

《数据结构与算法》课程设计大纲

（一）课程设计教学目的及基本要求

- 1、了解并掌握数据结构与算法的设计方法，具备初步的独立分析和设计能力；
- 2、初步掌握软件开发过程的问题分析、系统设计、程序编码、测试等基本方法和技能；
- 3、提高综合运用所学的理论知识和方法独立分析和解决问题的能力；
- 4、训练用系统的观点和软件开发一般规范进行软件开发，培养软件工作者所应具备的科学的工作方法和作风。

（二）课程设计内容及安排

- 1、问题分析和任务定义。根据设计题目的要求，充分地分析和理解问题，明确问题要求做什么？（而不是怎么做？）限制条件是什么？
- 2、逻辑设计。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。逻辑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功能说明)，各个主要模块的算法，并画出模块之间的调用关系图。
- 3、物理设计。定义相应的存储结构并写出各函数的伪码算法。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。详细设计的结果是对数据结构和基本操作作出进一步的求精，写出数据存储结构的类型定义，写出函数形式的算法框架。
- 4、程序编码。把详细设计的结果进一步求精为程序设计语言程序。同时加入一些注解和断言，使程序中逻辑概念清楚。
- 5、程序调试与测试。采用自底向上，分模块进行，即先调试低层函数。能够熟练掌握调试工具的各种功能，设计测试数据确定疑点，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，形成格式和风格良好的源程序清单和结果。
- 6、结果分析。程序运行结果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。算法的时间、空间复杂性分析。
- 7、编写课程设计报告。（具体格式参见附录课程设计报告格式。）

（三）课程设计考核方法及成绩评定

课程设计结束时，要求学生写出课程设计报告（附源程序），可运行的软件系统。课程设计成绩分两部分，设计报告占 30%，设计作品占 70%。

(四) 课程设计注意事项

1、课程设计报告要求

按格式要求完成实习报告，每人用 A4 纸打印课程设计报告（源代码可不打印），此外，请学习委员将所有同学的源代码收齐后刻在一张光盘上，光盘上分别为每位同学的资料建一个文件夹（名为“班学号姓名”，如 111161-01-XXX），其中存入该同学课程设计的源程序及课程设计报告的电子文档。

2、实习结束后一周内交。

3、第一次上机之前请各位同学对实现题目仔细研究，最好能有自己的思路。**第一次上机时**和指导老师交流对题目的理解。

4、请各位同学**自行完成**课程设计内容。**如有网上下载或同学间雷同，成绩将以不通过计 !!!**

5、请各位同学把实习题目当作产品来完成。追求完善，不要应付老师检查，程序要求有良好的结构及编码风格，有必要的注释。

6、为了减少调试时间，可以把测试数据及操作都使用文件存储，也可在“项目属性”中指定参数。

7、必须在**机房**完成实习，严格考勤！

《数据结构与算法》课程设计题目

1、数据结构综合应用

【内容简介】

本题是针对数据结构中常用的链表，数组，队列，哈希进行综合训练，共分四个小题，各小题之间无关联，要求分别设计独立的程序来完成。

【基本要求】

- (1) 要求使用给出测试数据文件或者自己编制输入数据文件，输出一律到文件中；
- (2) 要求算法的实现采用**类封装**完成，不使用类模板；
- (3) 四个题目整合到一个工程文件中；
- (4) 请各位同学在 <http://www.lintcode.com/zh-cn/> 网站上注册账号，报告中附出你的代码在网站上测试通过的截图（第三小题可以分开提交）

【提高要求】

- (1) 基于 Windows 对话框界面，可选择输入/输出文件名。
- (2) 采用 QT 或 MFC 开发环境。

1、K 组翻转链表【Lintcode 450】

给定一个链表以及一个 k,将这个链表从头指针开始每 k 个翻转一下。链表元素个数不是 k 的倍数，最后剩余的不用翻转。不允许引入额外的空间复杂度。

【输入数据】

文件第一行是组成链表的 N 个整数，逗号分隔。接下来的每行数据代表翻转数 k ($2 \leq k < N \leq 10000$)。

【输出数据】

翻转后的链表数据。

【样例输入】

1,2,3,4,5
2
3

【样例输出】

2,1,4,3,5
3,2,1,4,5

【测试数据】

输入数据自己随机产生，要求 $N > 100$ 。

2、最大矩形【Lintcode 510】

给定一个二维矩阵，权值为 False (0) 和 True (1)。要求找到一个最大的矩形，使得里面的值全部为 True，输出所有满足条件的子矩阵的起始/终止行/列号、面积。

