数据结构与算法项目说明

图书索引处理

北京大学数学科学学院

2014 年 11 月 20 日

本项目基于真实的索引处理工具 makeindex [1, 2]、xindy [3] 等，我们将使用 Python 语言实现这些工具功能的一个子集。

在项目中，我们将熟悉文件处理类程序的组织和编写，可能会涉及表（数组）、集合、字典、栈、正则表达式等数据结构及排序等算法的灵活使用。

1 图书索引

篇幅较长的科技图书经常会有关键词索引。索引的格式是：将关键词按字母顺序分类排序，在关键词的后面标记出关键字所在的页码。本项目说明的末尾就有一个简单的名词索引。

在现代图书出版中，名词索引通常是使用如下方式生成的：

1. 作者在撰写图书时，在正文中标记出需要做索引的词汇。
2. 排版软件处理图书正文时，将标记出的词汇与词汇所在的页码从正文中提取出来，成为单独的索引文件。
3. 索引处理工具对提取出的索引文件按字母分类、排序，并且将相同的条目合并起来。

现在，我们就是要编写一个索引处理工具，完成上面的第 3 个步骤。

2 索引的输入

不同的排版软件可能生成不同格式的索引文件，为方便计，我们采用比较简单的一种格式，它也是计算机科学的经典著作《The Art of Computer Programming》的索引文件样式。

每个索引文件许多行构成，每行是一个索引条目。

一个索引条目分为两个部分，索引词和词条所在的页码。索引词与页码之间用一个空格与感叹号分隔开。简单的条目如：

```
Fibonacci sequence !13
```

它表示索引词是“Fibonacci sequence”（斐波那契数列），出现在全书的第 13 页。

索引词可能会在多处出现，因此同一个索引词可能会有多个页码；同一页的索引词也可能被多次标记。因此，有可能出现这样的索引文件：

```
Fibonacci sequence !13
Fibonacci sequence !13
golden ratio !13
Fibonacci sequence !27
```

有时，作者在书中的一整段内容都是关于一个名词的，因而会为这个名词标记一个页码区间。可以使用如下两种方式：

1. 在索引词的后面加上若干个加号，表示这个词条的会向后延续加号这么多页。例如：

   ```
   memory+ !4
   assertions+++ !15
   ```

   表示“memory”这个词条出现在第 4 页到第 5 页，而“assertions”词条会出现在第 15–18 页。

2. 在索引词的后面与前面加上两个减号，分别表示词条的开始与结束的位置。例如：

   ```
   arithmetic operators of MIX-- !131
   --arithmetic operators of MIX !133
   ```

   表示词条“arithmetic operators of MIX”从 131 页延续到 133 页。

索引条目的页码都是阿拉伯数字。但是，并不保证索引输入文件中的索引条目是按页码顺序排列的。

如果对项目进行扩充，可以在上述描述的基础上，支持更多或更复杂的输入格式。

3 索引的排序

索引处理程序的主要工作就是对索引进行排序。

3.1 索引词排序与分类

大体上，需要将索引词逐字符按字典序进行从小到大排序。但有如下特殊规则：

1. 如果字符串只包含阿拉伯数字（自然数），则数字大小比较。例如，“9”应该排在“10”之前，尽管按字符串序符号“1”在符号“9”之前。
2. 以符号开头的字符串总是先于排在以数字开头的串，而这又先于纯数字的排序项和以字母开头的串。

3. 在比较两个字符串时，索引处理程序应该首先忽略字母大小写进行比较，如果此时结果相等，再按区分大小写进行比较，将大写字母排在小写字母之前。

排序后的索引将进行分组。共分为 28 个组：完全由阿拉伯数字的组成的条目（自然数）分入“Numbers”组，以符号开头的条目（包括数字开头但不是纯数字的条目）分为“Symbols”组，其他条目按首字母分为 26 个组。不需要考虑希腊字母、汉字等无关的内容。

3.2 页码排序与合并

同一索引词的输入条目将会合并为一项，该项可能会有多个页码。

对合并后的每个索引项，需要对页码从小到大排序，去除重复的页码。例如，输入的索引项为

```
Fibonacci sequence !13
Fibonacci sequence !27
golden ratio !13
Fibonacci sequence !13
```

输出应只包含两项，形如：

Fibonacci sequence, 13, 27
golden ratio, 13

需要注意页码区间的合并，例如：

```
item-- !1
item !5
item++ !7
--item !10
item !10
```

应该直接合并为

item 1–10

并且注意相邻的页码区间也要合并为同一项，如

```
item-- !1
--item !3
item-- !3
--item !5
item !6
```

应合并为

item 1–6

如果对项目进行扩充，可以在上述描述的基础上，支持更多或更复杂的排序规则。

4 索引结果输出

索引处理程序在对读入的索引项排序整理后，需要将结果输出。

结果按 28 个分组分别进行输出，即按数字组、符号组、26 个字母组输出，输出时跳过没有条目的分组。输出每个分组时，先输出分组名称（Symbols, Numbers, A, B, . . . , Z），空一行后开始输出这一分组的条目。分组之间空两行。

