数据结构与算法课程设计题目

(1) 马的遍历问题 难度：3- dfs(回溯)

**问题描述：**

在中国象棋盘上，对任意位置上放置一个马，均能选择一个合适的路线，使得该棋子能够按照象棋的规则不重复的走过棋盘上的每一位置。设计非递归实现。

**基本要求：**

（1）指定马的初始位置，输出马的行走路线。

（2）指定马的初始位置，输出马的多条以至全部行走路线。

（3）探讨每次选择位置的“最佳策略”，以减少回溯的次数。

（4）演示寻找行走路线的回溯过程。

（5）最好能画出棋盘的

形式，并在其上动态的标注行走过程。

(2) 英文单词的自动补全 难度：4 前缀树查找

**问题描述：**

在进行英文单词拼写时，很多时候会因为单词过长而无法正确拼写，导致工作效率下降，而很多输入法自动单词自动补全功能，当输入单词的前缀后，会出现几个可能的候选单词选项，大大提高了工作效率。

**基本要求：**

（1）英文单词由用户输入，候选单词的数量可以进行自行设置；

（2）使用合适的数据结构，来完成对巨量英文字典库的检索，以此实现；

（3）自行设置候选单词的出现策略，如随着用户的选择频率变化等等；

（4）鼓励使用图形化界面。

(3) 校园导游 难度：2- 图 最短路径

**问题描述**

用无向网表示校园景点平面图，图中顶点表示主要景点（建筑），存放景点的编号、名称、简介等信息，图中的边表示景点间的道路，存放路径长度等信息。要求能够回答有关景点介绍、游览路径等感兴趣的问题。

**基本要求：**

1)查询任意景点的相关信息；

2)查询图中任意两个景点间的最短路径。

3)查询图中任意两个景点间的所有路径。

4)增加、删除、更新有关景点和道路的信息。

5)指定某景点，查询感兴趣的一类景点，比如，查询距离某宿舍最近的食堂。

6)\*求多个景点的最佳（最短）游览路径。

(4) 教学计划编制问题 难度：3- 图 拓扑排序

**问题描述：**

大学的每个专业都要制订教学计划。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，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）、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；二是是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。

(5) 应用优先队列实现作业的优先调度 难度：3- 堆

**问题描述**

优先队列priority queue是一种可以用于很多场合的数据结构，设计并实现一个优先队列。应用该优先队列实现作业的优先调度：

一个作业ti =（si，ei），si为作业的开始时间（进入时间），ei为作业的结束时间（离开时间）。作业调度的基本任务是从当前在系统中的作业中选取一个来执行，如果没有作业则执行nop操作。本题目要求的作业调度是基于优先级的调度，每次选取优先级最高的作业来调度，优先级用优先数（每个作业一个优先数pi）表征，优先数越小，优先级越高。作业ti进入系统时，即si时刻，系统给该作业指定其初始优先数pi = ei - si，从而使越短的作业优先级越高。该优先数在作业等待调度执行的过程中会不断减小，调整公式为：pi = pi - wi，其中的wi为作业ti的等待时间：wi = 当前时间-si。一旦作业被调度，该作业就一直执行，不能被抢占，只有当前执行作业指向完成时，才产生下一轮调度。所以可以在每次调度前动态调整各作业的优先数。

编程实现这样一个作业调度系统。

**基本要求**

(1)分别以堆、左高树（或者其它更适合的结构）实现优先队列。

(2)作业集合中的各作业随机生成，根据作业的s属性和e属性动态调整作业队列，不断加入作业，作业结束删除作业。

(3)要对作业调度的结果给出清晰的输出信息，包括：何时作业进入，何时调度哪个作业，何时离开，每个作业等待多长时间，优先数的动态变化情况等。

(6) 媒体调度问题 难度：2 排序

**问题描述**

设一个多媒体播放系统用来播放一个video segment集合V，该集合中的每一个元素segment S=(B; L; F)，其中B为开始时间，L为媒体播放时间，即媒体长度，F为该媒体播放被延迟所造成的损失函数（此处可以假定F为关于延迟播放时间D的线性函数，如就定义F(S) = S.D，也可以自己设定）。

媒体播放时要求每个S的播放不能被中断（不可抢占），只能一个接一个的播放。本课程设计的基本目的是设计给出尽量少损失的调度算法。对于V={S1，S2，…，Sn}，V上的一个调度就是V集合中各元素的一个置换，即重新排列。

要求给出三个以上的调度方法（鼓励自己想出新的、更巧妙的调度方法），比如：

1. 最早开始时间最早调度 ，

将V中的各S按其B属性从小到大排序，排序结果就是调度结果。

②最短媒体最先调度

将V中的各S按其L属性从小到大排序，排序结果就是调度结果。

③ 贪心调度

保证每次调度都是此次调度引起最少损失的Si。

**基本要求**

① 随机的产生初始化V集合，V中的各个S也是随机产生的，即其中的B和L随机设定。

实现三种以上的调度算法实现调度，并计算比较各算法造成的损失。

1. 思考能完成最优调度的算法。

(7) 哈夫曼编码/译码器 难度：3-

**问题描述**

利用哈夫曼编码进行信息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。

一个完整的系统应具有以下功能：

1. 初始化（Initialization）。输入一个文本文件，统计文件中的字符及字符的频率，建立哈夫曼树，并将它存于文件hfmtree中。
2. 编码（Coding）。利用已建好的哈夫曼树（如不在内存，则从文件hfmtree中读入），对文本文件中的正文进行编码，然后将结果存入文件codefile中。
3. 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. 显示文本文件中的字符、字符频率、字符编码。
5. 打印代码文件（Print）。将文件codefile以紧凑格式显示在终端上。
6. 打印哈夫曼树（Tree printing）。将哈夫曼树以直观的方式显示在终端上。

**基本要求**

1. 定义并实现加权二叉树ADT，以及所应用优先队列的ADT，其中的基本操作根据应用设计；
2. 应用（1）中所实现的结构完成编译码系统的实现。

(8) 箱子装箱问题 难度：3- 竞赛树/平衡二叉树

**问题描述：**

在箱子装载问题中，有若干个容量为c的箱子和n个待装载入箱子中的物品。物品i需占s[i]个单元（0<s[i]≤c）。所谓成功装载（feasible packing），是指能把所有物品都装入箱子而不溢出，而最优装载（optimal packing）则是指使用了最少箱子的成功装载。

