2020 年软件工程测试题——搜索关键词

本文档共分为三部分：题目描述、输入输出、提交要求与考核标准。

A. 题目描述

现代搜索引擎和社交网络服务可以通过用户的操作信息，为用户提供个性化推荐。在本题目中，你将为一个假想中的社交网络执行一个简单的数据分析推荐算法。具体来说，你需要读取日志文件，获取每个用户的搜索记录。用户的搜索记录保存在多个日志文件中。日志中的每一行表示一个用户搜索记录，格式为：“用户 ID<空格>搜索词”，其中用户 ID 为一个 32 位无符号整数，“<空格>”为一个空格，搜索词为一个只包含数字、大小写英文字母、下划线和短横线的单词。你需要对用户之间关系进行简要分析，并为每个用户态提供搜索词推荐。在本题目中，搜索词又被称为关键词。

步骤 1．读取日志文件，统计每个用户的搜索词次数；

结构定义和变量命名：

在本步骤中，你需要设计并使用一个名为 Record 的结构。每个 Record 结构包括搜索词 keyword 和其出现的次数 count。具体结构可根据语言而有所不同，但请保证满足上述命名要求。

你需要使用一个名为 userKeywordCounts 的变量保存所有的用户 ID 以及每个用户 ID 的所有搜索词与出现次数。其中搜索词与出现次数需使用前面定义的 Record 结构。

具体过程：

你需要将本步骤实现在一个名为 loadKeywordCounts 的函数。在此函数中，你需要按照规定格式读取给定的日志文件，并统计日志中出现的每个用户搜索过的每个词的次数。

信息输出：

为了方便调试并保证此步骤的正确性，每当读取完一个日志文件后，请在标准输出中打印当前已经统计过的日志总条数和已经出现过的不同用户总数量。两个数字以空格隔开，每个日志文件输出一行。

步骤 2．统计全局热搜；

结构定义和变量命名：

在本步骤中，你需要使用一个名为 ranking 的**数组类型**变量保存关键词排名。数组中每个关键词应保存在前述的 Record 的结构中。

具体过程：

你需要将本步骤实现在一个名为 rankKeywords 的函数中。你需要先根据步骤 1 中得出的 userKeywordCounts，统计每个关键词被所有用户搜索的总次数。

之后，根据总次数将所有关键词从大到小进行排序，最终结果保存在 ranking 变量中。若两个关键词被搜索的总次数相同，则根据两个关键词的字典序进一步比较，字典序小则排名靠前。

信息输出：

在此步骤中，请在标准输出中依次打印被搜索次数最多的前 5 个搜索词和其被搜索次数（即 ranking 中的前 5 项）。每个搜索词与其次数用一个空格隔开，占一行。本步骤共打印 5 行。

步骤 3．计算每两个用户间的关联度；

结构定义：

在本步骤中的结构应保存在一个名为 relations 的变量中。其为一个二维结构，relations[A][B]表示用户 A 和用户 B 的搜索关联度。

具体过程：

本步骤中，你需要实现一个名为 calcRelations 的函数计算每两个用户之间的搜索关联度。

对于任意两个用户，他们之间的搜索关联度定义如下：

$$relations[A][B] = \sum_{k} \Big(\min\big(Count(A,k), Count(B,k)\big) \times Weights(k)\Big)$$

其中 k 需要遍历所有出现过的关键词。Count(A，k) 为用户 A 搜索关键词 k 的次数；Count(B，k) 为用户 B 搜索关键词 k 的次数。Weights(k)为关键词 k 的权重，数值为：关键词总数减去关键词 k 在 ranking 中的序号（下标）。假设关键词共有 N 个，则 ranking[0]对应的关键词权重为 N，ranking[1]对应的关键词权重为 N-1，以此类推。

信息输出：

在此步骤中，请在标准输出中打印搜索关联度最高的**两个用户的 ID** 和他们之间的搜索关联度。两个用户 ID 按照先小后大的顺序输出，三个数字以空格隔开，占一行。如果有多对用户具有相同的搜索关联度，以用户的 ID 排序，取 ID 最小的一对进行输出。（先比较较小 ID，再比较较大 ID。）

步骤 4．为每个用户推荐关键词；

具体过程：

本步骤应实现在一个名为 calcRecom 的函数中。在本步骤中，你需要根据用户之间的搜索关联度为每个用户提供关键词推荐。对于用户 A，为其推荐的关键词由以下方法得出：

