一步求精，划出问题的边界，指出具体的参量或前提条件，并规定该题的最低限度要求。测试数据部分旨在为检查学生上机作业提供方便，在完成实习题时应自己设计完整和严格的测试方案，当数据输入量较大时，提倡以文件形式向程序提供输入数据。在实现提示部分，对实现中的难点及其解法思路等问题作了简要提示。选做部分向那些尚有余力的读者提出了更严峻的挑战，同时也能开拓其他读者的思路，在完成基本要求时力求避免就事论事的不良思想方法，尽可能寻求具有普遍意义的解法，使得程序结构合理，容易修改扩充。

不难发现，这里与传统的做法不同，题目设计得非常详细。会不会限制读者的想象力，影响创造力的培养呢？回答是：软件发展的一条历史经验就是要限制程序设计者在某些方面的创造性，从而使其创造能力集中地用到特别需要创造性的环节之上。实习题目本身就给出了问题说明和问题分解求精的范例，使读者在无形中学会模仿，它起到把读者的思路引上正轨的作用，避免坏结构程序和坏习惯，同时也传授了系统划分方法和程序设计的一些具体技术，保证实现预定的训练意图，使某些难点和重点不会被绕过去，而且也便于教学检查。题目的设计策略是：一方面使其难度和工作量都较大，另一方面给读者提供的辅助和可以模仿的成份也较多。当然还应指出的是，提示的实现方法未必是最好的，读者不应拘泥于此，而应努力开发更好的方法和结构。

本书的一个特点是为实习制定了严格的规范(见下一节)。一种普遍存在的错误观念是，调试程序全凭运气。学生花两个小时的上机时间只找出一个错误，甚至一无所获的情况是常见的。其原因在于，很多人只认识到找错误，而没有认识到努力预先避免错误的重要性，也不知道应该如何努力。实际上，结构不好、思路和概念不清的程序可能是根本无法调试正确的。严格按照实习步骤规范进行实习不但能有效地避免上述种种问题，更重要的是有利于培养软件工作者不可缺少

的科学工作方法和作风。

2.课程设计要求

（1）选题方式和原则

- 一人一题；
- 同一班级不能有重复题目；
- 12 月 27日（周一）下午 6 点前确定选题结果，班长汇总发给王楠老师。

（2） 指导老师

张敬敏、张有华、李霞、王楠、李宁、吴聪聪

（3）课设时间

2021 年 12 月 27 日到 2022 年 1 月 6 日。

具体时间安排如下：

2021.12.27　选题：根据课程设计安排，学生完成选题，并针对所选的课题，查阅资料，完成需求分析。指导教师对学生的需求分析验收。

2021.12.28----2021.12.29概要设计：根据需求分析结果确定数据模型及相关运算的定义，给出数据结构的ADT 表示；设计系统的原型。指导教师对学生的概要设计验收。

2021.1230----2021.12.31　详细设计：给出数据的存储表示（存储结构），完成相关算法的设计。指导教师对学生的详细设计验收。

2022.1.2----2022.1.3 编码调试：完成系统的编码、测试程序，要有给定的正确数据、错误数据和边界数据，要有不同的结果并进行结果分析，对于出现的错误，要进行错误分析，并进行改正。指导教师对学生的程序验收。

2022.1.4----2022.1.5　撰写报告：完成课程设计报告的撰写。

2022.16 考核：指导教师对学生的课程设计过程及成果进行验收，给出成绩。

（4）机房安排

（5）纪律

- 上午 8：00-12：00，下午 2：00-6：00，不得迟到、早退。

  每天上下午都要点名。

-

- 上机过程中不准从事与课程设计无关的事情，尤其是玩游戏，一旦发现，成绩为零。
- 严禁抄袭，如果发现雷同或重复率超过 50%，则成绩为零。
- 报告提交截止日期为 1 月 6 日下午四点前，四点后我们需要整理成绩提交到教务处。

（6）考核标准

每位同学独立完成课程设计题，独立撰写课程设计报告。不允许相互间抄袭，否则均以零分计算。

- 评分标准

1） 课程设计阶段验收；
2） 到课率；
3）课程设计完成的效果，必要时会采用现场演示和答辩的方式；
4）课程设计报告的撰写质量。

（7）提交材料和形式

提交材料：课程设计报告、程序代码压缩文件及相关文档。其中，课程设计报告命名：学号姓名-计算机软件基础实习报告.doc。

提交形式：将上述所有材料放在一个文件夹提交，文件夹命名为：学号姓名。

3.课程设计步骤

随之计算机性能的提高，它所面临的软件开发的复杂度也日趋增加。然而，编制一个 10，000 行的程序的难度绝不仅仅是一个 5，000 行的程序两倍，因此软件开发需要系统的方法。一种常用的软件开发方法，是将软件开发过程划分为分析、设计、实现和维护四个阶段。虽然数据结构课程中的实习题的复杂度远不如(从实际问题中提出来的)一个“真正的”软件，但为了培养一个软件工作者所应具备的科学工作的方法和作风，我们制订了如下所述完成实习的五个步骤：

(一) 问题分析和任务定义

通常，实习题目的陈述比较简洁，或者说是有模棱两可的含义。因此，在进行设计之前，首先应该充分地分析和理解问题，明确问题要求做什么?限制条件是什么。注意本步骤强调的是做什么?而不是怎么做。对问题的描述应避开算法和所涉及的数据类型，而是对所需完成的任务作出明确的回答。例如:输入数据的类型、值的范围以及输入的形式；输出数据的类型、值的范围及输出的形式；

若是会话式的输入，则结束标志是什么?是否接受非法的输入?对非法输入的回答方式是什么等。这一步还应该为调试程序准备好测试数据，包括合法的输入数据和非法形式的输入数据。

(二) 数据类型和系统设计在设计这一步骤中需分概要设计和详细设计两步实现。

概要设计指的是，对问题描述中涉及的操作对象定义相应的数据类型，并按照以数据结构为中心的原则划分模块，定义主程序模块和各抽象数据类型；

详细设计则为定义相应的存储结构并写出各算法的伪码。

在这个过程中，要综合考虑系统功能，使得系统结构清晰、合理、简单和易于调试，抽象数据类型的实现尽可能做到数据封装，基本操作的规格说明尽可能明确具体。作为概要设计的结果，应写出每个抽象数据类型的定义(包括数据结构的描述和每个基本操作的规格说明)，各个主要模块的算法，并画出模块之间的调用关系图。详细设计的结果是对数据结构和基本操作的规格说明作出进一步的求精，写出数据存储结构的类型定义，按照算法书写规范用类 C 语言写出函数形式的算法框架。在求精的过程中，应尽量避免陷入语言细节，不必过早表述辅助数据结构和局部变量。

(三) 编码实现和静态检查编码是把详细设计的结果进一步求精为程序设计语言程序。程序的每行不要超过 60 个字符。每个函数体，即不计首部和规格说明部分，一般不要超过 40 行，最长不得超过 60 行，否则应该分割成较小的函数。要控制 if 语句连续嵌套的深度。如何编写程序才能较快地完成调试是特别要注意的问题。对于编程很熟练的读者，如果基于详细设计的伪码算法就能直接在键盘上输入程序的话，则可以不必用笔在纸上写出编码，而将这一步的工作放在上机准备之后进行，即在上机调试之前直接用键盘输入。

然而，不管你是否写出编码的程序，在上机之前，认真的静态检查是必不可少的。多数初学者在编好程序后处于以下两种状态之一：一种是对自己的“精心作品“的正确性确信不疑；另一种是认为上机前的任务已经完成，纠查错误是上机的工作。这两种态度是极为有害的。事实上，非训练有素的程序设计者编写的程序长度超过 50 行时，极少不含有除语法错误以外的错误。上机动态调试决不能代替静态检查，否则调试效率将是极低的。

静态检查主要有两种方法，一是用一组测试数据手工执行程序(通常应先分模块检查)；二是通过阅读或给别人讲解自己的程序而深入全面地理解程序逻辑，在这个过程中再加入一些注解和断言。如果程序中逻辑概念清楚，后者将比前者有效。

(四)上机准备和上机调试

上机准备包括以下几个方面：

(1) 高级语言文本(体现于编译程序用户手册)的扩充和限制。例如，常用的 BorlandC(C++)和 MicrosoftC(C++)与标准 C(C++)的差别，以及相互之间的差别。

(2) 如果使用 C 或 C++语言，要特别注意与教科书的类 C 语言之间的细微差别。

(3) 熟悉机器的操作系统和语言集成环境的用户手册，尤其是最常用的命令操作，以便顺利进行上机的基本活动。

(4) 掌握调试工具，考虑调试方案，设计测试数据并手工得出正确结果。

上机调试程序时要带一本高级语言教材或手册。调试最好分模块进行，自底向上，即先调试低层函数。必要时可以另写一个调用驱动程序。这种表面上麻烦

的工作实际上可以大大降低调试所面临的复杂性，提高调试工作效率。

调试中遇到的各种异常现象往往是预料不到的，此时不应“冥思苦想”，而应动手确定疑点，通过修改程序来证实它或绕过它。调试正确后，认真整理源程序及其注释，印出带有完整注释的且格式良好的源程序清单和结果。

(五)总结和整理实习报告

4.课程设计报告内容

课程设计报告的开头应给出题目、班级、姓名、学号和完成日期，并包括以下七个内容：

1. 需求分析

以无歧义的陈述说明程序设计的任务，强调的是程序要做什么?明确规定:

(1) 输入的形式和输入值的范围；

(2) 输出的形式；

(3) 程序所能达到的功能；

(4) 测试数据：包括正确的输入及其输出结果和含有错误的输入及其输出结果。

2. 概要设计

说明本程序中用到的所有抽象数据类型的定义、主程序的流程以及各程序模块之间的层次(调用)关系。

3. 详细设计

实现概要设计中定义的所有数据类型，对每个操作只需要写出伪码算法；对主程序和其他模块也都需要写出伪码算法(伪码算法达到的详细程度建议为:按照伪码算法可以在计算机键盘直接输入高级程序设计语言程序)；画出函数的调用关系图。

4. 调试分析

内容包括：

(1)调试过程中遇到的问题是如何解决的以及对设计与实现的回顾讨论和分析；

(2)算法的时空分析(包括基本操作和其他算法的时间复杂度和空间复杂度的分析)和改进设想；

(3)经验和体会等。

5. 测试结果

列出你的测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于需求分析中所列。

6. 附录

带注释的源程序。可以只列出程序文件名的清单。

在附录中提供了实习报告实例。值得注意的是，实习报告的各种文档资料，如：上述中的前三部分要在程序开发的过程中逐渐充实形成，而不是最后补写(当然也可以应该最后用实验报告纸或打印)。

ch1　线性表及其应用

本次实习的主要目的在于帮助学生熟练掌握线性表的基本操作在两种存储结构上的实现，其中以各种链表的操作和应用作为重点内容。

1. 运动会分数统计

【问题描述】

参加运动会的n个学校编号为1～*n*。比赛分成*m*个男子项目和*w*个女子项目，项目编号分别为1～*m*和*m*+1～*m*+*w*。由于各项目参加人数差别较大，有些项目取前五名，得分顺序为7，5，3，2，1；还有些项目只取前三名，得分顺序为5，3，2。写一个统计程序产生各种成绩单和得分报表。

【基本要求】

1） 可以输入各个项目的前三名或前五名的成绩；

2） 能统计各学校总分，

3） 可以按学校编号或名称、学校总分、男女团体总分排序输出；

4） 可以按学校编号查询学校某个项目的情况；可以按项目编号查询取得前三或前五名的学校。

5） 数据存入文件并能随时查询

6) 规定：输入数据形式和范围：可以输入学校的名称，运动项目的名称

输出形式：有中文提示，各学校分数为整型。

界面要求：有合理的提示，每个功能可以设立菜单，根据提示，可以完成相关的功能要求。

存储结构：学生自己根据系统功能要求自己设计，但是要求运动会的相关数据要存储在数据文件中。

测试数据：

【测试数据】

要求使用1、全部合法数据；2、整体非法数据；3、局部非法数据。进行程序测试，以保证程序的稳定。

例如，对于*n*=4，*m*=3，*w* =2，编号为奇数的项目取前五名，编号为偶数的项目取前三名，设计一组实例数据。

【实现提示】

可以假设*n*≤20，*m*≤30，*w*≤20，姓名长度不超过 20 个字符。每个项目结束时，将其 编号、类型符(区分取前五名还是前三名) 输入，并按名次顺序输入运动员姓名、校名(和成 绩)。