【输入数据】

矩阵数据采用 TXT 文件形式给出，解析文件时不允许修改文件格式。

【输出数据】

文件逐行给出每个子矩阵的起始行号、起始列号，终止行号、终止列号、矩阵面积，以逗号隔开。例如：1,1,1,6,6

【测试数据】

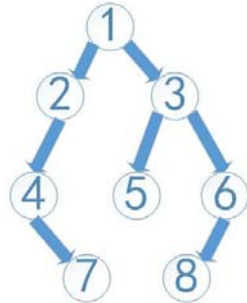
1.2-最大矩阵测试数据.txt

3、二叉树构建与最长距离【Lintcode 717 和 Lintcode 94】

给定一个二叉树，要求使用“队列”实现其节点值的交替层次遍历（先从左往右，下一层再从右往左，层与层之间交替进行），并使用“非递归方法”找出所有叶节点之间的最长路径。

【输入数据】

二叉树的层次遍历结果，例如：{1,2,3,4,#,5,6,#,7,#,8,#}



【输出数据】

交替层次遍历输出

```
[
  [1],
  [3,2],
  [4,5,6]
  [8,7]
]
```

最长路径：输出两个叶节点，及它们之间的路径。

(7,8) : (7,4,2,1,3,6,8)

4、重哈希【Lintcode 129】

哈希表容量的大小在一开始是不确定的。如果哈希表存储的元素太多（如超过容量的十分之一），我们应该将哈希表容量扩大一倍，并将所有的哈希值重新安排。假设你有如下的哈希表：

size=3, capacity=4

[null, 21, 14, null]

↓ ↓

9 null

↓

null

哈希函数为：

```
int hashCode(int key, int capacity) { return key % capacity; }
```

这里有三个数字 9, 14, 21，其中 21 和 9 共享同一个位置因为它们有相同的哈希值 $1(21 \% 4 = 9 \% 4 = 1)$ 。我们将它们存储在同一个链表中。重建哈希表将容量扩大一倍，我们将会得到：

size=3, capacity=8

index: 0 1 2 3 4 5 6 7

hash : [null, 9, null, null, null, 21, 14, null]

要求：给定一个哈希表，返回增容后的哈希表。

【输入数据】 [null, 21->9->null, 14->null, null]

【输出数据】 [null, 9->null, null, null, null, 21->null, 14->null, null]

2、JSON 编辑软件

【问题描述】

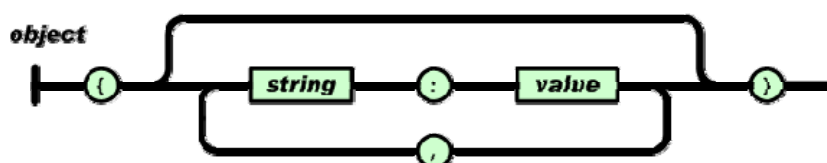
JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式, 采用完全独立于语言的文本格式, 已经成为当前服务器与 WEB 应用之间数据传输的公认标准。目前, 常见的 JAVA 版本的 JSON 类库有 Gson、JSON-lib 和 Jackson 等, C/C++ 版本的 JSON 类库也不少于 28 种。本课题要求自己设计并实现一款桌面级的 JSON 编辑器, 能实现 JSON 数据的生成、解析、path 查询等基本功能, 不允许直接使用“任何已有的 JSON 开发包或类库”。

【JSON 数据结构】

JSON 的两种常见的数据结构：

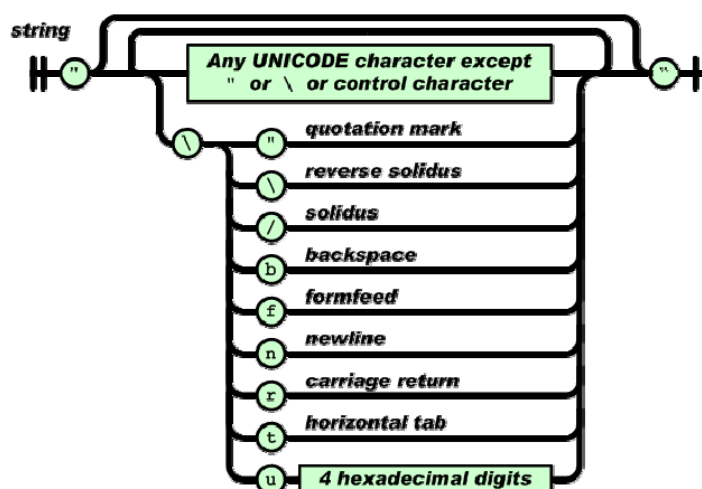
① 对象 (A collection of name/value pairs)