1．找到与用户 A 关联度最大的 3 名其他用户（B、C、D），如果关联度相同，则优先选择用户 ID 更小的用户，若不足 3 个则全部选择；

2．得出用户 B、C、D 搜索过的关键词的并集；

3．从该并集中去掉用户 A 搜索过的关键词；

4．在剩余的关键词中，取 ranking 中排名最靠前的 3 个单词，推荐给用户 A。不足 3 个则全部给出。

你需要将推荐结果保存在名为 “recommendations.txt”的文件中。

文件的每一行对应一名用户的推荐，格式为：“用户 ID<空格>推荐 1<空格>推荐 2<空格>推荐 3”。其中推荐的关键词以排名顺序进行排序，推荐 1 排名最靠前，推荐 2 排名次之，推荐三排名再次。**用户之间以用户 ID 从小到大排序。**

信息输出：

在此步骤中，请在标准输出中打印被推荐次数最多的搜索词和其推荐次数，以空格隔开，共占一行。如果两个搜索词被推荐次数相同，则取字典序较小者进行输出。

B．输入输出

题目的输入为 N 个日志文件，命名格式为“keywords-序号.log”，其中序号为从 1 到 N 的数字，如“keywords-2.log”。你的程序需要从标准输入读入数字 N，再从**当前工作目录下**读取日志文件。

你的程序需要输出一个名为“recommendations.txt”的文件。输出文件应保存在当前工作目录下。除了生成文件之外，你的程序还应在标准输出流中输出各个步骤的要求输出。

提示：由于使用当前工作目录，在程序中你可以直接使用“keywords-1.log”的形式作为文件名进行文件访问。在进行测试时，你可能需要将测试文件放在正确的路径下。

为了方便大家进行调试，我们将给出一个用于调试的测试样例，和一个小规模测试样例。对应的输入和输出文件将另行给出。

用于调试的测试样例：

样例输入

```
3
```

输入解释：表示有3个文件，分别为“keywords-1.log”、“keywords-2.log”和“keywords-3.log”。

样例输出

```
3 2
8 4
11 4
OnlineExam 4
WorkFromHome 3
SJTU 2
Cov2019 1
Exam-from-home 1
3 5 9
Cov2019 3
```

输出解释：

前3行为步骤1输出，分别表示读取完每个日志文件后的总日志条数和总用户数；

接下来5行为步骤2输出，表示排名前5的搜索词和他们的出现次数；

第9行为步骤3输出，表示用户3和用户5的关联度最大，为9；

第10行步骤4输出，表示Cov2019被推荐的次数最多（从下面的文件内容中可以看出Cov2019和Exam-from-home同样被推荐3次，但是Cov2019的字典序更靠前）。

下面为此测试中各个文件的内容：

```
Keywords-1.log
3 SJTU
3 WorkFromHome
5 WorkFromHome
Keywords-2.log:
4 Cov2019
5 WorkFromHome
7 OnlineExam
4 Exam-from-home
4 OnlineExam
keywords-3.log:
3 OnlineExam
5 OnlineExam
```

```
7 SJTU
recommendations.txt:
3 Cov2019 Exam-from-home
4 WorkFromHome SJTU
5 SJTU Cov2019 Exam-from-home
7 WorkFromHome Cov2019 Exam-from-home
```

小规模测试样例：

样例输入

```
3
```

样例输出

```
7 6
31 10
50 10
reconduction 6
senator 5
Corylaceae 4
Vilchez 4
ragmatical 4
3 9 52
Corylaceae 6
```

输出解释：

含义与上个测试用例相似，故不再赘述。

注意，此处的输出正确不代表程序正确。具体正确性还取决于生成的recommendations.txt文件内容。

C. 提交要求与评分标准

编码要求：

语言不限，但请按照题目要求进行设计和编码。

评分标准：

设计和实现程序，完成上述功能。如果不能完成全部程序功能，也请不要担心，我们会根据各

个方面和各步骤独立评分，具体标准如下：

1. 文件的正确读入和处理（步骤1）：15分；
2. 统计全局热搜（步骤2）：15分；
3. 计算关联度（步骤3）：15分；
4. 关键词推荐（步骤4）：15分；
5. 代码规范、命名标准、注释和说明等：10分；
6. 通过已给出的小规模测试占15分；
7. 通过大规模测试占15分。

提交要求：

请以将程序源代码使用ZIP压缩后命名为“Keywords_编号_姓名.zip”进行提交。其中编号为考生编号（申请编号），姓名为中文姓名。压缩包中应仅包含源代码，不包含测试文件、构建产生的中间文件或可执行文件。压缩包大小不超过1MB。

在考试结束之前，请确保将你的压缩包按照相应要求放在考试环境的桌面上。考试结束时，请退出考试环境（即断开连接），并等待收卷完毕。