【选作内容】

允许用户指定某项目采取其他名次取法。

2. 集合的并、交和差运算(此题目不能选，因为报告范例对应的就是该题目)

【问题描述】
编制一个能演示执行集合的并、交和差运算的程序。
【基本要求】
(1) 集合的元素限定为小写字母字符 ['a'..'z'] 。
(2) 演示程序以用户和计算机的对话方式执行。
【测试数据】
(1)Set1="magazine"，Set2="paper"，
Set1∪Set2="aegimnprz"，Setl ∩Set2="ae"，Set1-Set2="gimnz"。
(2)Set1= " 012oper4a6tion89"，Set2="error data"，
Set1∪Set2="adeinoprt"，Setl ∩Set2="aeort"，Set1-Set2="inp"。
【实现提示】
以有序链表表示集合。
【选作内容】
(1) 集合的元素判定和子集判定运算。
(2) 求集合的补集。
(3) 集合的混合运算表达式求值。
(4) 集合的元素类型推广到其他类型 ， 甚至任意类型。

3. 长整数四则运算

【问题描述】
设计一个实现任意长的整数进行加法运算的演示程序。
【基本要求】
利用双向循环链表实现长整数的存储，每个结点含一个整型变量。任何整型变量的范 围是－$(2^{15}-1)$~$(2^{15}-1)$ 。输入和输出形式：按中国对于长整数的表示习惯，每四位一组，组间用逗号隔开。
【测试数据】
(1) 0；0； 应输出 "0" 。
(2)－2345，6789； －7654，3211； 应输出 "－1，0000，0000" 。
(3)－9999，9999；1，0000，0000，0000； 应输出 "9999，0000，0001" 。
(4) 1，0001，0001；－1，0001，0001； 应输出 "0" 。
(5) 1，0001，0001；－1，0001，0000； 应输出 "1" 。
(6) -9999，9999，9999；－9999，9999，9999；应输出 "－1，9999，9999，9998" 。
(7) 1，0000，9999，9999；1； 应输出 "1，0001，0000，0000 " 。
【实现提示】
(1) 每个结点中可以存放的最大整数为 $2^{15}-1=32767$，才能保证两数相加不会溢出。但若这样存放，即相当于按32768进制数存放，在十进制数与32768进制数之间的转换十分不方便。故可以在每个结点中仅存十进制数的4位，即不超

过9999的非负整数，整个链表表示为万进制数。

(2) 可以利用头结点数据域的符号代表长整数的符号。相加过程中不要破坏两个操作数链表。不能给长整数位数规定上限。

【选作内容】

(1) 实现长整数的四则运算；

(2) 实现长整数的乘方和阶乘运算；

(3) 整型量范围是－ (2^n-1) ~ (2^n-1)， 其中，n是由程序读人的参量。输入数据的分

组方法可以另行规定。

4. 一元稀疏多项式计算器

【问题描述】

设计一个一元稀疏多项式简单计算器。

【基本要求】

一元稀疏多项式简单计算器的基本功能是：

(1) 输入并建立多项式 ；

(2) 输出多项式，输出形式为整数序列：n，c_1，e_1，c_2，e_2，…，c_n，e_n，其中n是多项式的项数，c_i 和e_i，分别是第 i 项的系数和指数，序列按指数降序排列；

(3) 多项式a和b相加，建立多项式$a+b$；

(4) 多项式a和b相减，建立多项式$a-b$ 。

(5) 多项式a和b相乘，建立多项式$a*b$ 。

【测试数据】

(1)$(2x+5x^8-3.1x^{11})+(7-5x^8+11x^9)=(-3.1x^{11}+11x^9+2x+7)$

(2)$(6x^{-3}-x+4.4x^2-1.2x^9)-(-6x^{-3}+5.4x^2-x^2+7.8x^{15})$

$=(-7.8x^{15}-1.2x^9+12x^{-3}-x)$

(3)$(1+x+x^2+x^3+x^4+x^5)+(-x^3-x^4)=(1+x+x^2+x^5)$

(4)$(x+x^3)+(-x-x^3)=0$

(5)$(x+x^{100})+(x^{100}+x^{200})=(x+2x^{100}+x^{200})$

(6)$(x+x^2+x^3)+0=x+x^2+x^3$

(7) 互换上述测试数据中的前后两个多项式

【实现提示】

用带表头结点的单链表存储多项式。

【选作内容】

(1) 计算多项式在x处的值。

(2) 求多项式 a 的导函数a' 。

(3) 多项式a和b相乘，建立乘积多项式ab。

(4) 多项式的输出形式为类数学表达式。例如 ，多项式 $-3x^8+6x^3-18$ 的输出形式为

$-3x\hat{}8+6x\hat{}3-18$，$x^{15}+(-8)x^7-14$的输出形式为$x\hat{}15-8x\hat{}7-14$。注意，数值为1的非零次项的输出形式中略去系数1，如项$1x^8$的输出形式为x^8，项 $-1x^3$的输出形式为$-x^3$。

(5) 计算器的仿真界。

5.幼儿园培训班学生信息管理

【问题描述】

幼儿园培训班有几种项目：绘画，唱歌，电子琴，钢琴，舞蹈，语言表演等，每个项目分为幼儿（3,4岁），小（5,6岁），中（6,7岁），大（8,9岁）班。设计功能模块，设计数据结构，完成对参加培训的小朋友信息的管理。

小朋友信息：编号，姓名，性别，年龄，班级，缴费日期，缴费年限。

【基本要求】

1.能够录入、删除、查询小朋友信息

2.能查询每个项目的信息，各班人员情况

【提示】

注意小朋友退出时，可以根据缴费日期和年限进行计算退费，另外能够根据这两项对到期限的小朋友提示缴费

6.通讯录管理系统

【问题描述】

模拟通讯录的管理，实现联系人信息的录入、删除、查找、分组等功能。

【基本要求】

1.通讯录信息必须放在文件中。

2.系统启动时，通讯录已有信息自动从外存文件调入到内存。

3.能够实现按姓名查询，按组名查询

ch2 栈和队列及其应用

仅仅认识到栈和队列是两种特殊的线性表是远远不够的，本次实习的目的在于使读者深入了解钱和队列的特性，以便在实际问题背景下灵活运用他们；同时还将巩固对这两种结构的构造方法的理解。

1. 停车场管理

【问题描述】

设停车场是一个可停放 n 辆汽车的狭长通道，且只有一个大门可供汽车进出。汽车在停车场内按车辆到达时间的先后顺序，依次由北向南排列(大门在最南端，最先到达的第一辆车停放在车场的最北端)，若车场内已停满 n 辆汽车，则后来的汽车只能在门外的便道上等候，一旦有车开走，则排在便道上的第一辆车即可开人；当停车场内某辆车要离开时，在它之后进入的车辆必须先退出车场为它让路，待该辆车开出大门外，其他车辆再按原次序进入车场，每辆停放在车场的车在它离开停车场时必须按它停留的时间长短交纳费用。试为停车场编制按上述要求进行管理的模拟程序。

【基本要求】

以栈模拟停车场，以队列模拟车场外的便道，按照从终端读入的输入数据序列进行模拟管理。每一组输入数据包括三个数据项：汽车“到达”或“离去”信息、汽车牌照号码以及到达或离去的时刻。对每一组输入数据进行操作后的输出信息为：若是车辆到达，则输出汽车在停车场内或便道上的停车位置；若是车辆离去，则输出汽车在停车场内停留的时间和应交纳的费用(在便道上停留的时间不收费)。栈以顺序结构实现，队列以链表结构实现。

2. 马踏棋盘

【问题描述】

设计一个国际象棋的马踏遍棋盘的演示程序。

【基本要求】

将马随机放在国际象棋的 8×8 棋盘 Board[8][8]的某个方格中，马按走棋规则进行移动。要求每个方格只进入一次，走遍棋盘上全部 64 个方格。编制非递归程序，求出马的行走路线，并按求出的行走路线，将数字 1，2，…，64 依次填入一个 8×8 的方阵，输出之。

3. 算术表达式求值演示

【问题描述】

表达式计算是实现程序设计语言的基本问题之一，也是栈的应用的一个典型例子。设计一个程序，演示用算符优先法对算术表达式求值的过程。

【基本要求】

以字符序列的形式从终端输入语法正确的、不含变量的整数表达式。利用教科书表 3.1 给出的算符优先关系，实现对算术四则混合运算表达式的求值，并仿照教科书的例 3-1 演示在求值中运算符栈、运算数校、输入字符和主要操作的变化过程。

【选做】实数表达式求之，加入若干变量的表达式求值

4. 银行业务模拟

【问题描述】

客户业务分为两种。第一种是申请从银行得到一笔资金，即取款或借款。第二种是向银行投入一笔资金，即存款或还款。银行有两个服务窗口，相应地有两个队列。客户到达银行后先排第一个队。处理每个客户业务时，如果属于第一种，且申请额超出银行现存资金总额而得不到满足，则立刻排入第二个队等候，直至满足时才离开银行；否则业务处理完后立刻离开银行。每接待完一个第二种业务的客户，则顺序检查和处理(如果可能)第二个队列中的客户，对能满足的申请者予以满足，不能满足者重新排到第二个队列的队尾。注意，在此检查过程中，一旦银行资金总额少于或等于刚才第一个队列中最后一个客户(第二种业务)被接待之前的数额，或者本次已将第二个队列检查或处理了一遍，就停止检查(因为此时已不可能还有能满足者)转而继续接待第一个队列的客户。任何时刻都只开一个窗口。假设检查不需要时间。营业时间结束时所有客户立即离开银行。

写一个上述银行业务的事件驱动模拟系统，通过模拟方法求出客户在银行内逗留的平均时间。

【基本要求】

利用动态存储结构实现模拟。

【测试数据】

一天营业开始时银行拥有的款额为 10000(元)，营业时间为 600(分钟)。其他模拟参量自定，注意测定两种极端的情况：一是两个到达事件之间的间隔时间很短，而客户的交易时间很长，另一个恰好相反，设置两个到达事件的间隔时间很长，而客户的交易时间很短。

【实现提示】

事件有两类：到达银行和离开银行。初始时银行现存资金总额为 total。开始营业后的第一今事件是客户到达，营业时间从 0 到 closetime。到达事件发生时随机地设置此客户的交易时间和距下一到达事件之间的时间间隔。每个客户要办理的款额也是随机确定的，用负值和正值分别表示第一类和第二类业务。变量 total，closetime 以及上述两个随机量的上下界均交互地从终端读入，作为模拟参数。

两个队列和一个事件表均要用动态存储结构实现。注意弄清应该在什么条件下设置离开事件，以及第二个队列用怎样的存储结构实现时可以获得较高的效率。注意：事件表是按时间顺序有序的。

模拟渡口：
某汽车轮渡口，过江渡船每次能载 10 辆车，每 10 分钟有一个渡轮到达。过江车辆分为客车和货车。上船有如下的规则：客车优先于货车，每上 4 辆客车才允许上一辆货车，若等待上船的客车数不足 4 辆，则以货车代替。编写程序，模拟渡口的管理，统计客车和货车的平均等待时间。假设车辆达到服从均匀分布。其他参数由用户指定。

5. 航空客运订票系统

【问题描述】

航空客运订票的业务活动包括：查询航线、客票预订和办理退票等。试设计一个航空客运订票系统，以使上述业务可以借助计算机来完成。

【基本要求】

(1)每条航线所涉及的信息有：终点站名、航班号、飞机号、飞行周日(星期几)、乘员定额、余票量、已订票的客户名单(包括姓名、订票量、舱位等级 1，2 或 3)以及等候替补的客户名单(包括姓名、所需票量)；

