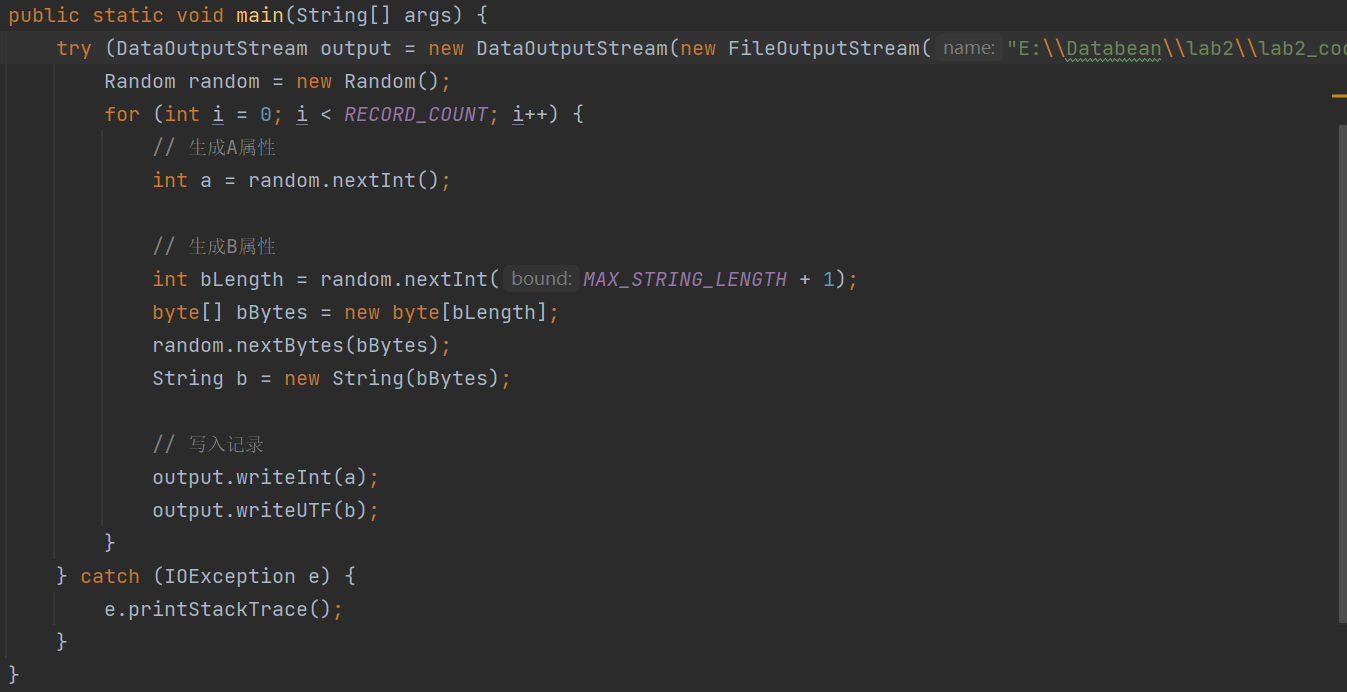
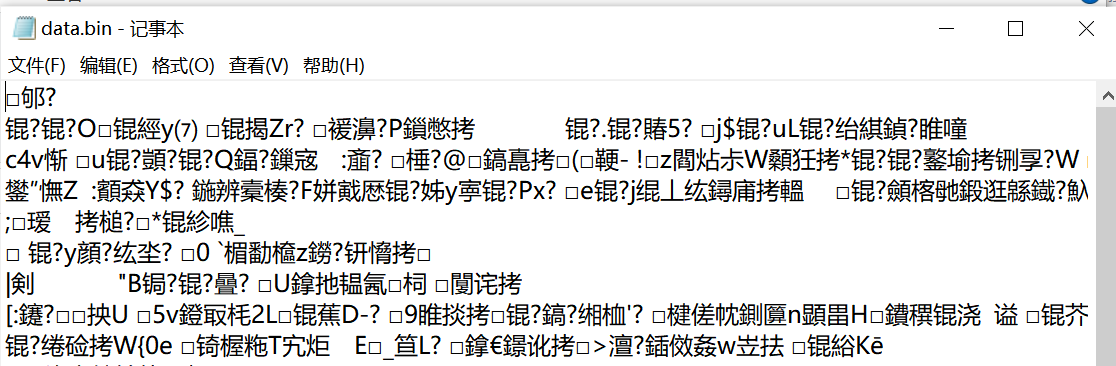
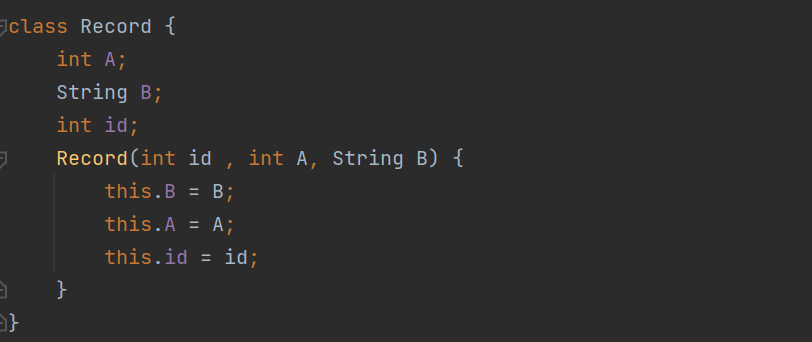
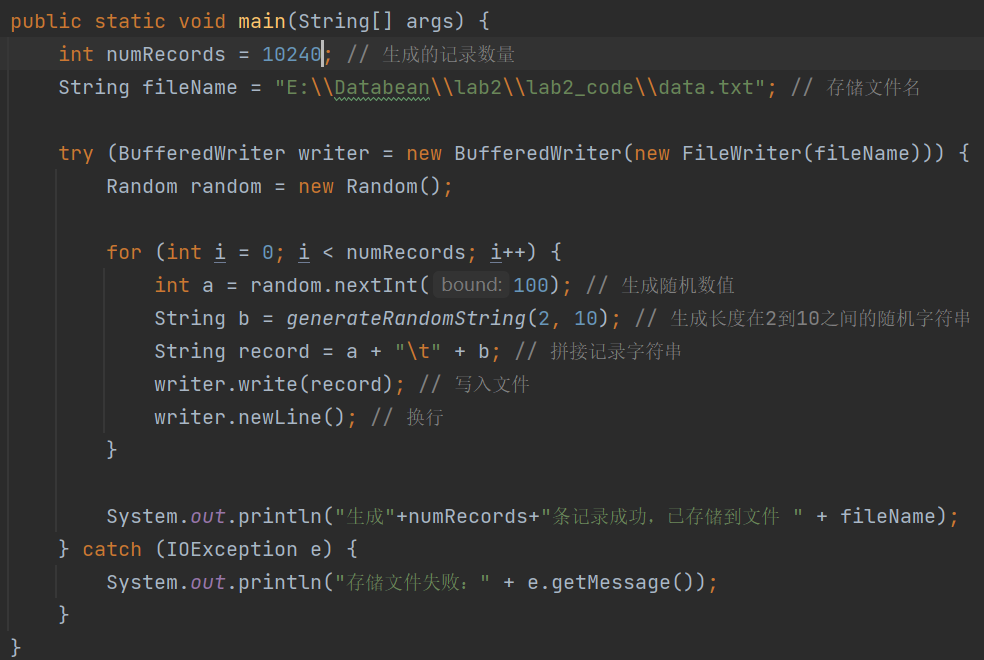
