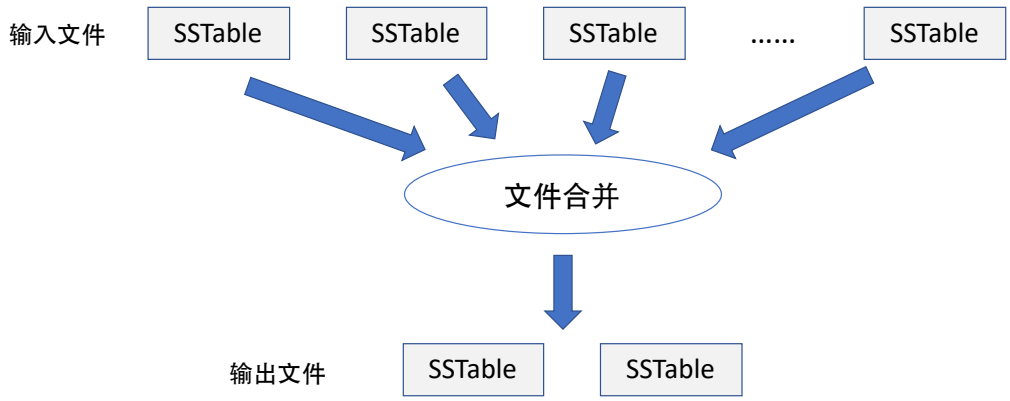
2020 年软件工程测试题

题目：文件合并

Log-structured Merge Tree （LSM树）是键值存储系统（Key-value Store）中常用的数据结构。两个热门的开源键值存储系统 LevelDB 和 RocksDB 均使用 LSM树作为核心数据结构保存其键值数据。

在LSM树中，键值数据以文件的形式保存在文件系统中，每个文件称为一个 SSTable。每个 SSTable 中以键（Key）的递增顺序保存键值对，以提高查找速度。同时，为了防止 SSTable 文件数量过多，LSM树会经常将文件系统中的多个 SSTable 文件进行合并（Compaction），并生成新的 SSTable 保存在文件系统中。

在本次测试中，你将模拟完成 SSTable 文件的合并操作。



A. 题目描述

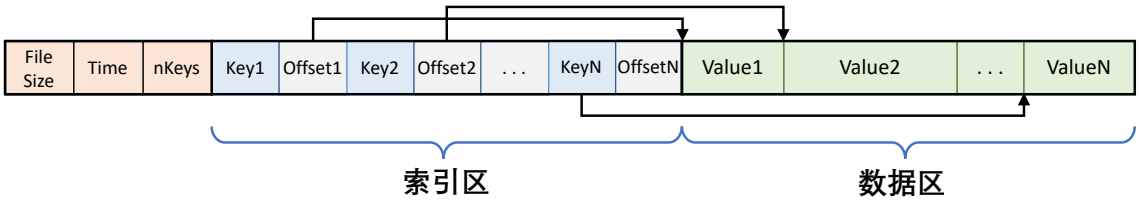
在本题目中，将给出一些SSTable文件作为输入。你需要将这些SSTable文件进行合并，并生成新的SSTable文件作为输出。

1) 文件格式

为简化题目，SSTable中的每个键值对记录包括键（Key）和值（Value）两部分。键为32位**有符号**整数；值为**字符串**。字符串中仅包含大小写字母和数字。字符串长度可以为0（表示键值对的删除记录，后面会详细介绍）。每个SSTable文件中的键

值对已经按照键（Key）的大小以递增顺序存放，且同一个文件中的键（Key）不会重复。

每个SSTable文件的格式如下图所示，文件中整数均为32位有符号整数（即占4个字节），并以小端模式（little endian）保存在文件中：



每个文件的前4个字节保存了该文件的大小 File Size（包括这4个字节）。随后4个字节保存了该文件被生成的时间 Time（此处为虚拟时间，仅用于比较两个文件的生成顺序。数字越大，说明文件生成的越晚）。再之后4个字节保存了该文件中的键值对数量 nKeys。此后的文件分为索引区和数据区两个区域。其中索引区以键递增的顺序，保存了每个键（Key）和其对应值在此文件中的偏移量（Offset）。如在下图中，Key1为2，Offset1为32，说明键2所对应的值保存在文件偏移量为32个字节处，又根据后面Key2的Offset2为42，得知Key1对应的值长度为10（即42−32）个字节。注意，文件中保存的值虽然为字符串，但是不以“\0”字符为结尾。