(2)系统能实现的操作和功能如下：

①录入：可以录入航班情况，全部数据可以只放在内存中，最好存储在文件中；

②查询航线：根据旅客提出的终点站名输出下列信息：航班号、飞机号、星期几飞行，最近一天航班的日期和余票额；

③承办订票业务：根据客户提出的要求(航班号、订票数额)查询该航班票额情况，若尚有余票，则为客户办理订票手续，输出座位号；若已满员或余票额少于订票额，则需重新询问客户要求。若需要，可登记排队候补；

④承办退票业务：根据客户提供的情况(日期、航班)，为客户办理退票手续，然后查询该航班是否有人排队候补，首先询问排在第一的客户，若所退票额能满足他的要求，则为他办理订票手续，否则依次询问其他排队候补的客户。

【测试数据】

由读者自行指定。

【实现提示】

两个客户名单可分别由线性表和队列实现。为查找方便，已订票客户的线性表应按客户姓名有序，并且，为插入和删除方便，应以链表作存储结构。由于预约人数无法预计，队列也应以链表作存储结构。整个系统需汇总各条航线的情况登录在一张线性表上，由于航线基本不变，可采用顺序存储结构，并按航班有序或按终点站名有序。每条航线是这张表上的一个记录，包含上述 8 个域、其中乘员名单域为指向乘员名单链表的头指针，等候替补的客户名单域为分别指向队头和队尾的指针。

【选作内容】

当客户订票要求不能满足时，系统可向客户提供到达同一目的地的其他航线情况。读者还可充分发挥自己的想象力，增加你的系统的功能和其他服务项目。

6. 迷宫问题

【问题描述】

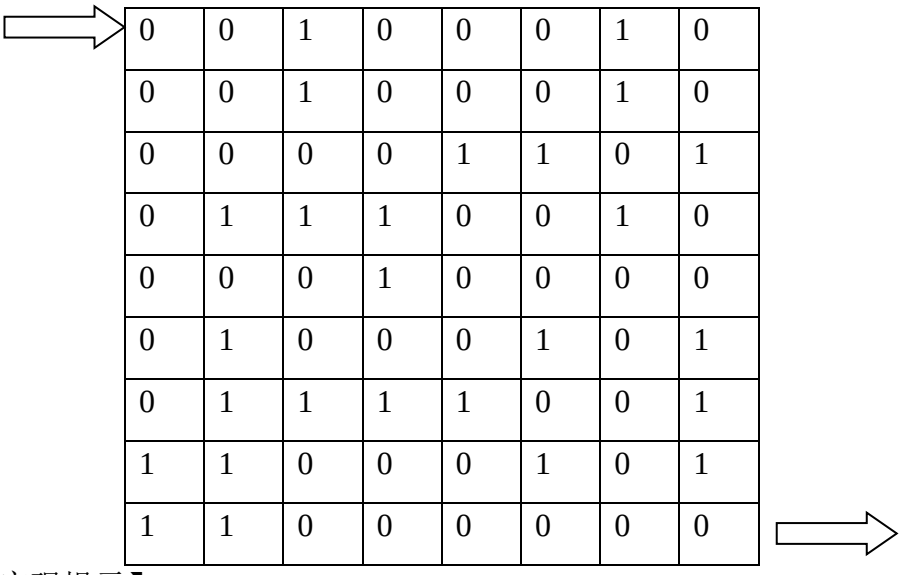
以一个 m×n 的长方阵表示迷宫，0 和 1 分别表示迷宫中的通路和障碍。设计一个程序，对任意设定的迷宫，求出一条从入口到出口的通路，或得出没有通路的结论。

【基本要求】

编写一个求解迷宫的非递归程序。求得的通路以三元组(i，j，d)的形式输出，其中：(i，j)指示迷宫中的一个坐标，d 表示走到下一坐标的方向。如：对于下列数据的迷宫，输出的一条通路为 z(1，1，1)，(1，2，2)，(2，2，2)，(3，2，3)，(3，1，2)，‥。

【测试数据】

迷宫的测试数据如下：左上角(1，1)为入口，右下角(8，9)为出口。



1	2	3	4	5	6	7	8
0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	0
0	0	0	0	1	1	0	1
0	1	1	1	0	0	1	0
0	0	0	1	0	0	0	0
0	1	0	0	0	1	0	1
0	1	1	1	1	0	0	1
1	1	0	0	0	1	0	1
1	1	0	0	0	0	0	0

【实现提示】

计算机解迷宫通常用的是“穷举求解“方法，即从人口出发，顺着某一个方向进行探索，若能走通，则继续往前进；否则沿着原路退回，换一个方向继续探索，直至出口位置，求得一条通路。假如所有可能的通路都探索到而未能到达出口，则所设定的迷宫没有通路。

可以二维数组存储迷宫数据，通常设定入口点的下标为(1，1)，出口点的下标为(n，n)。为处理方便起见，可在迷宫的四周加一圈障碍。对于迷宫中任一位置，均可约定有东、南、西、北四个方向可通。

【选作内容】

(l) 编写递归形式的算法，求得迷宫中所有可能的通路；

(2) 以方阵形式输出迷宫及其通路。

注：可参考教材 50 页

ch3　串及其应用

本实习单元的目的是熟悉串类型的实现方法和文本模式匹配方法,熟悉一般文字处理软件的设计方法,较复杂问题的分解求精方法。本实习单元的难度较大,在教学安排上可以灵活掌握完成此单元实习的时间。

编程技术训练要点:并行的模式匹配技术(3.1);字符填充技术(3.2,3.4);逻辑/物理概念隔离技术(GetAWord,3.2);活区操作技术(3.3);不定长对象的成块存储分配技术(3.3);命令识别与分析技术(3.3,3.4);串的动态组织技术(3.4);合理有效的错误处理方法(3.4);程序语法结构基本分析技术(3.5).

1. 文学研究助手

【问题描述】

文学研究人员需要统计某篇英文小说中某些形容词的出现次数和位置。试写一个实现这一目标的文字统计系统,称为"文学研究助手"。

【基本要求】

英文小说存于一个文本文件中。待统计的词汇集合要一次输入完毕,即统计工作必须在程序的一次运行之后就全部完成。程序的输出结果是每个词的出现次数和出现位置所在行的行号,格式自行设计。

【测试数据】

以你的C源程序模拟英文小说,C语言的保留字集作为待统计的词汇集。

【实现提示】

约定小说中的词汇一律不跨行。这样,每读入一行,就统计每个词在这行中的出现次数。出现位置所在行的行号可以用链表存储。若某行中出现了不止一次,不必存多个相同的行号。

如果读者希望达到选做部分(1)和(2)所提出的要求,则首先应把KMP算法改写成如下的等价形式,再将它推广到多个模式的情形。

```
i=1;j=1;
while(i!=s.curlen+1&&j!=t.curlerl十1)
{
    while(j!=O&&s.ch[i]!=t.ch[j])
        j=next[j];  //j==O或s.ch[i]==t.ch[j]
    j++;i++;//每次进入循环体,i只增加一次
}
```

【选作内容】

(1)模式匹配要基于KMP算法。

(2)整个统计过程中只对小说文字扫描一遍以提高效率。

(3)假设小说中的每个单词或者从行首开始,或者前置一个空格符。利用单词匹配特点另写一个高效的统计程序,与KMP算法统计程序进行效率比较。

(4)推广到更一般的模式集匹配问题,并设待查模式串可以跨行(提示:定义操作GetAChar)。

2. 文本格式化

【问题描述】

输入文件中含有待格式化〈或称为待排版〉的文本,它由多行的文字组成,例如一篇英文文章。每一行由一系列被一个或多个空格符所隔开的字组成,任何完整的字都没有被分割在两行(每行最后一个字与下一行的第一个字之间在逻辑上应该由空格分开),每行字符数不超过80。除了上述文本类字符之外,还存在着起控制作用的字符:符号"@"指示它后面的正文在格式化时应另起一段排放,即空一行,并在段首缩入8个字符位置。"@"自成一个字。

一个文本格式化程序可以处理上述输入文件,按照用户指定的版面规格重排版面z实现页内调整、分段、分页等文本处理功能,排版结果存入输出文本文件中。

试写一个这样的程序。

【基本要求】

(1)输出文件中字与字之间只留一个空格符,即实现多余空格符的压缩。

(2)在输出文件中,任何完整的字仍不能分割在两行,行尾不齐没关系,但行首要对齐(即左对齐)。

(3)如果所要求的每页页底所空行数不少于3,则将页号印在页底空行中第2行的中间位置上,否则不印。

(4)版面要求的参数要包含:

页长(PageLength)——每页内文字(不计页号)的行数。

页宽(PageWedth)——每行内文字所占最大字符数。

左空白(LeftMargin)——每行文字前的固定空格数。

头长(HeadingLength)——每页页顶所空行数。

脚长(FootingLength)——每页页底所空行数(含页号行)。

起始页号(StartingPageNumber)——首页的页号。

【测试数据】

略。注意在标点之后加上空格符。

【实现提示】

可以设:左空白数×2+页宽<=160,即行印机最大行宽,从而只要设置这样大的一个行缓冲区就足够了,每加工完一行,就输出一行。

如果输入文件和输出文件不是由程序规定死,而是可由用户指定,则有两种做法:一是像其他参量一样,将文件名交互地读入字符串变量中;更好的方式是让用户通过命令行指定,具体做法依机器的操作系统而定。

应该首先实现GetAWord(w)这一操作,把诸如行尾处理、文件尾处理、多余空格符压缩等一系列"低级"事务留给它处理,使系统的核心部分集中对付排版要求。

每个参数都可以实现缺省值②设置。上述排版参数的缺省值可以分别取56,60,10,5,5和1。

【选作内容】

(1)输入文件名和输出文件名要由用户指定。

(2)允许用户指定是否右对齐,即增加一个参量"右对齐否"(RightJustifying),缺省值可设为"y"(yes)。右对齐指每行最后一个字的字尾

要对齐,多余的空格要均匀分布在本行中各字之间。

(3)实现字符填充(characterstuffing)技术。"@"作为分段控制符之后,限制了原文中不能有这样的字。现在去掉这一限制:如果原文中有这样的字,改用两个"@"并列起来　表示一个"@"字。当然,如果原文中此符号夹在字中,就不必特殊处理了。

(4)允许用户自动按多栏印出一页。

3. 简单行编辑程序

【问题描述】

文本编辑程序是利用计算机进行文字加工的基本软件工具,实现对文本文件的插入、删除等修改操作。限制这些操作以行为单位进行的编辑程序称为行编辑程序。

被编辑的文本文件可能很大,全部读入编辑程序的数据空间(内存)的作法既不经济,也不总能实现。一种解决方法是逐段地编辑。任何时刻只把待编辑文件的一段放在内存,称为活区。试按照这种方法实现一个简单的行编辑程序。设文件每行不超过320个字符,很少超过80个字符。

【基本要求】

实现以下4条基本编辑命令:

(1) 行插入。格式:i<行号><回车><文本>.<回车>
　　将<文本>插入活区中第<行号>行之后。

(2) 行删除。格式:d<行号1>[<空格><行号2>]<回车>
　　删除活区中第<行号1>行(到第<行号2>行)。例如:"d10"和"d1014"。

(3) 活区切换。格式n<回车>
　　将活区写入输出文件,并从输入文件中读入下一段,作为新的活区。

(4) 活区显示。格式:p<回车>
　　逐页地(每页20行)显示活区内容,每显示一页之后请用户决定是否继续显示以后备页(如果存在)。印出的每一行要前置行号和一个空格符,行号固定占4位,增量为1。

各条命令中的行号均须在活区中各行行号范围之内,只有插入命令的行号可以等于活区第一行行号减1,表示插入当前屏幕中第一行之前,否则命令参数非法。

【测试数据】

自行设定,注意测试将活区删空等特殊情况。

【实现提示】

(1)设活区的大小用行数ActiveMULen(可设为100)来描述。考虑到文本文件行长通常为正态分布,且峰值在60到70之间,用320×ActiveMULen大小的字符数组实现存储将造成大量浪费。可以以标准行块为单位为各行分配存储,每个标准行块可含81个字符。这些行块可以组成一个数组,也可以利用动态链表连接起来。一行文字可能占多个行块。行尾可用一个特殊的ASCII字符(如(012)8)标识。此外,还应记住活区起始行号。行插入将引起随后各行行号的顺序下推。