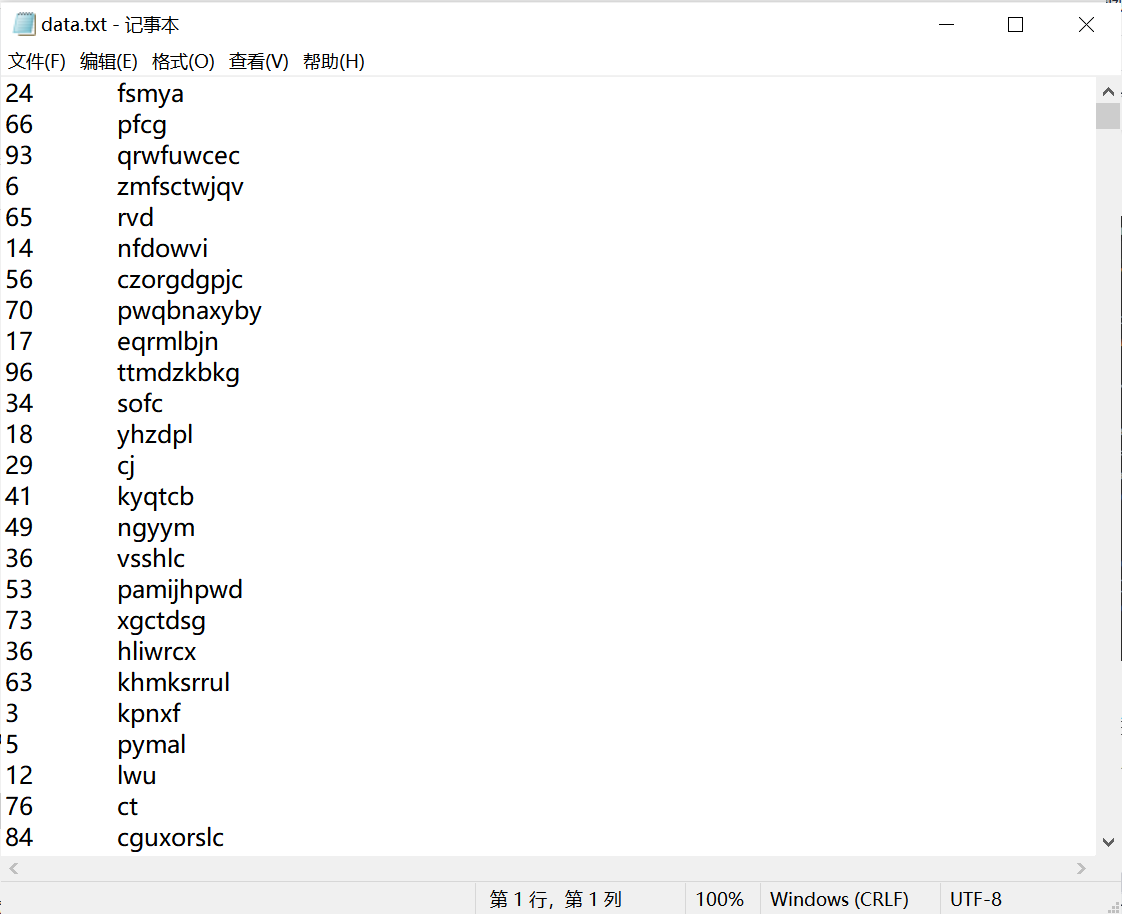
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **2023年春季数据库系统**  **实验二** | | | | |
|  | | | | |
| 数据库系统算法的实现 | | | | |
|  | 姓 名： | ： | 孙莹 |  |
| 学 号： | ： | 120L020414 |
| 班 级 ： | ： | 软件工程2037101 |
| 联系方式： | ： | 1099384634@qq.com |
| 二〇二三年5月 | | | | |