对于箱子装载问题，有4种流行的求解算法：

（1) 最先匹配法（First Fit, FF）

物品按1，2，⋯，n的顺序装入箱子。假设箱子从左至右排列。每一物品i放入可盛载它的最左箱子。

(2) 最优匹配法（Best Fit, BF）

设cAvail[j]为箱子j的可用容量。初始时，所有箱子的可负载容量为c。物品i放入具有最小cAvail且容量大于s[i]的的箱子中。

(3) 最先匹配递减法（First Fit Decreasing, FFD）

此方法与FF类似，区别在于各物品首先按容量递减的次序排列，即对于l≤i<n，有s[i]≥s[i+1]。

(4) 最优匹配递减法（ Best Fit Decreasing, BFD）

此法与BF相似，区别在于各物品首先按容量递减的次序排列，即对于l≤i<n，有s[i]≥s[i+1]。

编写一个程序比较以上四种方法（在时间上和所用箱子的数量上）的性能，

**基本要求**

（1）n依次取100,200,500,1000，比较以上四种方法（在时间上和所用箱子的数量上）的性能。

（2）定义并实现竞赛树结构ADT和AVL树结构ADT，结构上的而基本操作，根据本体的应用进行设计。

（3）FF FFD方法使用竞赛树结构，BF，BFD使用AVL树。

(9) 一元稀疏多项式计算器 难度：2 链表的操作 排序

**问题描述**

设计一个一元稀疏多项式简单计算器。

**基本要求：**

设计一个C++模板类Polynomial<T>，其中T 给出了系数的类型。类Polynomial 应该带有一个私有成员degree，它是多项式的阶数。当然，它还可能包含其他的私有成员。多项式类应支持以下操作：

(1) Polynomial() ——创建一个0阶多项式。这个多项式的阶数为0，不包含任何项。它是类的构造函数。

(2) Degree()——返回多项式的阶数。

(3) Input()——读入一个多项式。可以假定输入是由多项式的阶数和一个系数表构成，系数表中的系数按指数递增的次序排列。

(4) Output()——输出多项式。输出格式可以与输入格式相同或为类数学表达式；。

(5) Add(b)——把当前多项式加到多项式*b* 上，并返回所得结果。

(6) Subtract(b)——减去多项式*b* 并返回所得结果。

(7) Multiply(b)——乘以多项式*b* 并返回所得结果。

(8) Divide(b)——除以多项式*b* 并返回所得结果。

(9)Value(x)——返回按*x* 计算出的多项式的值。

对于(3) 至(9)，需要重载操作符< <、> >、+、-、\*、/ 和( )。对于(9)，语法P(x) 应返回多项式在x点的取值，其中P的类型为Polynomial。采用适当的多项式测试一元稀疏多项式简单计算器的基本功能。

(10)- LZW压缩算法及应用 难度：3 树/散列表

**问题描述：**

设计LZW压缩和解压缩程序。

(1)在一个文本文件上实现LZW压缩和解压缩，其中每个字符就是该文本的8位ASCII码。

(2)在实现LZW过程中需要仔细考虑如何在编译表中找到匹配或找不到匹配，需要注意匹配算法的时间、空间开销。

(3) 应用LZW算法实现256色灰度BMP图像文件的压缩和解压缩。

**基本要求 ：**

设计所使用字典的ADT，并给出实现，其中的基本操作根据本体的应用设计。

(11) 表达式类型的实现 难度：3 树

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

（2）WriteExpr(E)-用带括弧的中缀表达式输出表达式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（E2）。

(6) 合并常数操作MergeConst（E）—合并表达式E中所有常数运算。例如，对表达式E=（2+3-a）\*（b+3\*4）,进行合并常数操作后，求得E=（5-a）\*（b+12）

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，适当添加括号，以正确反映运算的优先次序。

(12) 等价类生成随机迷宫问题 难度：3 判断等价类 随机生成 图 最短路径

**问题描述：**

使用等价类来构造一个N\*N的从左上角到右下角的随机迷宫，然后在这一迷宫上寻找迷宫路径。该设计共包含如下是三个部分：

1. 构建随机迷宫，
2. 寻找迷宫路径，使用非递归形式。
3. 鼓励用图形方式画出随机迷宫及其上的最短路径。

**基本要求：**

1. 在生成随机迷宫时，考虑如何使选取墙的次数尽量少，并且任意两个方格之间都有路径。

(13)   文档集合上的查询 难度：3

**问题描述：**

设计数据结构完成在一个文档集合的存储，并构造算法实现其内容的查询。该设计包括三个部分：

(1)应用数据结构完成文档集合的内容(基于单词的)存储，并为下一步的查询建立索引。

(2)就单个单词的查询请求，设计算法进行查询。

(3)对多个单词通过AND和OR构造的复杂查询进行处理(此处可只做两个单词的情况)。

具体情形如下面的例子：

**Example**

*Doc1:* I like the class on data structures and algorithms.

*Doc2:* I hate the class on data structures and algorithms.

*Doc3:* Interesting statistical data may result from this survey.