(2)初始化函数包括:请用户提供输入文件名(空串表示无输入文件)和输出文件名,两者不能相同。然后尽可能多地从输入文件中读入各行,但不超过

ActiveMULen-LX的值可以自定,例如20。

(3)在执行行插入命令的过程中,每接收到一行时都要检查活区大小是否已达ActiveMaxLen。如果是,则为了在插入这一行之后仍保持活区大小不超过ActiveMaxLen应将插入点之前的活区部分中第一行输出到输出文件中;若插入点为第一行之前,则只得将新插入的这一行输出。

(4)若输入文件尚未读完,活区切换命令可将原活区中最后几行留在活区顶部,以保持阅读连续性;否则,它意味着结束编辑或开始编辑另一个文件。

(5)可令前三条命令执行后自动调用活区显示。

【选作内容】

(1)对于命令格式非法等一切错误作严格检查和适当处理。

(2)加入更复杂的编辑操作,如对某行进行串替换; 在活区内进行模式匹配等, 格式可以为S<行号>@<串1>@<串2><回车>和m<串><回车>。

4. 串基本操作的演示

【问题描述】

如果语言没有把串作为一个预先定义好的基本类型对待,又需要用该语言写一个涉及串操作的软件系统时,用户必须自己实现串类型。试实现串类型,并写一个串的基本操作的演示系统。

【基本要求】

在教科书4.2.2节用堆分配存储表示实现HString串类型的最小操作子集的基础上,实现串抽象数据类型的其余基本操作(不使用C语言本身提供的串函数)。参数合法性检查必须严格。

利用上述基本操作函数构造以下系统:它是一个命令解释程序,循环往复地处理用户键入的每一条命令,直至终止程序的命令为止。命令定义如下:

(1)赋值。 格式: A <串标识> <回车>

用<串标识>所表示的串的值建立新串,并显示新串的内部名和串值。例:A 'Hi!'

(2)判相等。格式: E <串标识1> <串标识2> <回车>

若两串相等,则显示"EQUAL",否则显示"UNEQUAL"。

(3)联接。 格式:C <串标识1> <串标识2> <回车>

将两串拼接产生结果串,它的内部名和串值都显示出来。

(4)求长度。格式:L <串标识> <回车>

显示串的长度。

(5)求子串。格式:S <串标识> +<数1>+<数2><回车>

如果参数合法,则显示子串的内部名和串值。<数>不带正负号。

(6)子串定位。格式:I <串标识1> <串标识2> <回车>

显示第二个串在第一个串中首次出现时的起始位置。

(7)串替换。格式: R <串标识1> <串标识2> <串标识3> <回车>

将第一个串中所有出现的第二个串用第三个串替换,显示结果串的内部名和串值,原串不变。

(8)显示。格式:P <回车>

显示所有在系统中被保持的串的内部名和串值的对照表。

(9)删除。格式:D <内部名> <回车>

删除该内部名对应的串,即赋值的逆操作。

(10)退出。格式:Q <回车>

结束程序的运行。

在上述命令中,如果一个自变量是串,则应首先建立它。基本操作函数的结果(即函数值)如果是一个串,则应在尚未分配的区域内新辟空间存放。

【测试数据】

自定。但要包括以下几组:

(1)E ‘’ ‘’<回车>,应显示"EQUAL"。

(2)E ‘abc’ ‘abcd’<回车>,应显示"UNEQUAL"。

(3)C ‘ ‘ ‘ ‘ <回车>,应显示"。

(4)I ‘a’ ‘’ <回车>,应报告:参数非法。

(5)R ‘aaa’ ‘aa’ ‘b’<回车>,应显示’ba’

(6)R ‘aaabc’ ‘a’ ‘aab’<回车>,应显示’aabaabaabbc’。

(7)R ‘Faaaaaaaa’ ‘aaaa’ ‘ab’,<回车>,应显示’abab’。

【实现提示】

【选作内容】

(1) 串头表改用单链表实现。

(2) 对命令的格式(即语法)作严格检查,使系统既能处理正确的命令,也能处理错误的命令。注意,语义检查(如某内部名对应的串已被删除而无定义等)和基本操作参数合法性检查仍应留给基本操作去做。

(3) 支持串名。将串名(可设不超过6个字符)存于串头表中。命令(1)(3)(5)要增加命令参数<结果串名>;命令(7)中的<串标识1> 改为<串名>,并用此名作为结果串名,删除原被替串标识,用<串名>代替<串标识>定义和命令解释中的内部名。每个命令执行完毕时立即自动删除无名串。

5. 程序分析

【问题描述】

读入一个C程序,统计程序中代码、注释和空行的行数以及函数的个数和平均行数,并利用统计信息分析评价该程序的风格。

【基本要求】

(1) 把 C 程序文件按字符顺序读入源程序;

(2) 边读入程序,边识别统计代码行、注释行和空行,同时还要识别函数的开始和结束,以便统计其个数和平均行数。

(3) 程序的风格评价分为代码、注释和空行三个方面。每个方面分为 A,B,C 和 D 四个等级 , 等级的划分标准是 :

	A级	B级	C级	D级
代码(函数平均长度)	10~15行	8~9或16~20行	5~7或21~24行	<5或>24行
注释(占总行数比率)	15~25%	10~14或26~30%	5~9或31~35%	<5%或>35%
空行(占总行数比率)	15~25%	10~14或26~30%	5~9或31~35%	<5%或>35%

【测试数据】

先对较小的程序进行分析。当你的程序能正确运行时,对你的程序本身进行分析。

【实现提示】

为了实现的方便,可作以下约定:

(1) 头两个字符是 FFF 的行称为注释行(该行不含语句)。除了空行和注释行外,其余均为代码行(包括类型定义、变量定义和函数头)。

(2) 每个函数代码行数(除去空行和注释行)称为该函数的长度。

(3) 每行最多只有一个"{" 、"}"、"switch" 和"struet"(便于识别函数的结束行)。

【选作内容】

(1) 报告函数的平均长度。

(2) 找出最长函数及其在程序中的位置。

(3) 允许函数的嵌套定义,报告最大的函数嵌套深度。

ch4 数组和广义表

本实习单元是作为从线性结构到非线性结构的过渡来安排的。数组和广义表可以看成其元素本身也是自身结构（ 递归结构 ）的线性表。广义表本质上是一种层次结构 ， 自顶向下识别并建立一个广义表的操作,可视为某种树的遍历操作:遍历逻辑的〈或符号形式的)结构,访问动作是建立一个结点。稀疏矩阵的十字链表存储结构也是图的一种存储结构。由此可见,这个实习单元的训练具有承上启下的作用。希望读者能深入研究数组的存储表示和实现技术,熟悉广义表的存储结构的特性。

编程技术训练要点:稀疏矩阵的表示方法及其运算的实现(4.1〉;共享数据的存储表示方法(4.2);形式系统的自底向上和自顶向下识别技术(4.3 〉； 递归算法的设计方法(4.3);表达式求值技术 (4.4) 。

1. 稀疏矩阵运算器

【问题描述】

稀疏矩阵是指那些多数元素为零的矩阵。利用 ″ 稀疏 ″ 特点进行存储和计算可以大大节省存储空间 ， 提高计算效率。实现一个能进行稀疏矩阵基本运算的运算器。

【基本要求】

以″带行逻辑链接信息″的三元组顺序表表示稀疏矩阵,实现两个矩阵相加、相减和相乘的运算。稀疏矩阵的输入形式采用三元组表示 ， 而运算结果的矩阵则以通常的阵列形式列出。

【实现提示】

1. 首先应输入矩阵的行数和列数 ， 并判别给出的两个矩阵的行、列数对于所要求作的运算是否相匹配。可设矩阵的行数和列数均不超过 20 。

2. 程序可以对三元组的输入顺序加以限制 ， 例如 ， 按行优先。注意研究教科书 5.3.2节中的算法 ， 以便提高计算效率。

3. 在用三元组表示稀疏矩阵时， 相加或相减所得结果矩阵应该另生成 ，乘积矩阵也可用二维数组存放。

【选作内容】

1. 按教科书 5.3.2 节中的描述方法 ， 以十字链表表示稀疏矩阵。

2. 增添矩阵求逆的运算 ， 包括不可求逆的情况。在求逆之前， 先将稀疏矩阵的内部表示改为十字链表。

2. 多维数组

【问题描述】

设计并模拟实现整型多维数组类型。

【 基本要求】

尽管 C 等程序设计语言已经提供了多维数组 , 但在某些情况下 , 定义用户所需的多维数组很有用的。通过设计并模拟实现多维数组类型, 可以深刻理解和掌握多维数组。整型多维数组应具有以下基本功能 :

(1) 定义整型多维数组类型 , 各维的下标是任意整数开始的连续整数 ;

(2) 下标变量赋值 , 执行下标范围检查 ;

(3) 同类型数组赋值 ;

(4) 子数组赋值 , 例如 ,a[1..n]=a[2..n+1];

a[2..4][3..5]=b[1..3][2..4];

(5) 确定数组的大小。

【 测试数据】

由读者指定。

【实现提示】

各基本功能可以分别用函数模拟实现 , 应仔细考虑函数参数的形式和设置。 定义整型多维数组类型时, 其类型信息可以存储在如下定义的类型的记录中：

```
#define MaxDim 5
Typedef struct
     int dim,
     BoundPtr lower, Upper;
     ConstPtr constants;
)NArray,*NarrayPtr;
```

整型多维数组变量的存储结构类型可定义为:

```
Typedef struct{
    ElemType  *elem;
    Int  num;
    NArrayType  TypeRecord;
}NArrayType;
```

【选作内容】

(1) 各维的下标是任意字符开始的连续字符。

(2) 数组初始化。

(3) 可修改数组的下标范围。

3. 识别广义表的头或尾

【问题描述】

写一个程序,建立广义表的存储结构,演示在此存储结构上实现的广义表求头/求尾操作序列的结果。

【基本要求】

(1) 设一个广义表允许分多行输入,其中可以任意地输入空格符,原子是不限长的仅由字母或数字组成的串。

(2) 广义表采用如教科书中图5.8所示结点的存储结构,试按表头和表尾的分解方法编写建立广义表存储结构的算法。

(3) 对已建立存储结构的广义表施行操作,操作序列为一个仅由"t"或"h"组成的串,它可以是空串(此时印出整个广义表),自左至右施行各操作,再以符号形式显示结果。

【测试数据】

由读者指定。

【实现提示】

(1) 广义表串可以利用 C 语言中的串类型或者利用实习三中已实现的串类型表示。

(2) 输入广义表时靠括号匹配判断结束,滤掉空格符之后,存于一个串变量中。

(3) 为了实现指定的算法，应在上述广义表串结构上定义以下4个操作:

Test(s):当 S 分别为空串、原子串和其他形式串时，分别返回‘N’，‘E’和‘O’(代表Null,Element和Other)。

hsub(s,h):s 表示一个由逗号隔开的广义表和原子的混合序列,h为变量参数,返回时为表示序列第一项的字符串。如果s为空串,则h也赋为空串。

tsub(s,t):s的定义同hsub操作；t为变量参数,返回时取从S中除去第一项(及其之后的逗号,如果存在的话)之后的子串。

strip(s,r)：s 的定义同 hsub 操作；r为变量参数。如果串S以"("开头和以")"结束,则返回时取除去这对括号后的子串,否则取空串。

(4) 在广义表的输出形式中,可以适当添加空格符,使得结果更美观。

【选作内容】

(1) 将hsub和tsub这两个操作合为一个(用变量参数h和t分别返回各自的结果),以便提高执行效率。

(2) 设原子为单个字母。广义表的建立算法改用边读入边建立的自底向上识别策略实现,广义表符号串不整体地缓冲。

ch5　树和图及其应用

1. 重言式判别

【问题描述】

一个逻辑表达式如果对于其变元的任一种取值都为真，则称为重言式；反之，如果对于其变元的任一种取值都为假，则称为矛盾式；然而，更多的情况下，既非重言式，也非矛盾式。试写一个程序，通过真值表判别一个逻辑表达式属于上述哪一类。