1. 描述实验环境的构建（记录和文件的准备）  
   1.实验要求记录格式至少包括2个属性（A和B），其中A为数值型，B作为记录的内容类型不限。编写代码，随机生成正整数作为A属性，随机生成正整数转为byte数组的形式作为B属性，并将其写入二进制形式的文件。

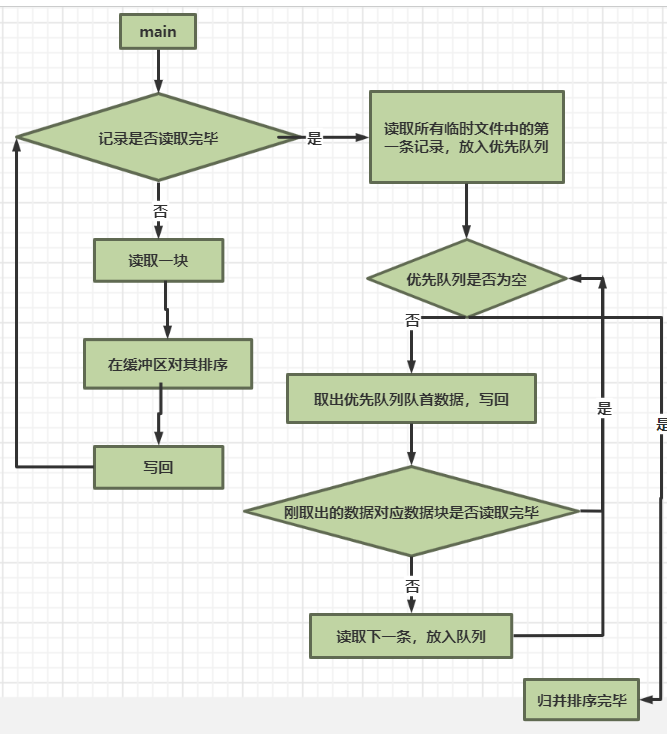


2.由于我在实验过程中发现二进制格式.bin的文件需要专门的编辑器读写，直接在记事本中打开会像下面一样为乱码。  


所以我又编写了程序，生成随机正整数作为A属性，生成随机长度的英文字符串作为B属性，然后写入txt文件。这样做方便检查程序效果。  


  
3.运行程序，生成了我们需要的记录文件  


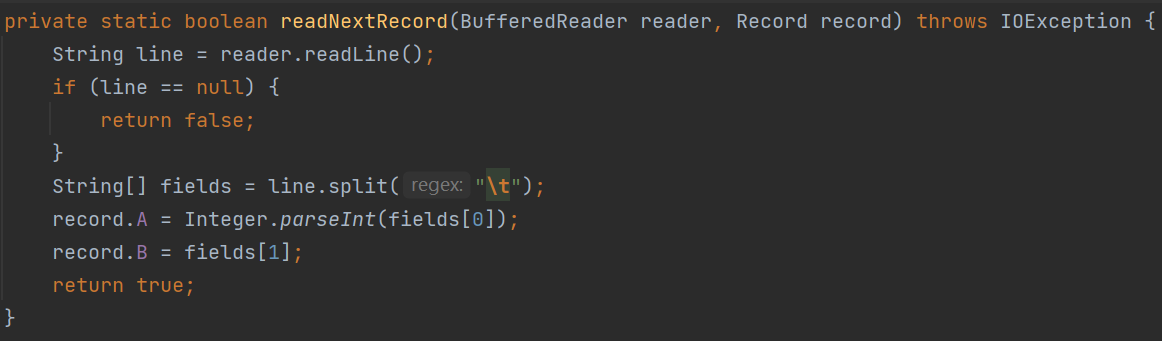
1. 基本算法设计的流程图  
    主要实现思路是将大文件拆分为多个较小的临时文件，对每个临时文件中的记录进行排序，然后通过优先队列进行 k 路归并，最终将排序结果输出到一个新的文件中。  
    对应流程图如下：



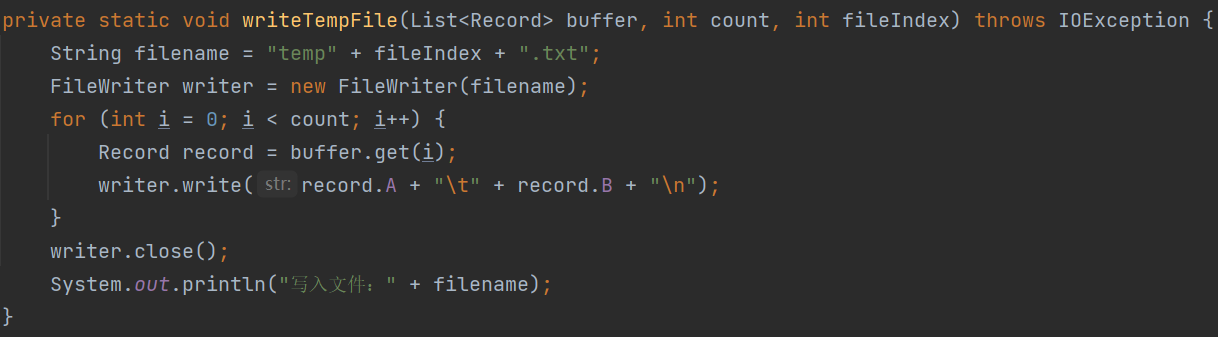
1. **主要函数的设计过程**

下面逐个函数解释它的设计过程：

1. **readNextRecord(BufferedReader reader, Record record)**

该函数的作用是从给定的 BufferedReader 中读取一行记录，并将其解析成 Record 类型的对象。函数返回 true 表示成功读取记录，否则表示文件为空。  


1. **writeTempFile(List<Record> buffer, int count, int fileIndex)**

将缓冲区中的记录写入一个临时文件中，临时文件的名称是 temp+i.txt，i 取值范围从 0 到 k-1（k 是拆分得到的临时文件数量）。其中，i 表示第 i 个临时文件。  