Here are the answers to some queries:

*Query 1:* data

Doc1, Doc2, Doc3

*Query2:* data AND structures

Doc1, Doc2

*Query 3:* like OR survey

Doc1, Doc3

文档集合上的查询实例

**基本要求：**

① 文档集合中的文档数不能少于20个。

② 数据结构的设计以及查找算法的构造应考虑如何最大程度的提高查询效率。

③ 查询效率的提高应是综合多种查询的，而不是只针对一种查询的优化。

④ 给出查询效率的模拟实验数据。

实现提示：

AND和OR查询可转变为单个单词查询结果的组合。

(14) 单词拼写检查 难度：3 hash/前缀树等

**问题描述：**

现在有一些英语单词需要做拼写检查，在进行拼写检查之前首先建立一本词典。需要检查的单词，有的是词典中的单词，有的与词典中的单词相似，编写程序发现词典中与给定单词相同或相似的单词。

单词A与单词B相似的情况有三种：

(1)删除单词A的一个字母后得到单词B；

(2)用任意一个字母替换单词A的一个字母后得到单词B；

(3)在单词A的任意位置增加一个字母后得到单词B。

**基本要求：**

（1）定义并实现字典ADT，字典使用散列表表示。基本操作根据应用需要设定。

（2）尽可能高效地实现单词的拼写检查。

(15) 字符串类型的实现及应用 难度：3 kmp String类的操作

**问题描述：**

定义并实现字符串的ADT，其上的基本操作包括：赋值，求串长，查找字符在串中的位置，替换子串，串链接、串比较等（可参考STL中的字符串类型进行定义）。应用该串类型设计实现一个文本编辑器。

**基本要求:**

1. 不能使用STL中的类型；
2. 设计一个文本编辑器，使其具有通常编辑器(如Notepad)所应具备的基本功能。要求该编辑器在以上串类型上构建。
3. 编辑器应具备如字符串查找，字符串剪切，字符串粘贴，字符串替换，统计字数，统计行数等基本功能。

(16) 小大根交替堆实现双端优先队列及应用 难度：3

**问题描述：**

双端优先队列是一个支持如下操作的数据结构：

•insert (x) – 插入元素x

•ExtractMin () –删除最小元素

•ExtractMax () –删除最大元素

•Min () –返回最小元素

•Max () –返回最大元素

•Size() – 返回队列中元素个数

•Empty() –判队列是否为空

可用小大根交替堆来实现对上述操作的支持。小大根交替堆是一个满足如下小大根交替条件的完全二叉树：如果该二叉树不空，那么其上的每个元素都有一个称为关键字的域，且针对该关键字，二叉树按层次形成了小大根交替的形式，即对于小大根交替堆中的任何一个结点x，如果x位于小根层次，那么x就是以x为根节点的二元树中键值最小的结点，并称该结点为一个小根结点。同样的道理，如果x位于大根层次，那么x就是以x为根节点的二叉树中键值最大的结点，并称该结点为一个大根结点。在小大根交替堆中根结点位于小根层次。

**基本要求：**

(1)给出双端优先队列的ADT描述，包括优先队列的逻辑结构及其上基本操作。

(2)给出小大根交替堆的 ADT 描述，并实现该 ADT。

(3)以小大根交替堆结构为辅助结构实现双端优先队列的存储表示并实现其上的基本操作。

(4)应用双端优先队列的 ADT 实现依据学生成绩对学生信息的查询。

(5)学生信息存放在文本文件中（格式自定，内容自行输入）。

(17) Treap结构上的基本操作 难度：3

**问题描述：**

如果将n个元素插入到一个一棵二叉搜索树中，所得到的树可能会非常不平衡，从而导致上面的查找时间很长。一种通常的处理办法是首先将这些元素进行随机置换，即先随机排列这些元素，然后再依次插入到二叉搜索树中，此时的二叉搜索树往往是平衡的，这种二叉搜索树称为随机二叉搜索树。但是这样的处理存在一个问题，即这样的处理方法只适用于固定的元素集合（已经预先知道了所有的元素）。如果没有办法同时得到所有的元素的话，即一次收到一个元素，则没有办法进行处理，此时可以用Treap来处理这种情况。

Treap=Tree+Heap

Treap是一棵二叉搜索树，它的左子树和右子树分别是一个Treap，和一般的二叉搜索树不同的是，Treap结构中每个节点x有两个域，一个是其关键字值key，一个是优先级数priority(它是一个独立选取的随机数)，对于Treap结构，其关键字遵循二叉搜索树性质，其优先级遵循堆性质。但是这里要注意的是Treap和二叉堆有一点不同，就是[二叉堆](http://baike.baidu.com/view/668854.htm" \t "_blank)必须是完全二叉树，而Treap可以并不一定是。

**基本要求：**

(1) 设计和实现Treap结构以及如下的基本操作等。

·find(k): 查询关键字为k的元素；

·insert(x): 插入元素x;

·erase(k)删除关键字为k的元素；

·index(k):查询关键字为k的元素排名;

·get(index):查询排名为 index的元素.

(2) 编程实现随机二叉搜索树。

(3) 产生若干个依次插入二叉搜索树的元素序列(元素个数自己设定)，针对该序列分别实现三种情况：插入到一般二叉搜索树、插入到 Treap、用随机二叉搜索树处理。

(4) 分别得出三种情况处理后的结果二叉树的高度，进行实际的数据对比，从对比结果中发现一些规律。

(18) 二项堆的实现和分析 难度：4

**问题描述：**

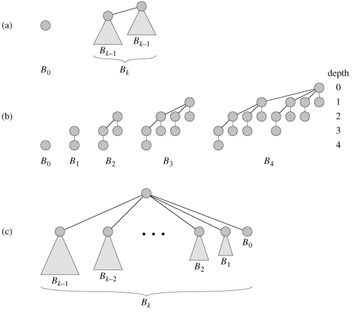
认识二项树、二项堆数据结构，并能应用该结构解决实际问题。

二项堆是二项树的集合，**二项树**是一种递归定义的有序树。它的递归定义如下：

(1) 二项树B0只有一个结点；

(2) 二项树Bk由两棵二项树B(k-1)组成的，其中一棵树是另一棵树根的最左孩子。

如下图所示：



图(a) 递归定义的二项树Bk，三角形表示有根子树。

图(b) 二项树B0 到B4 。

图(c)另一种角度看二项树Bk

二项树具有如下的性质：

1. 对于树Bk，该树含有2k个节点；

2.  树的高度是k；

3. 在深度为i中含有 节点，其中i = 0， 1，2 ... , k;

二项堆H是具备如下性质的二项树的集合：

1.  H中的每个二项树遵循最小堆性质；

2. 对于任意的整数k的话，H中最多有一个二项树的根的度数是k；

**基本要求：**

① 设计二项堆ADT，其上的基本操作包括：Make Heap (x), Find-Min, Union，

Insert，Extract-Min，Decrease Key (x)，Delete。

② 实现二项堆ADT，包括实现二项堆的存储结构以及其上的基本操作，并分析基本操作的时间复杂性。

1. 实现二项堆ADT的基本操作演示（要求应用图形界面）。

(19) 全国交通咨询模拟 难度：4 图的操作 如最短路径 遍历

**问题描述：**

处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地短，出门旅游的旅客则期望旅费尽可能省，而老年旅客则要求中转次数最少。编织一个全国城市间的交通资讯程序，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（如添加或删除）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（增设或删除）的功能。