对象是一个无序的“名称/值”对集合。一个对象以“{” (左括号) 开始, “}” (右括号) 结束。每个“名称”后跟一个“:” (冒号); “名称/值”对之间使用“,” (逗号) 分隔。

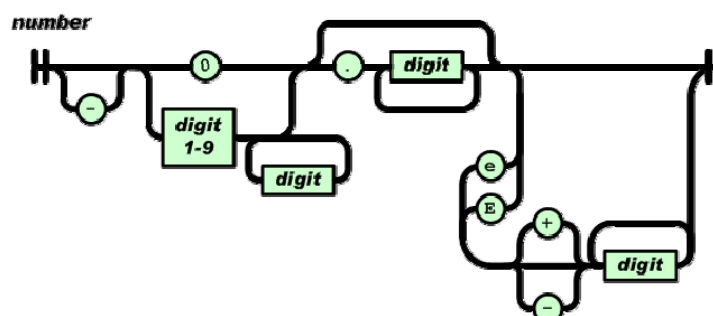


其中：

字符串 (*string*) 是由双引号包围的任意数量 Unicode 字符的集合, 使用反斜线转义。一个字符 (*character*) 即一个单独的字符串 (*character string*), 与 C/C++ 或者 Java 的字符串非常相似。

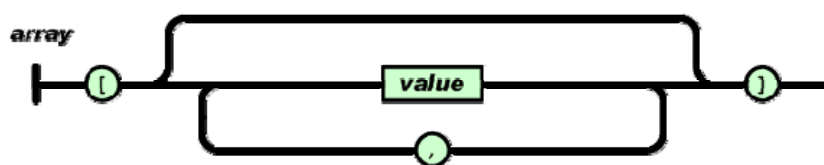


数值 (*number*) 也与 C/C++ 或者 Java 的数值非常相似, 除去未曾使用的八进制与十六进制格式。



② 数组 (An ordered list of values)

数组是值 (value) 的有序集合。一个数组以“[” (左中括号) 开始, “]” (右中括号) 结束。值之间使用“,” (逗号) 分隔。



其中, 值 (value) 可以是双引号括起来的字符串 (string)、数值(number)、true、false、null、对象 (object) 或者数组 (array)。这些结构可以嵌套。

【基本要求】

软件的基本功能如下：

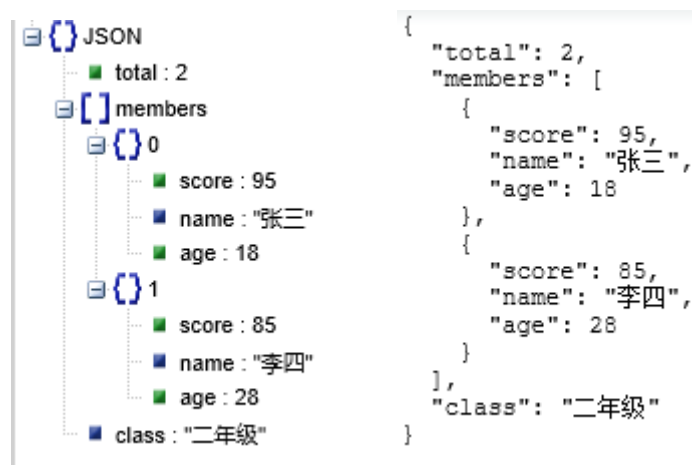
(1) 文件操作：提供基于磁盘的 JOSN 文件的创建、打开、保存、另存等功能。

(2) 编辑操作

提供 JOSN 的输入、编辑、删除、剪切、复制、粘贴等操作, 提供不少于 10 次的 Undo 与 Redo 操作。

(3) 多视图切换

提供多种可视化形式：树形、格式化、压缩 (清除所有空白字符)



```
{"total":2,"members":[{"score":95,"name":"张三","age":18},{"score":85,"name":"李四","age":28}],"class":"二年级"}
```

【提高要求】JOSN 查询

从 JOSN 文本中快速抽取指定信息 (如对象的某属性值), 支持单条件查询和复合查询。要求在软件界面输入满足 JsonPath 语法规则的表达式, 输出查询结果。语法规则如下:

JsonPath	描述
\$	根节点
@	当前节点
.or[]	子节点
..	选择所有符合条件的节点
*	所有节点
[]	迭代器标示, 如数组下标
[,]	支持迭代器中做多选
[start:end:step]	数组切片运算符
?()	支持过滤操作
()	支持表达式计算

符号	描述
\$	查询的根节点对象，用于表示一个 json 数据，可以是数组或对象
@	过滤器断言（filter predicate）处理的当前节点对象，类似于 java 中的 this 字段
*	通配符，可以表示一个名字或数字
..	可以理解为递归搜索，Deep scan. Available anywhere a name is required.
.<name>	表示一个子节点
['<name>' (, '<name>')]	表示一个或多个子节点
[<number> (, <number>)]	表示一个或多个数组下标
[start:end]	数组片段，区间为[start,end),不包含 end
[?(<expression>)]	过滤器表达式，表达式结果必须是 boolean

例如：

① 输出所有 author 的信息，"\$\$.store.book[*].author"

② 输出所有 price 的值，"\$\$.price"

【测试数据】

(1) 基本测试

对象：{"score":95,"name":"张三","age":18}

数组：[{"score":95,"name":"张三","age":18}, {"score":85,"name":"李四","age":28}]

嵌套：{"total":2,"members":[{"score":95,"name":"张三","age":18}, {"score":85,"name":"李四","age":28}], "class":"二年级"}

(2) 查询器测试

{"store":{"book":[{"category":"reference","author":"Nigel Rees","title":"Sayings of the Century","price":8.95}, {"category":"fiction","author":"Evelyn Waugh","title":"Sword of Honour","price":12.99,"isbn":"0-553-21311-3"}], "bicycle":{"color":"red","price":19.95}}}

查询条件：

JSONPath	Result
<code>\$.store.book[*].author</code>	the authors of all books in the store
<code>\$.author</code>	all authors
<code>\$.store.*</code>	all things in store, which are some books and a red bicycle.
<code>\$.store..price</code>	the price of everything in the store.
<code>\$.book[2]</code>	the third book
<code>\$.book[(@.length-1)]</code> <code>\$.book[-1:]</code>	the last book in order.
<code>\$.book[0,1]</code> <code>\$.book[:2]</code>	the first two books
<code>\$.book[?(@.isbn)]</code>	filter all books with isbn number
<code>\$.book[?(@.price<10)]</code>	filter all books cheaper than 10
<code>\$.*</code>	all Elements in XML document. All members of JSON structure.

测试用例可以使用：<https://zhuanlan.zhihu.com/p/30188199>

【实现提示】

(1) 剪切、复制、粘贴等操作使用 Windows 的 Clipboard 功能。

(2) 可以参考的软件：<http://jsonviewer.stack.hu/> 在线编辑；JsonPath 路径解析。

3、号码助手软件

【问题描述】

在很多实际应用中，动态索引结构在文件创建或初始装入记录时生成，在系统运行过程中插入或删除记录时，为了保持较好的检索性能，索引结构本身将随之发生改变。教材上已经介绍的动态查找数据结构包括：二叉搜索树（BST）、平衡二叉树（AVL）、红黑树（RBT）、B-树。本题要求选取一种已经学过的动态搜索树结构，设计并实现一个手机号码助手工具。

【基本要求】

一个完整的手机通讯录管理软件应具有以下功能：

(1) 支持复式联系人数据的存储，数据条目不少于 1000 条。

每个人名下可保存的信息包括：姓名、城市、手机号码、住宅电话号码、办公电话号码、电子邮件、公司、地址、所属群组、备注、添加时间等 11 个字段。

(2) 支持联系人记录的添加、删除、编辑等操作。

(3) 支持群组：将不同类型的人群按照城市、同事、朋友、家人、商务伙伴等分组，支持群组记录的添加、删除、编辑等操作。

(4) 支持所有联系人记录的导入、导出操作，外部数据采用 TXT 格式，内部数据采用自己设计的二进制数据文件格式。

(5) 支持联系人记录的各种灵活查询功能，具体包括：

① 逐条翻看

能显示所有的联系人记录，支持分屏查看。

② 多种方式查询

通过城市、添加时间、公司、地址、电子邮件、备注等任意字段都可搜索到联系人。

③ 电话号码查询

输入一个电话号码（手机、住宅、办公）的全部或者一部分，能将包含该号码的联系人记录显示出来。

④ 人名查找

输入一个人名（全名、部分名、拼音首字母、部分拼音），能将包含该姓名的联系人记录显示出来。

⑤ 群组查找

选择一种群组类型，能将属于该群组的所有联系人记录显示出来。

(6) 要求使用 BST 或者 AVL 实现动态索引结构。

【提高要求】

(1) 系统支持铃声库和图片库的数据存储，提供添加、删除、修改、播放等操作。铃声库和图片库可直接使用文件目录进行管理；铃声格式可使用 WAV、MP3 等格式；图片格式可使用 BMP、JPG 等格式。