对于每个条目，分别输出词条本身，与词条所在的页码。词条与页码之间、页码与页码之间用一个逗号和一个空格分隔。页码经过排序和合并，3 页或 3 页以上连续的页码区间在起迄页码中间用两个减号分开。词条最后输出一个西文句号。

下面是一个输出的例子：

```
Symbols

(), 164.
0-2-trees, 317.


Numbers

2048, 5.


A

Abel, Niels Henrik, 58, 498.
area of memory, 435.


B

Babbage, Charles, 1, 229.
Ballot problem, 536--537.
before and after diagrams, 260--261, 278, 281, 571.
```

如果对项目进行扩充，可以在上述描述的基础上，支持更多或更复杂的输出格式。

5 可能的扩展功能

索引程序可以有多方面的扩展，例如支持不同的输入格式，支持更复杂的结构等。

- 考虑如何在输入中进行转义处理。例如，当使用感叹号、加号、减号作为特殊字符时，如何处理索引条目中的感叹号、加号和减号。
- 考虑如何使输入输出的条目的格式变成可配置的，如让程序读入一个配置文件，在配置文件里面决定输入的特殊符号、输出的模板等。
- 考虑支持更复杂的结构，例如多级条目，一个条目下又可以有多个子条目。又如多级数字（可以表示章节编号而不只是页码）。
- 考虑支持更复杂的排序与分组规则。可以允许在条目中指定输出的文字与用来排序的文字不同，如希腊字母 α 可以按 alpha 来排序。设计对应的输入语法，输出的格式。
- 其他你认为有用的扩展功能。

其他可能的扩展功能可以看 makeindex [1, 2]、xindy [3] 等工具。

6 项目要求

完成下面的项目要求，将项目索引处理程序、测试用例文件、测试结果文件、项目报告一起打包，提交作业。

6.1 索引处理程序

编写索引处理程序，完成前述的基本功能，并在第 5 节中选择一两种扩展功能加以实现。
程序由若干个模块组成，其中包含一个 Python 脚本 `indexing.py`。脚本用法如下：

`python indexing.py` ⟨输入文件名⟩ ⟨输出文件名⟩

如果有扩展功能，可以在⟨输入文件名⟩之前增加其他的命令行参数。

6.2 索引表生成

自己设计一些用于测试的索引条目，生成索引表，验证程序的功能。你编写的测试示例应该能验证：

- 条目分组与排序的规则。
- 页码排序与合并的规则。
- 你设计的各种扩展功能。

随项目附有《The Art of Computer Programming》原书第一章的真实的索引数据文件 `taocp-1.idx`，请使用你编写的程序，根据此数据文件，完成索引表的生成。

6.3 项目报告

根据项目完成的情况，编写项目报告。报告内容包括：

- 对问题的简要描述与分析。应具体说明在你的项目中决定提供哪些功能，提供了哪些扩充。对于本项目说明没有详细说明的扩充功能，应详细分析其效果和用法。
- 程序模块划分与具体数据结构的设计。除了说明设计的结果，也要说明设计的原则与取舍过程。
- 程序实现中的关键问题和技术分析。
- 概述系统功能完成情况。分析测试用例的输出效果，加以适当说明。
- 程序重要算法的时间复杂性分析，及程序过程的空间复杂度分析。除了进行数学化的渐近复杂度分析，也可以给出实际运行的时间数据，说明程序的实际有效性。
- 进行系统完成后的回顾和分析。说明程序的优点、缺点，可能的改进问题，在编写项目中曾经遇到的困难与解决，走过的弯路等。

项目报告使用 Word 或 PDF 格式提交。报告写清楚标题、项目参与人的姓名学号、报告完成时间。报告应按内容合理划分成几个小节，注意算法、公式和代码的格式，使内容清楚，易于阅读。

名词索引

参考材料

[1] Leslie Lamport. *MakeIndex: An Index Processor For LaTeX*, February 1987. URL http://mirror.bjtu.edu.cn/CTAN/indexing/makeindex/doc/makeindex.pdf.

[2] Pehong Chen and Michael A. Harrison. Index preparation and processing. *Software—Practice and Experience*, 19(9):897–915, September 1988. ISSN 0038-0644. URL http://mirror.bjtu.edu.cn/CTAN/indexing/makeindex/paper/ind.pdf.

[3] Roger Kehr. *xindy Manual*, January, 2004. URL http://xindy.sourceforge.net/doc/manual.html