(3)提供两种最优决策：最快到达和最省钱到达。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那一次班机到何地。

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进行编辑，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：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，表示边的结构内除含有邻接点的信息外，还应包括交通工具、路程中耗费的时间和花费以及出发和到达的时间等多种属性。

(3)增加旅途中转次数最少的最优决策。

(20) 求第K短的最短路径 难度：4

**问题描述：**

最短路径问题是图论中的一个经典问题，主要研究成果有Dijkstra、Floyd等优秀算法，Dijkstra算法一直被认为是图论中的好算法，但这两个算法都有一定的缺陷：这里的最短路径指两点之间最短的那一条路径, 不包括次短、再次短等路径。实际上，在使用咨询系统或决策支持系统时，希望得到最优的决策参考外，还希望得到次优、再次优等决策参考。这同样反映在最短路径问题上，如一个交通咨询系统，除了希望得到最短路径以外，由于交通堵塞等问题，可能需要获知次短、第K短的最短路径。因此，有必要将最短路径问题予以扩充，能求出第K短最短路径。形式的表述就是想要在图中求出从起点到终点的前k短的路径（最短、第2短、第3短……第k短），并且需要这些路径都是无环的。

**基本要求：**

1. 给定一个加权图，依次输出所有的前k短无环最短路径。

② 候选路径集合用堆存储实现，方便快速选取最短的一条。

③ 分析算法的时间复杂性。

实现提示：较好的求解前k短无环路径的算法是Yen算法

(21) 使用B+树实现图书管理基本业务 难度：4

**问题描述：**

定义并实现M序B+树的ADT，其上的基本操作包括：Search，Insert，Delete，和Ascend。

使用B+树设计一个图书管理系统，实现图书管理基本业务。基本业务包括：采编入库，清除库存、借阅，归还。

**基本要求：**

(1)每种书的登记内容包括书号、书名、作者、现存量和总库存量

(2)关键字为书号，图书管理基本业务都是通过书号进行的，所以要用B-树对书号建立索引，以提高查找效率。

(3)系统主要功能如下：

采编入库：新购一种书，确定书号后，登记到图书账目表中，如果表中已有，则只将库存量增加；

清除库存：某种树已无保留价值，将它从图书账目中注销。

借阅：如果一种书的现存量大于0，则借出一本，登记借阅这的书证号和归还日期，改变现存量

归还：注销对借阅者的登记，改变该书的现存量。

显示：显示每次操作后的B+树。

(22) 伸展树(Splay Tree )的实现和分析 难度： 4

**问题描述：**

伸展树是一种二叉搜索树,伸展树具有的特点是:当某个节点被访问时，伸展树会通过旋转使该节点成为树根。

**基本要求：**

（1）设计并实现伸展树(Splay Tree )的ADT，该ADT包括Tree的组织存储以及其上的基本操作：包括初始化，查找，插入和删除，拆分和合并等。并分析基本操作的时间复杂性。

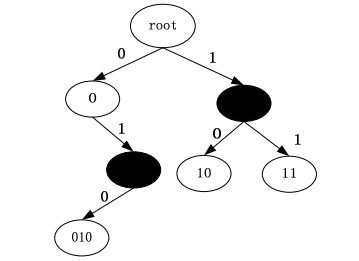
（2）实现伸展树(Splay Tree ) ADT的基本操作演示（鼓励应用图形界面）。

（3）采用伸展树，编写一个计算一文本文件中单词的出现频率并通过实验结果验证伸展树(Splay Tree )的操作性能。

(23) N-ary Trie的实现和分析 难度：4

**问题描述：**

哈夫曼算法输出的结果就是一个二元Trie，在二元Trie中，每个左子树分支用0表示，右子树分支用1表示。如下图就是就是一个二元Trie的示例图。

****

可以将二元Trie扩展到N-ary Trie。在N-ary Trie中，每个结点都有0~N之间任何个儿子结点，其中每个分支都用一个相应的符号表示（在N-ary Trie中有N个不同的符号）。例如一个电话本可用一个N-ary Trie表示，其中N=10，分别表示0~9的十个数字，具体情况如下图所示：



**基本要求：**

① 设计并实现N-ary Trie的ADT（N=26，建立在英语上的Trie），该ADT包括Trie的组织存储以及其上的基本操作：包括初始化，查找，插入，删除等。

② 应用Trie结构实现文本文档的索引化。首先扫描文本文档（存放在txt文件中），然后在此基础上用Trie登记单词出现的位置（行号），最后在Trie上实现查询。

③ 用户输入的查询可以是针对一个单词的，也可是针对一个字符序列的（以某几个字母开头的单词）。

(24) 逻辑表达式判别 难度：1

**问题描述**

一个逻辑表达式如果对于其变元的任意一种取值都为真，则称为重言式；反之，如果对于其变元的任意一种取值都为假，则称为矛盾式；然而，更多情况下，既非重言式，也非矛盾式。设计一个程序，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“|”、“&”、“~”，分别表示“或”、“与”、“非”，运算优先程度递增，但可以括号改变，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，用户选择输入求值或显示逻辑表达式的真值表，若用户选择输入求值，则用户对表达式中变元取定一组值，程序求出并显示表达式的值。

(25) 公交线路上的优化路径查询 难度：5

**问题描述**

最短路径问题是图论中的一个经典问题，其中的Dijkstra算法一直被认为是图论中的好算法，但有的时候需要适当的调整Dijkstra算法才能完成多种不同的优化路径的查询。

对于某城市的公交线路，乘坐公交的顾客希望在这样的线路上实现各种优化路径的查询。设该城市的公交线路的输入格式为：

线路编号：起始站名(该站坐标)；经过的站点1名(该站坐标)；经过的站点2名(该站坐标)；……；经过的站点n名(该站坐标)；终点站名(该站坐标)。该线路的乘坐价钱。该线路平均经过多少时间来一辆。车速。