Key1=2	Offset1=32	Key2=31	Offset2=42
02 00 00 00	20 00 00 00	1F 00 00 00	2A 00 00 00

2) 操作步骤

在本测试中要实现的文件合并操作，包括以下4个步骤：

步骤1.　打开SSTable文件，并读取其中数据；

结构定义：

在本步骤中，你至少需要设计两个结构（本题目中的结构在不同语言中有不同的

指代，在C语言中为结构体，在C++/Java等面向对象语言中为类，请根据使用的语言进行选择）：

结构1：对于键值对，你需要设计一个名为KVPair的结构。其中**至少**包含键（key）和值（value）两个成员。

结构2：对于SSTable文件，你需要设计一个名为SSTable的结构。其中**至少**包含文件被生成的的时间（time）和一组键值对（pairs）两个成员。其中pairs成员应为简单的数组类结构（如C语言中的数组、C++中的vector/list，Java中的ArrayList等），其中保存多个KVPair结构。

具体过程：

本步骤的主体部分应实现在一个名为loadSSTables的函数中。即一旦该函数返回，则表示此步骤的所有操作已经完成。你可以将部分逻辑实现在其他函数中，并在该函数中进行调用。下同。

在本步骤中，你将根据前述的文件格式，读取给定的多个SSTable文件。对于每个SSTable文件，生成一个SSTable结构，并将SSTable文件中的键值对保存在pairs成员中的KVPair结构中。注意：你的pairs中保存的键值对顺序，应与键值对在文件中的顺序相同。由于每个SSTable文件中的键值对已经按照键（Key）的大小以递增顺序存放，pairs中保存的键值对同样应以递增的顺序存放。

信息输出：

为了方便调试并保证此步骤的正确性。请按照顺序读取每个SSTable文件，每个SSTable读取完毕后，请在标准输出中输出SSTable中保存的键值对个数、最小键（Key）和最大键（Key）。每个SSTable的此三个数字以空格隔开，每个文件占一行。

步骤2. 使用多路归并排序对多个SSTable文件的键值对进行排序；

结构定义：

本步骤中不要求设计新的结构，但是你需要使用一个名为sortedKVPairs的对象/变量。其类型应与SSTable结构中的pairs成员相同，用于保存排序后的所有键值对。

具体过程：

本步骤的主体部分应实现在一个名为sortSSTables的函数中。

由于每个SSTable中的键值对已经按照键（Key）以递增顺序进行存放，多路归并排序的方法如下：

1. 创建一个sortedKVPairs对象，初始为空；
2. 在所有SSTable结构中，找出当前最小的没有加入到sortedKVPairs的键值对。（键值对比较标准：Key越小，则越小；若Key相同，时间越小，则键值对越小）
3. 将最小的键值对加入到sortedKVPairs的末端；
4. 重复步骤2和3，直至所有键值对都加入到sortedKVPairs中。

由于每个SSTable中以升序保存键值对，且我们按照每次选最小的键值对放入sortedKVPairs中，排序后sortedKVPairs中保存的键值对应为非降序。

信息输出：

为了方便调试并保证此步骤的正确性。请在标准输出中输出排序后的最小键（Key）和最大键（Key）。两个数字以空格隔开，占一行。

步骤3.处理键值对的覆盖和删除操作；

结构定义：

本步骤中不要求设计新的结构，但是你需要使用一个名为cleanKVPairs的对象/变量。其类型应与sortedKVPairs相同，用于保存清理后的所有键值对。

具体过程：

本步骤的主体部分应实现在一个名为cleanSSTables的函数中。

由于多个文件中可能会存在多个相同键（Key）的键值对，在步骤2中生成的sortedKVPairs中会有一些重复的键值对。根据键值存储系统的语义，对于相同键（Key）的多个键值对，只保留最后一个（生成时间最晚，即时间最大）即可。若最后一个键值对的值长度为零，则说明此键（Key）被删除，无需保留该键（Key）的任何键值对。