(2) 联系人记录信息支持：来电铃声、来电图片等信息，用户可通过界面编辑或者浏览某条联系人记录的来电铃声、来电图片。

(3) 绚丽主题：支持换肤，提供多款精品皮肤主题。

(4) 使用红黑树或者 B-树的数据结构，来实现动态索引结构。

【测试数据】

号码助手数据中 3 个文件（1000、5000、10000 条联系人数据记录）。

【实现提示】

(1) 设计合适的二进制数据文件格式；

(2) 设计合适的索引文件格式。


4、拼图游戏软件

【问题描述】

打开任意一个图片，按照可以设定的切片数进行分割，并打乱排序；然后可以利用鼠标拖动切片到不同位置进行互换，直到拼合出原来图像。

这个问题其实可以简单表述成：3*3 的格子装了 1 至 8 的 8 个数字，数字是随机分布于各个格子中，是否可以利用空格的格子，移动装有数字的格子最终达到某种序列，如下图所示：

1	6	3
4		8
7	5	2



1	2	3
4	5	6
7	8	

实现原理：状态空间搜索，就是将问题求解过程表现为从初始状态到目标状态寻找这个路径的过程。由于求解问题的过程中分枝有很多，主要是求解过程中求解条件的不确定性，不完备性造成的，使得求解的路径很多这就构成了一个图，我们说这个图就是状态空间。问题的求解实际上就是在这个图中找到一条路径可以从开始到结果。这个寻找的过程就是状态空间搜索。常用的状态空间搜索有深度优先和广度优先，它们的缺陷在于都是在一个给定的状态空间中穷举。因此，需要使用启发式搜索。

启发式搜索就是在状态空间中的搜索对每一个搜索的位置进行评估，得到最好的位置，再从这个位置进行搜索直到目标，这样可以省略大量无谓的搜索路径以提到效率。在启发式搜索中，对位置的估价是十分重要的。启发中的估价是用估价函数表示的，如： $f(n) = d(n) + h(n)$ ，其中 $f(n)$ 是节点 n 的估价函数， $d(n)$ 实在状态空间中从初始节点到 n 节点的实际代价， $h(n)$ 是从 n 到目标节点最佳路径的估计代价。

在拼图游戏中，可以采用如下的评价函数， $f(n) = d(n) + h(n)$ ，其中 $d(n)$ 为当前状态从初始状态开始移动的步数， $h(n)$ 计算当前状态与目标状态相比错位的个数。搜索过程总是往 $f(n)$ 最小的分枝方向进行，以便快速达到最终状态。

【基本要求】

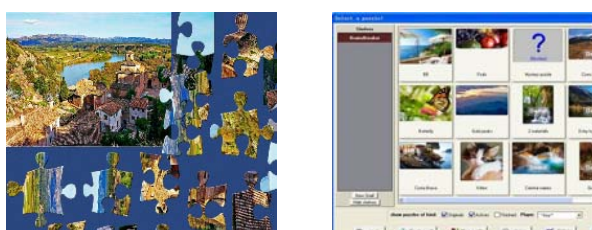
- (1) 支持 JPG、BMP 等文件格式，图片分割由系统自动完成，用户可以设置切片数量。游戏开始时，图片位置随机生成；
- (2) 支持“查看原图”，提供原图像的浏览，方便用户参考；
- (3) 用户可以拖动各个分片进行重新组合，直到拼合出原图像；系统能够自动检查到用户拼合出了原图像并提示用户；
- (4) 系统支持难度等级：初级、中级和高级，不同级别可设置移动次数和时间限制；
- (5) 界面美观，操作方便，音乐播放。除了鼠标操作，还支持键盘的上、下、左、右四个