例如：63：A(32,45)；B(76,45)；C(76,90)；……；N(100,100)。1元。5分钟。1/每分钟。 假定线路的乘坐价钱与乘坐站数无关，假定不考虑公交线路在路上的交通堵塞。对这样的公交线路，需要在其上进行的优化路径查询包括：任何两个站点之间最便宜的路径；任何两个站点之间最省时间的路径等等。

**基本要求：**

① 根据上述公交线路的输入格式，定义并建立合适的图模型。

② 针对上述公交线路，能查询获得任何两个站点之间最便宜的路径，即输入站名S，T后，可以输出从S到T的最便宜的路径，输出格式为：线路x：站名S，…，站名M1；换乘线路x：站名M1，…，站名M2；…；换乘线路x：站名MK，…，站名T。共花费x元。

③ 针对上述公交线路，能查询获得任何两个站点之间最省时间的路径（不考虑在中间站等下一辆线路的等待时间），即输入站名S，T后，可以输出从S到T的考虑在中间站等下一辆线路的等待时间的最省时间的路径，输出格式为：线路x：站名S，…，站名M1；换乘线路x：站名M1，…，站名M2；…；换乘线路x：站名MK，…，站名T。共花费x时间。

④ 针对上述公交线路，能查询获得任何两个站点之间最省时间的路径（要考虑在中间站等下一辆线路的等待时间），即输入站名S，T后，可以输出从S到T的考虑在中间站等下一辆线路的等待时间的最省时间的路径，输出格式为：线路x：站名S，…，站名M1；换乘线路x：站名M1，…，站名M2；…；换乘线路x：站名MK，…，站名T。共花费x时间。

实现提示 ：

需深入考虑，应根据不同的应用目标，即不同的优化查询来建立合适的图模型。

(26) 红黑树的实现和分析 难度：5

**问题描述：**

红黑树（Red Black Tree） 是一种自平衡二叉搜索树，是在进行插入和删除操作时通过特定操作保持二叉搜索树的平衡，从而获得较高的搜索性能。

**基本要求：**

（1）设计并实现红黑树（Red-Black Tree）的ADT，该ADT包括Tree的组织存储以及其上的基本操作：包括初始化，查找，插入和删除等。并分析基本操作的时间复杂性。

（2）实现Red-Black Tree ADT的基本操作演示（鼓励应用图形界面）。

（3）采用红黑树，编写一个小型的英汉词典索引，并实现简单的检索功能，词典可以手工建立，也可以网上寻找。

\*（4）实现红黑树的合并操作，并应用于字典的合并。

(27) 后缀数组的实现 难度：5

**问题描述:**

后缀数组是按字典顺序存放一个字符串所有后缀起始位置的一维数组。在后缀数组上可以进行折半查找。对于任意给定的字符串，设计并实现下列算法，并分析各个算法的时间复杂性。

**基本要求：**

a) 对任意给定的字符串S，建立其后缀数组；

b) 查找一个字符串S是否包含子串T；

c) 统计S中出现T的位置和次数；

d) 找出S中最长的重复子串。所谓重复子串是指出现了两次以上的子串。

(28) k-d树的实现与分析 难度：5

**问题描述：**

k-d树（k-dimensional树的简称），可以看作是二叉搜索树的多为推广，k-d树（k-dimensional tree），是一棵二叉树，树中存储的是一些k维数据。在一个k维数据集合上构建一棵k-d树代表了对k维数据集合构成的k维空间的一个划分。对于一维的情况，所有的点都在数轴上面，1-d树其实就是二叉搜索树。2-d树中的每一个节点使用两个关键字（或属性）来表征，在奇数层上的分支按照第一个关键字进行，而在偶数层上的分支按照第二个关键字进行。

用k-d树查找某个特定项（精确匹配查询），也可以进行范围查询。许多应用问题，都可归结为范围查询，比如，在校友数据库里查询1990到2010级的学生，或者查询IP介于161.111.68.1至161.111.68.255之间的在线节点，可以用k-d树完成这一类的搜索。

**基本要求：**

（1）设计并实现2-d树的ADT，该ADT包括Tree的组织存储以及其上的基本操作：包括初始化，查找，插入和删除等。并分析基本操作的时间复杂性。

（2）实现2-d树 ADT的基本操作演示（鼓励应用图形界面）。

（3）\*给出3-d树或k-d树的结构实现。

（4）选择一范围查询的应用实例，采用k-d树，编写实现的精确匹配查询和范围查询的功能，初始数据可以手工建立，也可以网上寻找。

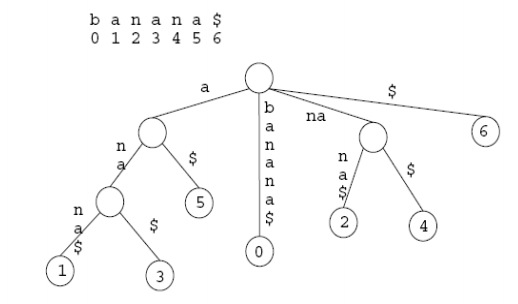
(29) 后缀树的构造 难度：5

**问题描述：**

后缀树是一种数据结构，一个具有m个字符的字符串S的后缀树T，就是一个包含一个根节点的有向树，该树恰好带有m+1个叶子(包含空字符)，这些叶子被赋予从0到m的标号。每一个内部节点，除了根节点以外，都至少有两个子节点，而且每条边都用S的一个子串来标识。出自同一节点的任意两条边的标识不会以相同的字符开始。

后缀树的关键特征是：对于任何叶子i，从根节点到该叶子所经历的边的所有标识串联起来后恰好拼出S的从i位置开始的后缀，即S[i,…,m]。

后缀树的图示:



**基本要求：**

1) 对任意给定的字符串S，建立其后缀树；

2) 查找一个字符串S是否包含子串T；

3) 统计S中出现T的次数；

4) 找出S中最长的重复子串。所谓重复子串是指出现了两次以上的子串；

5) 分析以上各个算法的时间复杂性。

6) 应用后缀树，查找两个字符串Q和R中最长的共有子串。分析时间复杂性并通过实验结果验证。

(30) 家谱管理系统 难度：4 树

**问题描述：**

实现具有下列功能的家谱管理系统。

**基本要求：**

1）输入文件以存放最初家谱中各成员的信息，成员的信息中均应包含以下内容：姓名、出生日期、婚否、地址、健在否、死亡日期（若其已死亡），也可附加其它信息、但不是必需的。

2）实现数据的文件存储和读取。

3）鼓励以图形方式显示家谱。

4）显示第n 代所有人的信息。

5）按照姓名查询，输出成员信息（包括其本人、父亲、孩子的信息）。

6）按照出生日期查询成员名单。

7）输入两人姓名，确定其关系。

8）某成员添加孩子。

9）删除某成员（若其还有后代，则一并删除）。

10）修改某成员信息。

11）要求建立至少20个成员的数据，以较为直观的方式显示结果，并提供文稿形式以便检查。

12）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。

13）存储结构：根据系统功能要求自行设计，但是要求相关数据要存储在数据文件中。测试数据：要求使用1、全部合法数据；2、局部非法数据。进行程序测试，以保证程序的稳定。

(31) 斐波那契堆 难度：5

**问题描述：**

斐波那契堆(Fibonacci heap)是最小堆有序树的集合。它和二项式堆有类似的性质，可用于实现合并优先队列。

**基本要求：**

（1）设计并实现斐波那契堆（Fibonacci heap）的ADT，该ADT包括Heap的组织存储以及其上的基本操作：包括初始化，合并两个堆，查找最小结点，插入结点，删除最小结点等。并分析基本操作的时间复杂性。

（2）实现Fibonacci heap ADT的基本操作演示（鼓励应用图形界面）。

（3）采用斐波那契堆对堆优化Dijkstra进行优化，对比普通的堆优化Dijkstra，分析斐波那契堆的实用价值。

(32) 使用哈希表实现通讯录管理系统 难度：3

**问题描述：**

利用哈希表完成通讯录的一般性管理工作。其中，每条记录至少包括、手机号、QQ、电子、地址等信息。分别以手机号和用户名为关键字建立不同的哈希表。

**基本要求：**

(1)添加信息：可新增人员信息；

(2)显示信息：按照哈希表的存储位置信息排序显示；

(3)查找：用名字和手机号分别作为查找的依据，进展查找；

(4)编辑信息：修改完善人员信息；

(5)删除信息：删除人员信息；

(6)界面友好，演示程序以用户和计算机的对话方式进展，可反复操作。

(7)利用外部.txt文件同步存储通讯录信息。

(33) 景区旅游信息管理系统 难度：3 图 最小生成树/最短路径

**问题描述：**

建立一个景区旅游信息管理系统，实现的主要功能包括制订旅游景点导游线路策略和制订景区道路铺设策略。任务中景点分布是一个无向带权连通图，图中边的权值是景点之间的距离。

**基本要求：**

（1）景区旅游信息管理系统中制订旅游景点导游线路策略，首先通过遍历景点， 给出一个入口景点，建立一个导游线路图，导游线路图用有向图表示；

（2）为了使导游线路图能够优化，需要判断路线图中是否存在回路；

（3）在导游线路图中， 还为一些不愿按线路走的游客提供信息服务， 比如从一个景点到另一个景点的最短路径和最短距离。在本线路图中将输出任意景点间的最短路径和最短距离；

（4）在景区建设中， 道路建设是其中一个重要内容。 道路建设首先要保证能连通所有景点，但又要花最小的代价，本任务中假设修建道路的代价只与它的里程相关。

（5）程序保证界面友好，提供菜单选项供用户选择不同的功能。

(34) 社会关系网络 难度：3 建图 图的遍历

**问题描述：**

在某社会关系网络系统中，一个人属性包括所在地区、就读的各级学校、工作单位等，没一个人有众多好友，并可以根据个人兴趣及社会活动加入到某些群组。现在需要设计一算法，从该社会关系网络中某一个人出发，寻找其可能认识的人。例如根据两个人共同好友数 量及所在群组情况，来发现可能认识的人；通过就读学校情况发现可能认识的同学。

**基本要求：**

（1）建议通过图形化界面，显示某一个人的社会关系网络。

（2）寻找某一人可能认识的人（不是其好友），并查看这些人与其关联度（共同好友数）。

（3）根据可能认识的关联度对这些人进行排序。

(35) 英语单词学习助手 难度：4 建立倒排索引

**问题描述：**

以大学英语相关英语文章为语料素材，设计有效的数据结构及其存储结构表示英语单词表，并建立相应的倒排索引，帮助英语学习者在遇到生词时能方便找到生词的相应例句，熟悉其应用语境与地道的用法；设计有效的算法对语料进行清理与分句处理，实现基于索引的快速例句搜索程序。

**基本要求：**

（1）输入某一个（或若干个）英语单词，要求返回相应的英语例句。

（2）根据单词与语句建立倒排索引，并且索引要求物化到外存，以文件形式保存，每次启动程序时不必重新建立索引，只需将索引文件导入内存。

（3）鼓励采用图形界面，便于输入单词，例句展现直观，界面布局合理。

**提示**：

按三步进行：（1）准备英语语料。寻找英语文章，可下载英语新闻，托福、GRE文章，或大学英语课文等。（2）处理语料。对语料进行清理、分句、索引、生成字典。需要进行取词干的操作，分句可以直接根据标点符号处理。（3）根据索引进行查询。支持一个或多个查询，基于对词干的处理，当查go、going等时也能够有返回。由于查询的结果是语句，如果直接按照词与文章的关系建立索引，这样需要从文章中找句子，太多的串匹配操作可能导致查询较慢，所以要设计好索引的粒度。

(36) 售票处的服务系统 难度：3 链表/队列

**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：**

民航售票处的计算机系统可以为客户提供下列各项服务：

（1）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；

（2）承办订票业务：根据客户提出的要求（日期、航班号、订票数额）查询该航班票额情况，若尚有余额，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，则需要重新询问客户要求。若需要，可预约登记排队等候。