在本步骤中，你需要根据以上规则扫描sortedKVPairs，将需要保留的键值对在保证顺序的情况下写入到cleanKVPairs中。

信息输出：

为了方便调试并保证此步骤的正确性。请在标准输出中输出清理后键值对的个数，最小键（Key）和最大键（Key）。三个数字以空格隔开，共占一行。

步骤4. 生成新的SSTable文件。

结构定义：

本步骤中不要求设计新的结构。

具体过程：

本步骤的主体部分应实现在一个名为saveSSTables的函数中。

在本步骤中，你需要将cleanKVPairs中的键值对以前述的文件格式保存在一个或多个SSTable文件中。

注意，每个SSTable文件的大小不得不超过256KB（即256*1024字节）。在生成SSTable文件时，应保证每个SSTable文件中尽可能多的保存键值对，但文件大小不得超过256KB。因此若一个文件不足以保存全部的键值对，则会产生多个SSTable文件。每个SSTable中保存的键值对仍以递增顺序保存；后生成的SSTable文件中的键（Key）大于先生成的SSTable中的所有键（Key）。

所有生成文件中的时间标记（Time）应固定为数字0x00FFFFFF。

生成的SSTable文件名应满足“output–序号.sst”的命名规则，如“output–1.sst”表示生成的第一个SSTable文件。

信息输出：

为了方便调试并保证此步骤的正确性。请在标准输出中输出生成的SSTable的个数。一个数字，共占一行。

B. 输入输出

题目的输入为N个SSTable文件。命名格式为“sstable-序号.sst”，其中序号为1到N的数字，如“sstable-2.sst”。

你的程序需要从标准输入中读入数字N，再从**当前工作目录**下读取SSTable文件。

你生成的SSTable文件名应满足“output-序号.sst”的命名规则，如“output-1.sst”表示生成的第一个SSTable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“sstable-1.sst”或“./sstable-1.sst”等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SSTable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“sstable-1.sst”、“sstable-2.sst”和“sstable-3.sst”。

样例输出

```
3 1 4
3 1 5
4 1 4
1 5
4 1 5
1
```

输出解释：

前3行为步骤1输出，分别表示每个输入文件的键值对个数，最小键和最大键；

第4行为步骤2输出，表示排序后的最小键为1，最大键为5；

第5行为步骤3输出，表示清理后共剩余3810个键值对，最小键为1，最大键为5；

第6行步骤4输出，表示生成1了个SSTable文件，名为“output-1.sst”。

在此测试用例中，各个SSTable文件中保存的键值对如下，其中(1, “a”)表示一个键为1，值为“a”的键值对。注意，为了便于大家理解，此处给出的文件内容与文件中

实际存储的方式不同。请以前述文件格式进行文件访问。

```
sstable-1.sst:
(1, “a”)
(2, “b”)
(4, “d”)
sstable-2.sst:
(1, “x”)
(2, “”)
(5, “e”)
sstable-3.sst:
(1, “y”)
(2, “z”)
(3, “c”)
(4, “”)
output-1.sst:
(1, “y”)
(2, “z”)
(3, “c”)
(5, “e”)
```

小规模测试样例：

样例输入

```
3
```

输入解释：表示有3个文件，分别为“sstable-1.sst”、“sstable-2.sst”何“sstable-3.sst”。

样例输出

```
4539 4 49988
7598 4 49988
6811 4 49988
4 49988
4639 28 49983
2
```

输出解释：

前3行为步骤1输出，分别表示每个输入文件的键值对个数，最小键和最大键；

第4行为步骤2输出，表示排序后的最小键为4，最大键为49988；

第5行为步骤3输出，表示清理后共剩余4639个键值对，最小键为28，最大键为49983；

第6行步骤4输出，表示生成2了个SSTable文件，名为“output-1.sst”和“output-2.sst”。

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SSTable文件内容。

C. 提交要求和考核标准：

编码要求：

语言不限。但请按照题目要求进行设计和编码。不可使用标准库算法库或其他类库中提供的排序算法。不可使用带有排序功能的容器（如std::map）。

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. 实现Key排序（步骤2）：15分；
3. 实现删除和合并（步骤3）：15分；
4. 输出文件的生成（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“Compaction_编号_姓名.zip”进行提交。其中编号为你所使用的上机考试环境的端口号（“200XX”的格式，其中XX为两位数字），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，并等待收卷完毕。