1. **multiwayMergeSort(String filename, int memSize, int colIndex, int chunkSize)**

这是最主要的函数，它包含了整个多路归并排序的实现过程。具体包括以下几步（具体代码比较长，就不截图放了）：

①从 filename 所指定的文件中逐个读取记录，并将其存储在大小为 chunkSize 的缓冲区（buffer）中。

②对缓冲区中的记录按照属性 A 进行排序，然后将排序后的结果写入到一个临时文件中。

③重复第①和第②步, 直到全部记录都被读取和处理过。

④读取所有临时文件中的第一条记录，并将其存储在优先队列（winnerTree）中。

⑤从优先队列中找出最小的记录，将其输出到输出文件中，然后从该记录所在的临时文件中读取下一条记录，并将其存储在优先队列中。

⑥重复第 ⑤ 步，直到所有记录都被输出到输出文件为止。

⑦删除所有临时文件。

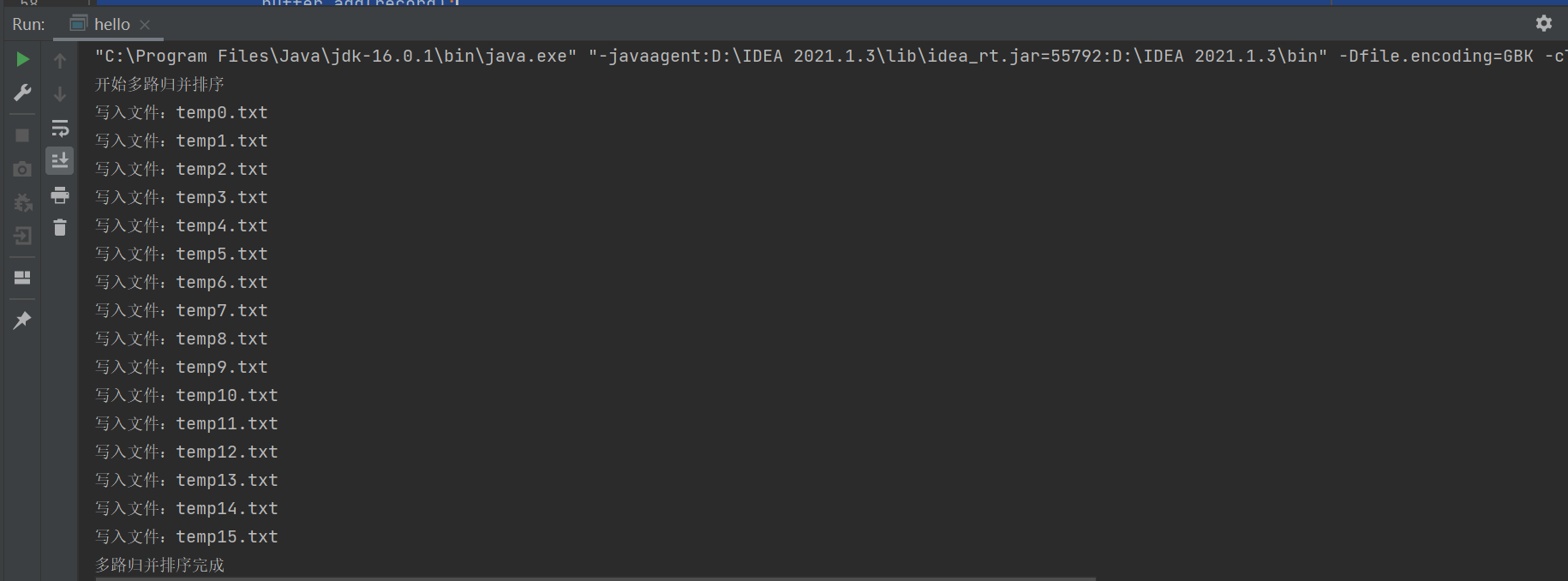
**4.main(String[] args)**

这是程序的主函数，它用于调用 multiwayMergeSort 函数来对指定的大文件进行多路归并排序。具体包括以下几步：

①定义了需要排序的文件名（filename）、内存大小（memSize）、排序属性的索引（colIndex）以及每个缓冲区的大小（chunkSize）；

②调用 multiwayMergeSort 函数来对文件进行排序。

1. **程序运行结果**

运行程序，控制台输出如下，说过程运行正常：  
  
查看输出的output.txt文件,发现所有数据已经被正确排序:  