（3）承办退票业务：根据客户提供的情况（日期、航班、退票数额），为客户办理退票手续，然后查询该航班是否有人预约登记，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预约的客户。

（4）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它航线情况。

（5）还可以自己新增其他功能和服务。

(37) 购物网站信息管理 难度：3 链表 hash 排序

**问题描述：**

设计一个程序，对商铺信息管理，商铺信息包括：商铺编号，商铺名，信誉度（0-5），（商品名称1，价格1，销量1），（商品名称2，价格2，销量2），（商品名称3，价格3，销量3）…。

商品名称包括毛巾，牙刷，牙膏，肥皂，洗发水，沐浴露等6种以上商品，每个商铺具有其中事先确定若干商品及价格，由文件输入，销量初始为0。

**基本要求：**

（1）建立一个单向链表存储所有商铺信息（至少30个），以编号为序，编号从1开始递增，从文件中读取数据，并能将数据存储在文件。商铺信息结点的数据结构自行设计。

（2）可以增、删商铺。增加商铺，编号自动加一，插入链表尾部；删除商铺，以编号为准，并修改后续结点的编号，保持编号连续性。可增、删商品。

（3）查询某一种商品名称，建立一个双向循环链表，结点信息是包含该商品的所有商铺编号、商铺名、信誉度、商品名称、价格、销量，以信誉度从高至低，并按销量排序，并逐一显示。

（4）购买某一商铺的商品，修改单向链表中商品的信息的销量。

（5）建立一个顺序结构，按商品名Hash分配地址，存储当前每种商品总销量并输出。

（6）任何的商铺信息变化，实现文件存储。

(38) 用户登录验证系统 难度：3 平衡二叉树

**问题描述：**

在登录服务器系统时，都需要验证用户名和密码。用户输入用户名和密码后，服务器程序会首先验证用户信息的合法性。由于用户信息的验证频率很高，系统有必要有效地组织这些用户信息，从而快速查找和验证用户。另外，系统也会经常会添加新用户、删除老用户和更新用户密码等操作，因此，系统必须采用动态结构，在添加、删除或更新后，依然能保证验证过程的快速。请采用相应的数据结构模拟用户登录系统，其功能要求包括用户登录、用户密码更新、用户添加和用户删除等。

**基本要求：**

（1）自己编程实现二叉树结构及其相关功能，以存储用户信息，不允许使用标准模板类的二叉树结构和函数。同时要求根据二叉树的变化情况，进行相应的平衡操作，即AVL平衡树操作，四种平衡操作都必须考虑。测试时，各种情况都需要测试，并附上测试截图。

**提示**：

（1）用户信息(即用户名和密码)可以存储在文件中，当程序启动时，从文件中读取所有的用户信息，并建立合适的查找二叉树；

（2）验证过程时，需要根据登录的用户名，检索整个二叉树，找到匹配的用户名，进行验证；更新用户密码时，也需要检索二叉树，找到匹配项后进行更新，同时更新文件中存储的用户密码。

（3）添加用户时，不仅需要在文件中添加，也需要在二叉树中添加相应的节点；删除用户时，也是如此。

(39) 低风险出行系统 难度：5 图的相关操作

**问题描述：**

城市之间有各种交通工具（汽车、火车和飞机）相连，有些城市之间无法直达，需要途径中转城市。某旅客于某一时刻向系统提出旅行要求。考虑在当前COVID-19疫情环境下，各个城市的风险程度不一样，分为低风险、中风险和高风险三种。系统根据风险评估，为该旅客设计一条符合旅行策略的旅行线路并输出；系统能查询当前时刻旅客所处的地点和状态（停留城市/所在交通工具）。

**基本要求：**

（1）城市总数不少于10个，为不同城市设置不同的单位时间风险值：低风险城市为0.2；中风险城市为0.5；高风险城市为0.9。各种不同的风险城市分布要比较均匀，个数均不得小于3个。旅客在某城市停留风险计算公式为：旅客在某城市停留的风险=该城市单位时间风险值\*停留时间。

（2）建立汽车、火车和飞机的时刻表（航班表），假设各种交通工具均为起点到终点的直达，中途无经停。不能太简单，城市之间不能总只是1班车次；整个系统中航班数不得超过10个，火车不得超过30列次；汽车班次无限制；

（3）旅客的要求包括：起点、终点和选择的低风险旅行策略。其中，低风险旅行策略包括：最少风险策略：无时间限制，风险最少即可；限时最少风险策略：在规定的时间内风险最少。

（4）旅行模拟系统以时间为轴向前推移，每10秒左右向前推进1个小时(非查询状态的请求不计时，即：有鼠标和键盘输入时系统不计时)；

（5）不考虑城市内换乘交通工具所需时间。

（6）系统时间精确到小时。

（7）建立日志文件，对旅客状态变化信息进行记录。

(40) 文本文件单词的检索与计数 难度：2

**问题描述：**

英文小说存于一个文本文件中，每个单词不包含空格且不跨行，单词由字符序列构成且区分大小写；统计给定单词在文本文件中出现的总次数；检索输出某个单词出现在文本中的行号、在该行中出现的次数以及位置。

**基本要求：**

1.建立文本文件，文件名由用户用键盘输入；

2.给定单词的计数，输入一个不含空格的单词，统计输出该单词在文本中的出现次数；

3.检索给定单词，输入一个单词，检索并输出该单词所在的行号、该行中出现的次数以及在该行中的相应位置。

4. 不能使用STL中的串类型；

(41) 拯救007 难度：1

**问题描述：**

在老电影“007之生死关头”（Live and Let Die）中有一个情节，007被毒贩捉住，并且关到一个湖中心的小岛上，湖中有很多鳄鱼。他用了一种极为大胆的方法逃脱 —— 直接踩着池子里一系列鳄鱼的头上跳上岸去！

设湖是边长为L米的正方形，中心坐标为 (0, 0)，且东北角坐标为 (L/2, L/2)。湖心岛是以 (0, 0) 为圆心、直径是D米的圆。给定池中分布的鳄鱼的坐标、以及007一次能跳跃的最大距离S，你需要给他指一条最短的到达湖边的路径 —— 他逃出去的路径的长度等于他跳的次数。