【基本要求】

(1) 逻辑表达式从终端输入，长度不超过一行。逻辑运算符包括 "|"，"&" 和 "~"，分别表示或、与和非，运算优先程度递增，但可由括号改变，即括号内的运算优先。逻辑变元 为大写字母。表达式中任何地方都可以含有多个空格符。

(2) 若是重言式或矛盾式，可以只显示"True forever"，或"False forever"，否则显示 "Satisfactible" 以及变量名序列，与用户交互。若用户对表达式中变元取定一组值，程序就求出并显示逻辑表达式的值。

【测试数据】

(1) (A|~A)&(B|~B)

(2) (A&~A)&C

(3) A|B|C|D|E|~A

(4) A&B&C&~B

(5) (A|B)&(A|~B)

(6) A&~B|~A&B;

2. 哈夫曼编/译码器

【问题描述】

利用哈夫曼编码进行通信可以大大提高信道利用率，缩短信息传输时间，降低传输成本。但是，这要求在发送端通过一个编码系统对待传数据预先编码，在接收端将传来的数据进行译码(复原)。对于双工信道(即可以双向传输信息的信道)，每端都需要一个完整的编/译码系统。试为这样的信息收发站写一个哈夫曼码的编/译码系统。

【基本要求】

一个完整的系统应具有以下功能：

(1)I：初始化(Initialization)。从终端读入字符集大小 n ，以及 n 个字符和 n 个权值，建立哈夫曼树，并将它存于文件 hfmTree 中。

(2)E：编码(Encoding)。利用已建好的哈夫曼树(如不在内存，则从文件 hfmTree 中读入)，对文件 ToBeTran 中的正文进行编码，然后将结果存入文件 CodeFile 中。

(3)D: 译码(Decoding)。利用已建好的哈夫曼树将文件 CodeFile 中的代码进

行译码，结果存入文件 TextFile 中。

(4)P：打印代码文件(Print)。将文件 CodeFile 以紧凑格式显示在终端上，每行 50 个代码。

(5)T：打印哈夫曼树(Tree printing)。将已在内存中的哈夫曼树以直观的方式(树或凹入表形式)显示在终端上，同时将此字符形式的哈夫曼树写入文件 TreePrint 中。

【测试数据】

(1)利用教科书例 6-2 中的数据调试程序。

(2)用下表给出的字符集和频度的实际统计数据建立哈夫曼树 ，并实现以下报文的编码和译码："THIS PROGRAM IS MY FAVORITE"。

字符		A	B	C	D	E	F	G	H	I	J	K	L	M
频度	186	64	13	22	32	103	21	15	47	57	1	5	32	20
字符	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
频度	57	63	15	1	48	51	80	23	8	18	1	16	1	

（3）根据（2）中哈夫曼树，对英文文件中的内容进行编码和译码，文件内容自定，但要求字符总数>100。

【选作内容】

(1)上述文件 CodeFile 中的每个"0"或"1"实际上占用了一个字节的空间，只起到示意或模拟的作用。为最大限度地利用编码存储能力，试改写你的系统，将编码结果以二进制形式存放在文件 CodeFile 中。

(2)修改你的系统，实现对你的系统的原程序的编码和译码(主要是将行尾符编/译码问题)。

(3)实现各个转换操作的源/目文件，均由用户在选择此操作时指定。

3. 图的存储及相互转换

【问题描述】图一般常用邻接表和邻接矩阵进行存储，两种存储形式都可以存储各种图，请你将用户输入（或文本中存储的）图的信息存储到内存用邻接表或邻接矩阵，并实现两种存储结构的互相转换。

【基本要求】

（１） 能将用户输入（文件存储）的图存入邻接表

（２） 能将用户输入（文件存储）的图存入邻接矩阵

（３） 能将邻接表存储的图转换成邻接矩阵存储

（４） 能将邻接矩阵存储的图转换成邻接表存储

（５） 输出邻接表、邻接矩阵数据

4. 图遍历的演示

【问题描述】

很多涉及图上操作的算法都是以图的遍历操作为基础的。试写一个程序，演示在连通的无向图上访问全部结点的操作。

【基本要求】

以邻接多重表为存储结构，实现连通无向图的深度优先和广度优先遍历。以

用户指定的结点为起点，分别输出每种遍历下的结点访问序列和相应生成树的边集。

【测试数据】

教科书图 7.33。暂时忽略里程，起点为北京。

【实现提示】

设图的结点不超过 30 个，每个结点用一个编号表示(如果一个图有 n 个结点，则它们的编号分别为 1，2，…，n)。通过输入图的全部边(存于数据文件中，从文件读写)输入一个图，每个边为一个数对，可以对边的输入顺序作出某种限制。注意，生成树的边是有向边，端点顺序不能颠倒。

【选作内容】

(1)借助于栈类型(自己定义和实现)，用非递归算法实现深度优先遍历。

(2)以邻接表为存储结构，建立深度优先生成树和广度优先生成树，再按凹入表或树形打印生成树。

5. 教学计划编制问题

【问题描述】

大学的每个专业都要制定教学计划。假设任何专业都有固定的学习年限，每学年含两学期，每学期的时间长度和学分上限值均相等。每个专业开设的课程都是确定的，而且课程在开设时间的安排必须满足先修关系。每门课程有哪些先修课程是确定的，可以有任意多门，也可以没有。每门课恰好占一个学期。试在这样的前提下设计一个教学计划编制程序。

【基本要求】

(1)输入参数包括：学期总数，一学期的学分上限，每门课的课程号(固定占 3 位的字母数字串)、课程名、学分和直接先修课的课程号。

(2)允许用户指定下列两种编排策略之一：一是使学生在各学期中的学习负担尽量均匀；二是使课程尽可能地集中在前几个学期中。

(3)若根据给定的条件问题无解，则报告适当的信息，否则将教学计划输出到用户指定的文件中。计划的表格格式自行设计。

【测试数据】

学期总数：4；学分上限：20；该专业共开设 12 门课，课程号从 C01 到 C12，学分顺序为 2，3，4，3，2，3，4，4，7，5，2，3。先修关系见教科书图 7.26。

6. 校园导游咨询

【问题描述】

设计一个校园导游程序，为来访的客人提供各种信息查询服务。

【基本要求】

(1) 设计你所在学校的校园平面图，所含景点不少于 10 个。以图中顶点表示校内各景点，存放景点名称、代号、简介等信息；以边表示路，存放路径长度等相关信息。

(2) 为来访客人提供图中任意景点相关信息的查询。

(3) 为来访客人提供图中任意景点的问路查询，即查询任意两个景点之间的

一条最短的简单路径。

【测试数据】

由读者根据实际情况指定。

【实现提示】

一般情况下，校园的道路是双向通行的，可设校园平面图是一个无向网。顶点和边均含有相关信息。

【选作内容】

(1) 求校园图的关节点。

(2) 提供图中任意景点问路查询，即求任意两个景点之间的所有路径。

(3) 提供校园图中多个景点的最佳访问路线查询，即求途经这多个景点的最佳（短）路径。

(4) 校园导游图的景点和道路的修改扩充功能。

(5) 扩充道路信息，如道路类别（车道、人行道等）、沿途景色等级，以及可按客人所需分别查询人行路径或车行路径或观景路径等。

(6) 扩充每个景点的邻接景点的方向等信息，使得路径查询结果能提供详尽的导向信息。

7. 高校专用通信网络建设

【问题描述】

中国移动公司正在积极推广 4G 通信应用，计划在河北省高校之间建立一个专用通信网络，请为其规划一个投资最省的通信线路架设方案。

【基本要求】

（1） 用无向网模拟该系统，顶点表示各高校，边表示线路建设成本；

（2） 高校数量不少于 10 个，覆盖石家庄、天津、唐山等地的高校；

（3） 输出方案的结果直观、明确；

（4） 交互式改变某些线路的建设成本，可重新输出新方案

（5）以图形和文本两种形式输出生成树中各条边以及他们的权值。

【实现提示】

（1）利用克鲁斯卡尔算法求网的最小生成树。

（2）实现教科书 6.5 节中定义的抽象数据类型 MFSet。以此表示构造生成树过程中的连通分量。

（3）通信线路一旦建立，必然是双向的。因此，构造最小生成树的网一定是无向网。设图的顶点数不超过 30 个，并为简单起见、网中边的权值设成小于 100 的整数，可利用 C 语言提供的随机数函数产生。

（4）图的存储结构的选取应和所作操作相适应。为了便于选择权值最小的边，此题的存储结构应选择存储边 (带权) 的邻接矩阵表示图。

【选作内容】

利用堆排序(参见教科书 10.4.3 节) 实现选择权值最小的边。

8. 表达式类型的实现

【问题描述】

一个表达式和一棵二叉树之间，存在着自然的对应关系。写一个程序，实现基于二叉树表示的算术表达式 Expression 的操作。

【基本要求】

假设算术表达式 Expression 内可以含有变量 (a~z) 、常量 (0~9) 和二元运算符 (+, - , *, /, ^(乘幂))。实现以下操作：

(1) ReadExpk(E) ------以字符序列的形式输入语法正确的前缀表示式并构造表达式 E。

(2) WriteExpk(E) ------用带括弧的中缀表示式输出表达式 E 。

(3) Assign(V,c) ------实现对变量 V 的赋值 (V=c), 变量的初值为 0。

(4) Value(E) ------对算术表达式 E 求值。

(5) CompoundExpr(P,E1,E2) ------构造一个新的复合表达式(E1)P(E 2)。

【测试数据】

(1) 分别输入 0; a; -91; +a*bc; +*5^x2*8x; +++ *3^ X3*2^ x2x6 并输出。

(2) 每当输入一个表达式后 对其中的变量赋值，然后对表达式求值。

【实现提示】

(1) 在读入表达式的字符序列的同时，完成运算符和运算数(整数)的识别处理以及相应的运算。

(2) 在识别出运算数的同时，要将其字符形式转换成整数形式。

(3) 用后根遍历的次序对表达式求值。

(4) 用中缀表示输出表达式 E 时，可添加括号，以正确反映运算的优先次序。

【选作内容】

(1) 增加求偏导数运算 Diff (E,V) ------求表达式 E 对变量 V 的导数。

(2) 在表达式中添加三角函数等初等函数的操作。

(3) 增加常数合并操作 MergeConsk(E) ------合并表达式 E 中所有常数运算。例, 对表达式 E=(2+3-a) *(b+3*4)进行合并常数的操作后，求得 E=(5-a)*(b+12)。

(4) 以表达式的原书写形式输入。

9. 全国交通咨询模拟

【问题描述】

出于不同目的的旅客对交通工具有不同的要求。例如，因公出差的旅客希望在旅途中的时间尽可能短，出门旅游的游客则期望旅费尽可能省，而老年旅客则要求中转次数最少。编制一个全国城市间的交通咨询程序，为旅客提供两种或三种最优决策的交通咨询。

【基本要求】

(1) 提供对城市信息进行编辑 (如：添加、修改、删除) 的功能。

(2) 城市之间有两种交通工具，火车和飞机。提供对列车时刻表和飞机航班进行编辑(添加、修改、删除) 的功能。

(3) 提供两种最优决策：最快到达或最省钱到达。全程只考虑一种交通工具。

(4) 旅途中耗费的总时间应该包括中转站的等候时间。

(5) 咨询以用户和计算机的对话方式进行。由用户输入起始站、终点站、最优决策原则和交通工具，输出信息：最快需要多长时间才能到达或者最少需要多少旅费才能到达，并详细说明依次于何时乘坐哪一趟列车或哪一次班机到何地。

【测试数据】

参考教科书 7.6 节图 7.33 的全国交通图，自行设计列车时刻表和飞机航班。

【实现提示】

(1) 对全国城市交通图和列车时刻表及飞机航班表的编辑，应该提供文件形式输入和键盘输入两种方式。飞机航班表的信息应包括：起始站的出发时间、终点站的到达时间和票价；列车时刻表则需根据交通图给出各个路段的详细信息；例如：基于教科书 7.6 节，图 7.33 的交通图，对从北京到上海的火车，需给出北京至天津，天津至徐州及徐州至上海各段的出发时间、到达时间及票价等信息。