【提高要求】

- (1) 系统有帮助功能，提示某个切片的位置；
- (2) 拼图形状可以不限于矩形（华容道游戏）。

【参考软件】

BrainsBreaker 5.7 以上版本（图片管理、选择、任意形状设置等）



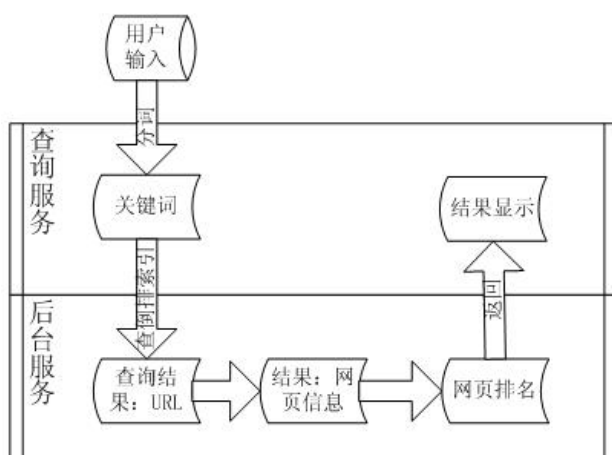
5、搜索引擎的实现（选做，可替换题目 3）

【问题描述】

实现一个搜索引擎通常需要完成下面三个步骤：

- 1) 将网络上所有网页爬下来，这也就是通常所说的网络爬虫；
- 2) 对所有爬下来的网页进行索引，以便更有效率的搜寻；
- 3) 对数据库中的每个网页进行重要程度的评价，最重要的网页会在搜索结果中排列在前面。

搜索引擎的工作流程如下图所示：



简单的查询服务过程如下：对于用户的输入，首先进行分词；对于每个词组，搜索倒排文件索引（Inverted File Index）获取包含该词组的网页 URL 信息，找到各个分词对应的 URL 集合中共同的 URL，根据结果 URL 集合查询网页索引获得 URL 对应的网页信息，整合网页信息之后进行返回。

本课题要求基于数据结构中的搜索树知识，针对给定的大量英文网页（>10 万）资料，采用图形化界面形式，设计并实现一款桌面级的搜索引擎，实现搜索引擎的第 2 和第 3 两个步骤。

【数据介绍】

数据集以文本文件形式给出，文本中含有有 10 万+网页文本构成，每个网页文本的内容结构如下：

```
P      http://blogs.abcnews.com/politicalpunch/2008/09/obama-says-mc-1.html
T      2008-09-09 22:35:24
Q      that's not change
Q      you know you can put lipstick on a pig
Q      what's the difference between a hockey mom and a pit bull lipstick
Q      you can wrap an old fish in a piece of paper called change
L      http://reuters.com/article/politicsnews/idusn2944356420080901?pagenumber=1&virtualb
L      http://cbn.com/cbnnews/436448.aspx
L      http://voices.washingtonpost.com/thefix/2008/09/bristol_palin_is_pregnant.html?hpid:
```

每一行的第一个字符，描述了本行数据的类型：

P: URL of the document

T: time of the post (timestamp)

Q: phrase extracted from the text of the document

L: hyper-links in the document (links pointing to other documents on the web)

注意：some documents have zero phrases or zero links（可能没有 Q 或者 L 项目）

数据网址：<http://snap.stanford.edu/data/memetracker9.html>

【基本要求】

- (1) 支持多个关键字（单词）的检索，单词不区分顺序。如：not change 和 change not 的检索结果应该相同。
- (2) 选择合适的动态索引结构（BST 树、B-树、B+树等）或者 倒排文件索引(Inverted File Index), 设计合适的磁盘索引数据文件格式。允许系统第一次启动时有一定的初始化时间（构造索引），后续启动应无等待时间。
- (3) 采用 Page rank (<https://en.wikipedia.org/wiki/PageRank>) 算法，根据网页间的链接情况计算每个页面的可信值；有多个网页满足查询结果时，显示出满足条件的网页总数，并且可信的网页排在前面。
- (4) 可视化界面，从界面进行关键字输入搜索，得到的结果显示在界面中（每页显示最可信的 10 个网页 URL 和内容，多条记录分页显示）。
- (5) 任意 2 个关键字的查询响应时间应该小于 1-2 秒。

【提高要求】

- (1) 系统启动后，直接显示出最可信的 10 个网页 URL 和内容。
- (2) 数据集（网页）内容可以动态增加，增量索引。
- (3) 索引建立时多线程技术的使用。

【测试数据】

- (1) quotes_2008-08-100k.txt
- (2) quotes_2008-08-560k.txt