**基本要求：**

1. 设湖中鳄鱼数量 N（≤100），湖的边长L，湖心岛直径D，007一次能跳跃的最大距离S都为整数。

（2）每一条鳄鱼的 (x,y) 坐标由输入确定，x,y都为整数。注意：不会有两条鳄鱼待在同一个点上。

（3）如果007有可能逃脱，输出007需要跳跃的最少步数，以及从湖心岛到岸边的最短路径。如果最短路径不唯一，则输出第一跳最近的那个解，题目保证这样的解是唯一的。如果没可能逃脱，输出相应信息。

 (42) 网组识别 难度：1+

**问题描述：**

考虑N\*N网格，网格中的一些方格有黑色圆形占据，若两个方格共享一条边，则他们属于同一组。如下图所示。编程进行网格中组的识别。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | ● |
|  |  | ● |  |  |  |  | ● |
|  |  | ● |  |  |  |  |  |
|  |  |  |  |  | ● |  |  |
|  |  | ● |  |  | ● |  |  |
|  | ● |  |  |  | ● | ● |  |
|  |  |  |  |  |  |  |  |
|  |  |  | ● | ● |  |  |  |

**基本要求：**

1. N输入指定后，随机生成一个初始网格。初始网格以图形方式显示。
2. 当给出一个方格时，计算其所在组的大小，一个组的大小是该组含有的黑圆的个数。
3. 计算网格中不同组的个数。
4. 列出网格中所有的组。

(43) 银行业务模拟 难度：2

**问题描述：**

假设银行有K个窗口提供服务， 从早晨银行开门（开门9：00am，关门5：00pm）起不断有客户进入银行。由于每个窗口在某个时刻只能接待一个客户，对于刚进入银行的客户（建议：客户进入时间使用随机函数产生），首先前往取号机取服务序号，取号机会自动打印出一张服务单。单上显示服务号及该服务号前面正在等待服务的人数。当银行职员服务完一个顾客后，只需按呼叫器相应键，即可自动呼叫下一位顾客。

**基本要求：**

(1)编制一个程序以模拟银行的这种业务活动；

(2)计算一天中N位客户的在银行的平均等待时间、最长等待时间、最后完成时间，并且统计每个窗口服务了多少名顾客。

 (3)建议有如下设置：

  a)客户到达时间随机产生，一天客户的人数设定为N人。

  b)银行业务员对每一个顾客的处理时间随机产生，平均处理时间M分钟。

(44) 文本文件相似度计算 难度：2

**问题描述：**

实现一种简单原始的文件相似度计算，即以两文件的公共词汇占总词汇的比例来定义相似度。

**基本要求：**

**(**1)、为简化问题，不考虑中文，单词只由英文字母组成，不区分大小写，只考虑长度不小于m、且不超过n 的英文单词（比如只考虑长度不小于3、且不超过10 的英文单词），长度超过n 的英文单词只考虑前n个字母。

(2)、 输入一组文件对，每一文件对是两个文件名，则输出该组文件对的一组相似度。

(45) 凸包问题 难度：2

**问题描述：**

多边形（polygon）是指至少有三条直线边的平面封闭图形。凸多边形（convex polygon）的任意两个点（位于边线上或多边形内）之间的连线都包含在多边形内。平面上一个点集S的凸包（convex hull）是指包含所有这些点的最小凸多边形，该多边形的角是S的极点(extreme point)。寻找平面点集对应凸包的问题是计算几何中的基本问题。编写程序，求平面点集S对应的凸包。

**基本要求：**

**（1）**平面点集S的输入为点（序号，x坐标，y坐标）的序列，从键盘或文件输入，输出以点（序号，x坐标，y坐标）的序列形式或图形形式。

（2）点集分别采用两种结构表示。

(46) 稀疏矩阵运算器 难度：2

**问题描述：**

实现一个能进行稀疏矩阵基本运算的运算器。

**基本要求：**

扩充类LinkedMatrix，增加下列操作并重载操作符< <、> >、+、-、\*

1) 存储一个元素，已知行号、列号和元素的值。

2) 在矩阵中查找具有给定行号和列号的元素。

3) 两个稀疏矩阵相加。

4) 两个稀疏矩阵相减。

5) 两个稀疏矩阵相乘。

采用适当的稀疏矩阵测试并验证LinkedMatrix中各操作。

(47) 8数码问题 难度：2

**问题描述：**

一个8数码问题是将如图所示的初始格局中的小格子中的数字经过若干步移动后（只能水平和垂直移动到一个空白处）形成如图所示的目标格局。



**基本要求：**

① 针对8数码问题分别使用两种方法解决，并比较两种方法的区别。

实现提示：

可应用贪婪法和分支限界法。

(48) 长整数的代数计算 难度：2

**问题描述：**

定义并实现长整数类型ADT，基本操作包括长整数的加、减、乘、除等基本代数运算。

**基本要求：**

1. 长整数长度在一百位以上。
2. 实现两长整数的加、减、乘、除操作，除要返回商和余数。
3. 输入输出均在文件中。
4. 分析算法的时空复杂性。

(49) N皇后 难度：2

**问题描述：**

在NxN的棋盘上，国际象棋规定放置皇后时，任何一个皇后的水平、竖直和斜45º都不能有另一个皇后。

**基本要求：**

1) 输入任意整数N（N<15），输出此时所有的皇后摆放情况（图形形式）；

2) 若N=8，指定单个皇后的初始位置，判断此时能否得到合法的摆放，若能得到，给出摆放情况（图形形式）；

3) 请使用多种回溯算法来解决该问题，比较其优劣；

(50) 最小满覆盖 难度：2

**问题描述：**

在8×8的国际象棋棋盘上，如果在放置若干个马以后，使得整个棋盘的任意空位置上所放置的棋子均能被这些马吃掉，则称这组放置为棋盘的一个满覆盖。若去掉满覆盖中的任意一个棋子都会使这组放置不再是满覆盖，则称这一满覆盖为极小满覆盖。

**基本要求：**

1) 求解一个极小满覆盖；

2) 画出棋盘的图形形式，并在其上动态的演视试探过程。

3) 程序能方便移植到其他规格棋盘上，提高通用性。