(2) 以邻接表作交通图的存储结构，表示边的结点内除含有邻接点的信息外，还应包括交通工具、路程中消耗的时间、花费以及出发和到达的时间等多项属性。

【选作内容】

增加旅途中转次数最少的最优决策。

10. 关键路径问题

【问题描述】

工程中或工厂的流水线中经常会有一些工作或活动直接影响到整个工期，起到关键作用，将这些活动提前完成可以使整个工期缩短。理解课本中的关键路径的概念。实现用户输入（或文件读取）一个工程的信息，实现求出这个工程中的关键路径。

【基本要求】

(1) 将工程信息存入邻接表。

(2) 判断工程中有无环路。

(3) 求出工程中的关键路径并输出。

【选作】

试判断每个关键活动最多能使工程缩短多长时间

11.学校超市选址问题

【问题描述】

学校有很多的办公楼和宿舍楼，教学楼。各楼之间的距离不同，购买力也不同。现在要在学校开一个超市，请你为该超市选一个合适的地址（楼），实现总体最优。

【基本要求】

学校的楼房至少要 10 座

【提示】注意边上权的设定标准

ch6　存储管理、查找和排序

与前五个实习单元不同，本实习单元旨在集中对几个专门的问题作较为深入的探讨和理解，不强调对某些特定的编程技术的训练。

动态存储管理问题的实习遇到高级语言限制方面的困难。6.1 题绕过了这个限制。尽管与实际情况有差距，例如，求得伙伴地址以后不能用它寻址得到伙伴的头，但还是较完整地体现了伙伴系统的主要框架和意图。希望选择此题的读者认真思考:如何修改自己的程序才能得到实用的系统。

6.2 题和 6.5 题集中地探讨了不同的索引技术。散列技术是索引技术中的一种非常重要和有效的技术，但与实际问题(主要是关键字集的形态和特点)关系甚密。哈希函数的选择和冲突解决方法的选用都带有较强的技巧性和经验性，自己动手试一试是非常有益的：平衡树和键树有其一定的实用范围：B 树是动态索引文件的一种极好的组织方式，也是物理数据库实现的基本技术。尽管此题难度大了一些，但给读者带来的提高也相应地大些，其中所含的程序设计技巧也比较多。

6.6 题除了使读者对各种内部排序方法及效率获得深入理解之外，还可以给读者以启发：对于一个一般的问题而言，开发高效算法的可能性如何?应该如何寻找和构造高效算法?6.7 题是一个多关键字，的排序问题。

1. 伙伴存储管理系统演示

【问题描述】

伙伴存储管理系统是一种巧妙而有效的方法。试写一个演示系统，演示分配和回收存储块前后的存储空间状态变化。

【基本要求】

程序应不断地从终端读取整数 n。每个整数是一个请求。如果 n>0，则表示用户申请大小为 n 的空间：如果 n<0，则表示归还起始地址(即下标)为-n 的块才日果 n=0，则表示结束运行。每读入一个数，就处理相应的请求，并显示处理之后的系统状态。

系统状态由占用表和空闲表构成。显示系统状态意味着显示占用表中各块的始址和长度，以及空闲表中各种大小的空闲块的始址和长度。

【测试数据】

l，一<①1>，3，4，4，4，一<①4>，一<①3>，2，2，2，2，一<②4>，一<①2>，一<②2>，一<③2>，一<④2>，一<③4>，40，0。其中，<③，2>表示第③次申请大小为 2 的空间使得块的始址。其余类推。

【实现提示】

可以取 m=5，即 Spacesize=2^5，数据结构如下:

```
Typedef struct BlkHeader{
    BlkHeader   *llink，*rlink；
    int             tag；
    Int             kvalue；
    Int             blkstart；    //块起始地址
}BlkHeader，*Link；
Typedef struct{
    Int     blksize；
    Link    first；
}ListHeader；
Typedef char ell；        //cell 也可以是其他单位
```

主要变量是:

```
cell space[Spacesize] //被管理的空间
ListHeader    avail[m+1]；      //可用空间表
Link      ailocated；.//占用表的表头指针
```

在这里，我们把每块的块头分离出来，通过 blkstart 域与相应的块建立联系。每个块一旦被分配，其块头就进入占用表，其中的各块头由 rlink 域链接在一起。tag 域实际上不起作用，但为了与实际伙伴管理系统更接近，没有把它去掉。显然，在这种模拟实现方法中，不对数组 space 作任何引用或赋值。

【选作内容】

(1)同时还用直观的图示方式显示状态。

(2)写一个随机地申请和归还各种规格的存储块的函数考验你的伙伴系统。

2. 哈希表设计

【问题描述】

针对某个集体(比如你所在的班级)中的"人名"设计一个哈希表，使得平均查找长度不超过 R，完成相应的建表和查表程序。

【基本要求】

假设人名为中国人姓名的汉语拼音形式。待填入哈希表的人名共有 30 个，取平均查找长度的上限为 2。哈希函数用除留余数法构造，用伪随机探测再散列法处理冲突。

【测试数据】

取读者周围较熟悉的 30 个人的姓名。

【实现提示】

如果随机函数自行构造，则应首先调整好随机函数，使其分布均匀。人名的长度均不超过 19 个字符(最长的人名如 z 庄双双(ZhangStmarlgShang〉。字符的取码方法可直接利用 C 语言中的 toascii 函数，并可对过长的人名先作折叠处理。

【选作内容】

(1)从教科书上介绍的几种哈希函数构造方法中选出适用者并设计几个不同的哈希函数，比较它们的地址冲突率(可以用更大的名字集合作试验〉。

(2)研究这 30 个人名的特点，努力找一个哈希函数，使得对于不同的拼音名一定不发生地址冲突。

(3)在哈希函数确定的前提下尝试各种不同处理冲突的方法，考查平均查找长度的变化和造好的哈希表中关键字的聚簇性。

3. 图书管理

【问题描述】

图书管理基本业务活动包括:对一本书的采编入库、清除库存、借阅和归还等等。试设计一个图书管理系统，将上述业务活动借助于计算机系统完成。

【基本要求】

〈1〉每种书的登记内容至少包括书号、书名、著者、现存量和总库存量等五项。

〈2〉作为演示系统，不必使用文件，全部数据可以都在内存存放。但是由于上述四项基本业务活动都是通过书号(即关键字〉进行的，所以要用 B 树〈24 树〉对书号建立索引，以获得高效率。

〈3〉系统应实现的操作及其功能定义如下:

① 采编入库 z 新购入一种书，经分类和确定书号之后登记到图书账目中去。如果这种书在账中已有，则只将总库存量增加。

② 清除库存:某种书已无保留价值，将它从图书账目中注销。

③ 借阅:如果一种书的现存量大于零，则借出一本，登记借阅者的图书证号和归还期限。

④ 归还 z 注销对借阅者的登记，改变该书的现存量。

⑤ 显示:以凹入表的形式显示 B 树。这个操作是为了调试和维护的目的而设置的。

【测试数据】

入库书号:35，16，18，70，5，50，22，60，13，17，12，45，25，毡，15，90，30，7 然后清除:45，90，50，22，42

其余数据自行设计。由空树开始，每插入删除一个关键字后就显示 B 树的状态。

【实现提示】

(1)24 树的查找算法是基础，入库和清除操作都要调用。难点在于删除关键字的算法，因而只要算法对 2-3 树适用就可以了，暂时不必追求高阶 B 树也适用的删除算法。

(2)每种书的记录可以用动(或静)态链式结构。借阅登记信息可以链接在相应的那种书的记录之后。

【选作内容】

(l)将一次会话过程(即程序一次运行)中的全部人机对话记入一个日志文件"log"中去。

(2)增加列出某著者全部著作名的操作。思考如何提高这一操作的效率，参阅教科书 12.6.2 节。

(3〉增加列出某种书状态的操作。状态信息除了包括这种书记录的全部信息外还包括最早到期(包括已逾期)的借阅者证号，日期可用整数实现，以求简化。

(4)增加预约借书功能。

4. 平衡二叉树操作的演示

【问题描述】

利用平衡二叉树实现一个动态查找表。

【基本要求】

实现动态查找表的三种基本功能:查找、插入和删除。

【测试数据】

由读者自行设定。

【实现提示】

(1)初始，平衡二叉树为空树，操作界面给出查找、插入和删除三种操作供选择。每种操作均要提示输入关键字。每次插入或删除一个结点后，应更新平衡二叉树的显示。

(2)平衡二叉树的显示可采用凹入表形式，也可以采用图形界面画出树形。

(3)教科书已给出查找和插入算法，本题重点在于对删除算法的设计和实现。假设要删除关键字为 x 的结点。如果 x 不在叶子结点上，则用它左子树中的最大值或右子树中的最小值取代 x。如此反复取代，直到删除动作传递到某个叶子结点。删除叶子结点时，若需要进行平衡变换，可采用插入的平衡变换的反变换(如，左子树变矮对应于右子树长高)。

【选作内容】

(1)合并两棵平衡二叉树。

(2)把一棵平衡二叉树分裂为两棵平衡二叉树，使得在一棵树中的所有关键字都小于或等于几，另一棵树中的任一关键字都大于几。

5. 英语词典的维护和识别

【问题描述】

Trie 树通常作为一种索引树，这种结构对于大小变化很大的关键字特别有用。利用 Trie 树实现一个英语单词辅助记忆系统，完成相应的建表和查表程序。

【基本要求】

不限定 Trie 树的层次，每个叶子结点只含一个关键字，采用单字符逐层分割的策略，实现 Trie 树的插入、删除和查询的算法，查询可以有两种方式:查询一个完整的单词或者查询以某几个字母开头的单词。

【测试数据】

自行设定。

【实现提示】

以实习三中已实现的串类型或 C 语言中提供的长度不限的串类型表示关键字，叶子结点内应包括英语单词及其注音、释义等信息。

【选作内容】
限定 Trie 树的层次，每个叶子结点可以包含多个关键字。

6. 内部排序算法比较

【问题描述】
在教科书中，各种内部排序算法的时间复杂度分析结果只给出了算法执行时间的阶，或大概执行时间。试通过随机数据比较各算法的关键字比较次数和关键字移动次数，以取得直观感受。
【基本要求】
(1)对以下 6 种常用的内部排序算法进行比较 z 起泡排序、直接插入排序、简单选择排序、快速排序、希尔排序、堆排序。
(2)待排序表的表长不小于 1005 其中的数据要用伪随机数产生程序产生：至少要用 5 组不同的输入数据作比较：比较的指标为有关键字参加的比较次数和关键字的移动次数(关键字交换计为 3 次移动)。
(3)最后要对结果作出简单分析，包括对各组数据得出结果波动大小的解释。
【测试数据】
由随机数产生器生成。
【实现提示】
主要工作是设法在已知算法中的适当位置插入对关键字的比较次数和移动次数的计数操作。程序还可以考虑几组数据的典型性，如，正序、逆序和不同程度的乱序。注意采用分块调试的方法。
【选作内容】
(1)增加折半插入排序、二路插入排序、归并排序、基数排序等。
(2)对不同的输入表长作试验，观察检查两个指标相对于表长的变化关系。还可以对稳定性作验证。

7. 多关键字排序

【问题描述】
多关键字的排序有其一定的实用范围。例如:在进行高考分数处理时，除了需对总分进行排序外，不同的专业对单科分数的要求不同，因此尚需在总分相同的情况下，按用户提出的单科分数的次序要求排出考生录取的次序。
【基本要求】
(1)假设待排序的记录数不超过 10000，表中记录的关键字数不超过 5，各个关键字的范围均为 0 至 100。按用户给定的进行排序的关键字的优先关系，输出排序结果。
(2)约定按 LSD 法进行多关键字的排序。在对各个关键字进行排序时采用两种策略:其一是利用稳定的内部排序法，其二是利用"分配"和"收集"的方法。并综合比较这两种策略。
【测试数据】
由随机数产生器生成。

【实现提示】

用 5 至 8 组数据比较不同排序策略所需时间。由于是按 LSD 方法进行排序，则对每个关键字均可进行整个序列的排序，但在利用通常的内部排序方法进行排序时，必须选用稳定的排序方法。借助"分配"和"收集"策略进行的排序，如同一趟"基数排序"，由于关键字的取值范围为 0 至 100，则分配时将得到 104 个链表。

【选作内容】

增添按 MSD 策略进行排序，并和上述两种排序策略进行综合比较。

ch7 文件操作

1. 文件索引

已知职工文件中包括职工号、职工姓名、职务和职称等若干数据项(见下表)。职务有校长、系主任、室主任和教员；校长领导所有系主任，系主任领导他所在系的所有室主任，室主任领导他所在室的全体教员；职称有教授、副教授和讲师3 种。

请给该文件建立索引,通过该索引文件,能做到:

(1)能够检索出全体职工间领导与被领导的情况；

(2)能够分别检索出全体教授、全体副教授、全体讲师。要求指针数量尽可能少，给出各指针项索引的名称及含义即可

职工号	职工姓名	职务	职称
1	张军	教员	讲师
2	沈灵	系主任	教授
3	叶明	校长	教授
4	张莲	室主任	副教授
5	叶宏	系主任	教授
6	周芳	教员	教授
7	刘光	系主任	教授
8	黄兵	教员	讲师
9	李民	室主任	教授
10	赵松	教员	副教授
…	…	…	…

2. 成绩分析问题

[问题描述]

录入、保存一个班级学生多门课程的成绩，并对成绩进行分析。

[基本要求]

1、通过键盘输入各学生的多门课程的成绩，建立相应的文件 input.dat。

2、对文件 input.dat 中的数据进行处理，要求具有如下功能：

(1)按各门课程成绩排序，并生成相应的文件输出。

(2)计算每人的平均成绩，按平均成绩排序，并生成文件。

(3)求出各门课程的平均成绩、最高分、最低分、不及格人数、60~69 分人数、70~79 分人数、80~89 分人数、90 分以上人数。

(4)根据姓名或学号查询某人的各门课成绩，重名情况也能处理。

3、界面美观。

3.模拟小商店管理系统

【问题描述】
实现对一个小商店的进货，卖货和统计进行模拟。
【基本要求】
1、进货：通过键盘输入或从文件导入进货的信息（货品名称，编号，进货数量，进价，售价，进货日期，厂家），存入库存文件 goods.txt。
2、卖货：通过键盘输入卖货的信息（编号，数量），修改库存文件 goods.txt。
3、统计：
(1)对同一货品名的商品，按照销售率（销售率=卖出数量/进货量*100%）对厂家排序，并生成相应的文件输出。
(2)统计所有货品的销售率，将销售率在 60%以上的商品按销售率排序并将信息输出。
(3)统计小商店的总利润
3、界面友好美观。
【提示】
（1）货品的信息包括：货品名称，编号，进货数量，进价，进货日期，厂家，售价，卖出数量
（2）货品名称可以相同，编号不能相同

4.期刊管理

【问题描述】
建立读者与期刊档案，能够随时统计期刊的借阅情况，查询期刊的入库情况，统计读者情况。管理人员能够录入新来期刊信息，借阅者借阅信息录入，能够随时统计期刊借出的情况，并根据借阅人的要求建立期刊借出预定表，等借出期刊还回时自动通知预定的借阅人。
【基本要求】
1、期刊查询：
（1）管理员能够查询所有期刊的信息，所有借阅人的信息
（2）借阅人可以根据期刊名称查询期刊的情况
2、期刊录入：管理员能够录入期刊的信息，录入借阅人的信息
3、借阅期刊：借阅人可以对未借出的期刊进行借阅，设置归还时间
4、归还期刊：对已借出的期刊根据借阅人设置的归还时间在系统中将该期刊自动设置成归还
5、预定期刊：借阅人可以对借出的期刊进行借阅预定，当该期刊归还时自动提醒该借阅人
【提示】
以上均为模拟过程，学生可以适当的丰富内容和限制条件

附录 1：课程设计报告范例-集合的并、交和差运算

数据结构课程设计报告

题目 编制一个演示集合的并、交和差运算的程序

班级 _____

姓名 _____

学号 _____

指导老师 吴聪聪、张有华、张敬敏、王楠、李霞、 李宁

完成日期 2020 年 1 月 3 日

一、需求分析

1. 本程序中，集合的元素限定为小写字母字符 ['a'..'z']，集合的大小 n<27。集合输入的形式为一个以"回车符"为结束标志的字符串，串中字符顺序不限，且允许出现重复字符或非法字符，程序应能自动滤去。输出的运算结果字符串中将不含重复字符或非法字符。

2. 演示程序以用户与计算机交互方式执行，即在计算机终端上显示"提示信息"之后，由用户在键盘上输入演示程序中规定的运算命令；相应的输入数据(滤去输入中的非法字符)和运算结果显示在其后。

3. 程序执行的命令包括：

(1)构造集合 1；(2)构造集合 2；(3)求并集；(4)求交集；(5)求差集；(6)结束。

构造集合 1 和构造集合 2 时，需以字符串的形式键入集合元素。

4. 测试数据

(1) Setl="magazine", Set2="paper",
Setl ∪ Set2="egimnprz",
Setl ∩ Set2="ae", Setl-Set2="gimnz"

(2) Setl="0120per4a6tion89", Set2="error data",
Setl ∪ Set2="deinoprt", Setl ∩ set2="aeort", Setl-Set2="inp"

二、概要设计

为实现上述程序功能，应以有序链表表示集合。为此，需要两个抽象数据类型：有序表和集合。

1. 有序表的抽象数据类型定义为：

```
ADT  OrderedList
{
    数据对象：D={ai|ai∈CharSet, i=1,2,...,n,  n≥ 0}
    数据关系：Rl={<ai-1,ai>|ai-1,ai∈D,ai-1<ai,i=2,...,n}
    基本操作：
      InitList(&L)
      操作结果：构造一个空的有序表 L。
      DestroyList(&L)
      初始条件：有序表 L 已存在。
      操作结果：销毁有序表 L。
      ListLength(L)
      初始条件：有序表 L 已存在。
```

操作结果：返回有序表 L 的长度。

ListEmpty(L)

初始条件：有序表 L 已存在。

操作结果：若有序表 L 为空表，则返回 True，否则返回 False。

GetElem(L ,pos)

初始条件：有序表 L 已存在。

操作结果：若 1≤pos≤Length(L)，则返回表中第 pos 个数据元素。

LocateElem(L,e , &q)

初始条件：有序表 L 已存在。

操作结果：若有序表 L 中存在元素 e，则 q 指示 L 中第一个值为 e 的元素的位置，并返回函数值 TRUE，否则 q 指示第一个大于 e 的元素的前驱的位置 ，并返回函数值 FALSE。

Append (&L,e)

初始条件：有序表 L 已存在。

操作结果：在有序表 L 的未尾插入元素 e。

InsertAfter(&L,q,e)

初始条件：有序表 L 已存在，q 指示 L 中一个元素。

操作结果：在有序表 L 中 q 指示的元素之后插入元素 e。

ListTraverse(q,visit())

初始条件：有序表 L 已存在，q 指示 L 中一个元素。

操作结果：依次对 L 中 q 指示的元素开始的每个元素调用函数 visit()。

}ADT OrderedList

2. 集合的抽象数据类型定义为：

ADT Set{

数据对象：D={ai|ai 为小写英文字母且互不相同，i=1,2,...,n,0≤n≤26}

数据关系：R1={}

基本操作：Createset(&T,Str)

初始条件：Str 为字符串。

操作结果：生成一个由 Str 中小写字母构成的集合 T。

Destroyset(&T)

初始条件：集合 T 已存在。

操作结果：销毁集合 T 的结构。

Union(&T,SL S2)

初始条件：集合 S1 和 S2 存在。

操作结果：生成一个由 S1 和 S2 的并集构成的集合 T。

Intersection(&T,SL S2)

初始条件：集合 S1 和 S2 存在。

操作结果：生成一个由 S1 和 S2 的交集构成的集合 T。

Difference(&T,S1,S2)

初始条件：集合 S1 和 S2 存在。

操作结果：生成一个由 S1 和 S2 的差集构成的集合 T。

Printset(T)

初始条件：集合 T 已存在。

操作结果：按字母次序顺序显示集合 T 的全部元素。

}ADT Set

3. 本程序包含三个模块

1) 主程序模块

```
void main(){
    do
   {  集合的初始化：
      集合的并运算；
      集合的交运算；
      集合的差运算；
      .......
      退出
    }while(!退出)
}
```

2) 集合单元模块　实现集合的抽象数据类型；

3) 有序表单元模块　实现有序表的抽象数据类型；

各模块之间的调用关系如下：

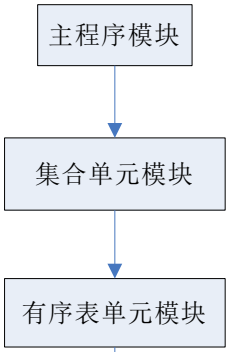


图 2-1　模块调用关系图

三、详细设计

1. 元素类型、结点类型和指针类型

```
typedef char ElemType; /*元素类型*/
typedef struct NodeType{
```

```
    ElemType data;
    NodeType    *next;
}NodeType,LinkType;   /*结点类型,指针类型*/
status MakeNode(LinkType &p,ElemType e)
{   /* 分配由 p 指向的数据元素为 e、后继为"空"的结点,并返回 TRUE,扩若分配失败,则返回 FALSE*/
        p=(LinkType)malloc(sizeof(NodeType);
        if(!p) return FALSE;
        p->data=e;
        p->next =NULL;
        return TRUE;
}
void FreeNode(LinkType &p)
{/* 释放 p 所指结点*/
}
LinkType Copy(LinkType p)
{/*复制生成和指针 p 所指结点有同值元素的新结点并返回,若分配空间失败,则返回空指针。新结点的指针域为 NULL */
        s=(LinkType)malloc(sizeof(NodeType))
        if(!s) return NULL;
        s->data=p->data;
        s->next =NULL;
        return s;
}
ElemType Elem(LinkType p)
{/*若指针 p!=NULL,则返回 p 所指结点的数据元素,否则返回'#'*/
}
LinkType SuccNode(LinkType p)
{
        /*若指针 p!=NULL,则返回指向 p 所指结点的后继元素的指针,否则返回NULL*/
}
```

2. 根据有序表的基本操作的特点，有序表采用有序链表实现。链表设头、尾两个指针和表长数据域，并附设头结点，头结点的数据域没有实在意义。

```
typedef struet{
        LinkType head,tail; /*分别指向线性链表的头结点和尾结点*/
        Int size; /*指示链表当前的长度*/
   }OrderedList; /*序链表类型*/
```

有序链表的基本操作定义如下：

```
bool InitList(OrderedList  &L);
    //构造一个带头结点的空的有序链表 L,并返回 TRUE;
    //若分配空间失败,则令 L.head 为 NULL,并返回 FALSE;
void DestroyList(OrderedList  &L);
    // 扩销毁有序链表 L
bool ListEmpty(OrderedList  L);
    // 若 L 不存在或为"空表"，则返回 TRUE，否则返回 FALSE
int ListLength(OrderedList  L);
    // 返回链表的长度
LinkType GetElemPos(OrderedList  L，tnt  pos);
    // 若 L 存在且 0<pos<L.size+1，则返回指向第 pos 个元素的指针，否则返回 NULL
bool LocateElem(OrderedList  L ,ElemType e ,LinkType &q);
    //若有序链表 L 存在且表中存在元素 e,则 q 指示 L 中第一个值为 e 的结点的位置，并返回 TRUE；否则 q 指示第一个大于 e 的元素的前驱的位置，并返回 FALSE
void Append(OrderedList &L, LinkType s);


    //在已存在的有序链表 L 的末尾插入指针 s 所指结点
void InsertAfter(OrderList  &L, LinkType  q, LinkType s);
    //已存在的有序链表 L 中 q 所指示的结点之后插入指针 s 所指结点
void ListTraverse(LinkType  p,status(* visit)(LinkType q));
    //从 p(p!=NULL)指示的结点开始，依次对每个结点调用函数 visit
```

其中部分操作的伪码算法如下:

```
bool InitiList(OrderedList  &L)
{
    if(MakeNode(head,''))
    {  //头结点的虚设元素为空格符"
        L.tail =L.head;  L.size =0;  return TRUE;
    }
    else
    {  L.head =NULL ;return FALSE ;}
}//InitList

void DestroyList(OrderedList &L)
```

```
{
    p=L.head;
    while(p){q=p; p=SuccNode(p); FreeNode(q);}
    L.head =L.tail =NULL ;
}//DestroyList

LinkType GetElemPos(OrderedList L, int pos)
{
    if(!L.head||pos<1||pos>L.size) return NULL;
    else    if(pos==L.size) return L.tail;
    else{
        p=L.head->next;
        k=1;
        while(p&&k<pos){ p=SuccNode(p); k++;}
        return p;
      }
}//GetElemPos;

status LocateElem(OrderedList L, ElemType e,LinkType    &p)
{
    if(L.head)
    {
        pre=L.head; p=pre->next;
            //pre 指向*p 的前驱,p 指向第一个元素结点
        white( p&&P->data<e)
        {
            pre=p; p=SuccNode( p);
        }
        if(p&&p->data==e) return TRUE;
        else{p=pre; return FALSE;}
    }
    else return FALSE;
}//LocateElem

void    Append(OrderedList    &L, LinkType s)
{
    if(L.head &&s)
    {
```

```
            if(L.tail!=L.head) L.tail->next =s;
            else    L.head->next=s;
            L.tail =s; L.size++;
        }
} //Append

void InsertAfter(OrderList    &L,LinkType    q,LinkType s)
{
   if(L.head&&q&&s)
   {
      s->next=q->next; q->next =s;
      if(L.tail ==q) L.tail =s;
      L.size++;
      }
}//InsertAfter

void ListTraverse(LinkType p, status (*visit)(LinkType q))
{
      while(p){ visit(p); p=SuccNode(p);}
}//ListTraverse
```

3.集合 Set 利用有序链表类型 OrderedList 来实现,定义为有序集 OrderedSet;

```
typedef OrderedList OrderedSet;
```

集合类型的基本操作的类 C 伪码描述如下:

```
void Createset(OderedSet    &T,char *s)
{//生成由串 s 中小写字母构成的集合 T,IsLower 是小写字母判别函数
if(InitList(T);     //构造空集 T
for(i=l; i<=length(s);    i++)
   if(islower(s[I])&&!LocateElem(T,s[I],p))
         //过滤重复元素并按字母次序大小插入
         if(MakeNode(q,s[i]))InsertAfter(T,p,q);
}// Createset
void Destroyset(OrderedSet &T)
{    //销毁集合 T 的结构
      DestroyList(T);
}//DestroyList

void Union(OrderedSet    &T,OrderedSet    S1,    OrderedSet S2)
```

```
{//求已建成的集合 Sl 和 S2 的并集 T,即:S1.head!=NULL 且 S2.head!=NULL
    if(InitList(T){
        pl=GetEiemPos(Sl,1);
        p2=GetElemPos(S2,l);
        while(pl&&p2)
        {
          cl=Elem(pl);    c2=Elem(p2);
          if(cl<=c2)
          {
              Append(T,Copy(pl);
              pl=SuccNode(pl);
              if(cl==c2) p2=SuccNode(p2);
          }
          else
          {    Append(T,Copy(p2)); p2=SuccNode(p2);    }
          while(pl)
              { Append( T,Copy(pl));    pl=SuccNode(pl);}
            while(p2)
              {Append(T,Copy(p2));    p2=SuccNode(p2);}
    }
}//Union

votd Intersection(OrderedSet    &T,OrderedSet    S1; OrderedSet S2)
{
    //求集合  Sl  和  S2  的交集  T
    if(!InitList(T)) T.head =NULL;
    else{
        pl=GetElemPos(S1,1);
        p2=GetElemPos(S2,l);
        while(pl&&p2){
        c1=Elem(p1);
        c2=Elem(p2);
        if(cl<c2) pl=SuccNode(pl);
        else if(cl>c2) p2=SuccNode(p2);
        else{    //cl==c2
            Append(T,Copy(pl));
            pl=SuccNode(pl);
            p2=SuccNode(p2);
```

```
                }//else
            }//while
        }// else
}//Intersection
void Difference(OrderedSet &T,OrderedSet S1,OrderedSet S2)
{//求集合 Sl 和 S2 的差集 T
        if(!InitList(T)) T.head =NULL;
        else {
            pl =GetElemPos(S1,l);p2=GetElemPos(S2,1);
            while(pl&&p2)
            {
                cl=Elem(pl);c2=Elem(p2);
                if(cl<c2){
                    Append(T,Copy(pl));
                    pl=SuccNode(pl)
                    else if(cl>c2) p2=SuccNode(p2);
                        else // Cl ==c2
                        {pl =SuccNode(p1);p2=SuccNode(p2);}
            }//while
            while(pl)
            {
               Apend(T,Copy(pl));
               p =SuccNode(pl);
            }
        }//else
}//Difference
void WriteSetElem(LinkType p)
{//显示集合的一个元素
        pramtk'Jh WriteElem(Elem(p));
}//WriteSetElem

votd Printset(OrderedSet T)
{//显示集合的全部元素
   p=GetElemPos(T,1);
   printf('[' );
   if(p){
          WriteElem(Elem(p);
          p=SuccNode(p);
```

```
    }
  ListTraverse(p,WriteSetElem());
  Prtntf(') ');
}//Printset
```

4. 主函数和其他函数的伪码算法

```
void main()
{//  主函数
      Initialization();   //初始化
      do{
            ReadCommand(cmd);//读入一个操作命令符
            Interpret(cmd);   //解释执行操作命令符
      }while(cmd!='q'&&cmd!='Q') ;
}//main
void Initialization()
{//系统初始化
  clrscr();   //清除在屏幕上方显示操作命令清单
              //Makesetl--------l
              //Maltesed--------2
              //Union-----------U
              //Intersaction-----I
              //Difference------d
              //Quit--------------q
              //在屏幕下方显示操作命令提示框;
  Createset(Set1,''); Createset(Set2,'');
}//Initialization
Printset(Set1); Printset(Setl);
//构造并显示空集 Set1 和 Set2 空集
void ReadCommand(char cmd)
{ //读入操作命令符,显示键入操作命令符的提示信息;
      do (cmd=getch())
      while (cmd['1','2','u', 'U', 'i', 'I','d','D,'q','Q']) ;
}
void Interpret (char cmd)
{//  解释执行操作命令 cmd
  switch(cmd)
  {
        case 'l':   //显示以串的形式键入集合元素的提示信息;
```

```
            scanf('读入集合元素到串变量',v);
            Createset(Set1,v);
            Printset(Setl);   //构造并显示有序集 Setl
            Break;
        case '2':   //显示以串的形式键入集合元素的提示信息;
            scanf('读入集合元素到串变量',v);
            Createset(Set2,v);
            Printset(Set2); //构造并显示有序集 Set2
            Break;
        case 'U':
        case 'u':
            Union(Set3,Set1,Set2);   //求有序集 Set1 和 Set2 的并集 Set3
            Printset(Set3);   //显示并集  Set3
            DestroyList(Set3);//销毁并集 Set3
            Break;
        case 'i':
        case 'I':
            Intersection(Set3,Set1,Set2); //求有序集 Setl 和 Set2 的交集 Set3
            Printset(Set3);
            Destroy(set3);
            break;
        case 'd':
        case 'D':
            Difference(Set3,Set1,Set2);//求集合 Setl 和 Set2 的差集 Set3
            Printset(Set3); DestroyList(Set3);
    }//switch
}//Interpret
```

5. 函数的调用关系图反映了演示程序的层次结构 ：

XXXXXXXXXXXXXXXXXXXX

XXXXXXXXXXXXXXXXXXXXX

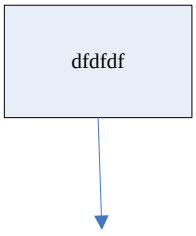


图 3-1 函数调用图

四、调试分析

1. 由于对集合的三种运算的算法推敲不足,在有序链表类型的早期版本未设置尾指针和 Append 操作,导致算法低效。

2. 刚开始时曾忽略了一些变量参数的标识"&"，使调试程序时费时不少。今后应重视确定参数的变量和赋值属性的区分和标识。

3. 本程序的模块划分比较合理,且尽可能将指针的操作封装在结点和链表的两个模块中，致使集合模块的调试比较顺利。反之，如此划分的模块并非完全合理，因为在实现集合操作的编码中仍然需要判别指针是否为空。按理，两个链表的并、交和差的操作也应封装在链表的模块中，而在集合的模块中，只要进行相应的应用即可。

4. 算法的时空分析

l)由于有序表采用带头结点的有序单链表，并增设尾指针和表的长度两个标识，各种操作的算法时间复杂度比较合理。InitList,ListEmpty,Listlength,Append 和 InsertAfter 以及确定链表中第一个结点和之后一个结点的位置都是 O(l)，DestroyList，LocateElem 和 TraverseList 及确定链表中间结点的位置等则是 O(n)的，n 为链表长度。

2)基于有序链表实现的有序集的各种运算和操作的时间复杂度分析如下：构造有序集算法 Createset 读入 n 个元素，逐个用 LocateElem 判定不在当前集合中及确定插入位置后，才用 InsertAfter 插入到有序集中,所以时间复杂度是 $O(n^2)$。

求并集算法 Union 利用集合的"有序性"将两个集合的 m+n 个元素不重复地依次利用 Append 插入到当前并集的末尾，故可在 O(m+n)时间内完成。

可对求交集算法 Intersection 和求差集算法 Difference 作类似地分析，它们也是 O(m+n)。

销毁集合算法 Destroyset 和显示集合算法 Printset 都是对每个元素调用一个 O(1)的函数，因此都是 O(n)。

除了构造有序集算法 CreateSet 用一个串变量读入 n 个元素，需要 O(n)的辅助空间外，其余算法使用的辅助空间与元素个数无关，即是 O(0)。

5. 本实习作业采用数据抽象的程序设计方法，将程序划分为四个层次结构；元素结点、有序链表、有序集和主控模块。使得设计时思路清晰，实现时调试顺利。各模块具有较好的可重用性。确实得到了一次良好的程序设计训练。

五、测试结果

执行命令 1

键入 magazine 后，构造集合 Setl[a,e,g,i,m,z]

执行命令 2

键入 paper 后，构造集合 Set2[a.e,p,r]

执行命令 d

构造集合 Setl 和 Set2 的并集 zh,e,g,i,mA 币 ,hz 〕

执行命令 I

构造集合 Setl 和 Set2 的交集 z[a,e]

执行命令 d

构造集合 Setl 和 Set2 的差集；

图 6-1　集合 1 数据输入

图 6-1（续）　集合 1 数据输入

六、附录 源程序文件名清单

Node.h　//元素结点单元。

List.h　//有序链表单元

Orderset.h //有序集单元

List.c//有序链表实现

Orderset.c //有序集实现

SetDemos.c　//主程序

课程设计报告排版要求

一、页面设置

纸张大小：A4 型（297mm×210mm）。

页边距：上为 2.5cm，左 2.5cm，下、右各为 2cm。

装订线：左侧 0.5cm。

正文行间距采用固定值 18 磅，字符间距采用标准设置、缩放 100%。

二、课程设计报告正文

一级标题：首行缩进两个字符，小三号、黑体、顶格；段前、段后间距各 0.5 行

正文：首行缩进两个字符，小四号宋体，行间距为 18 磅。伪码算法缩进不受此限制。

比如：

一、需求分析（一级标题，排版要求：首行缩进两个字符，小三号、黑体、顶格；段前、段后间距各 0.5 行）

提示：本节给出问题的描述、算法输入、算法输出。（此为正文，排版要求：首行缩进两个字符，小四号宋体，行间距为 18 磅）

三、图题

按大标题编号，字体宋体五号字居中。