**2022年春（2020级软卓）**

**《软件体系结构》**

**实验二：比较不同的软件架构风格**

**姓名 赵庆举**

**学号 202024100738**

**班级 软卓1班**

**郑州大学计算机与人工智能学院**

**2022年 4月 13日**

报告内容

调试分析KWIC，理解不同架构风格

1、画出每一种风格对应程序的静态结构图（如类图）。

2、打印出程序的执行过程。

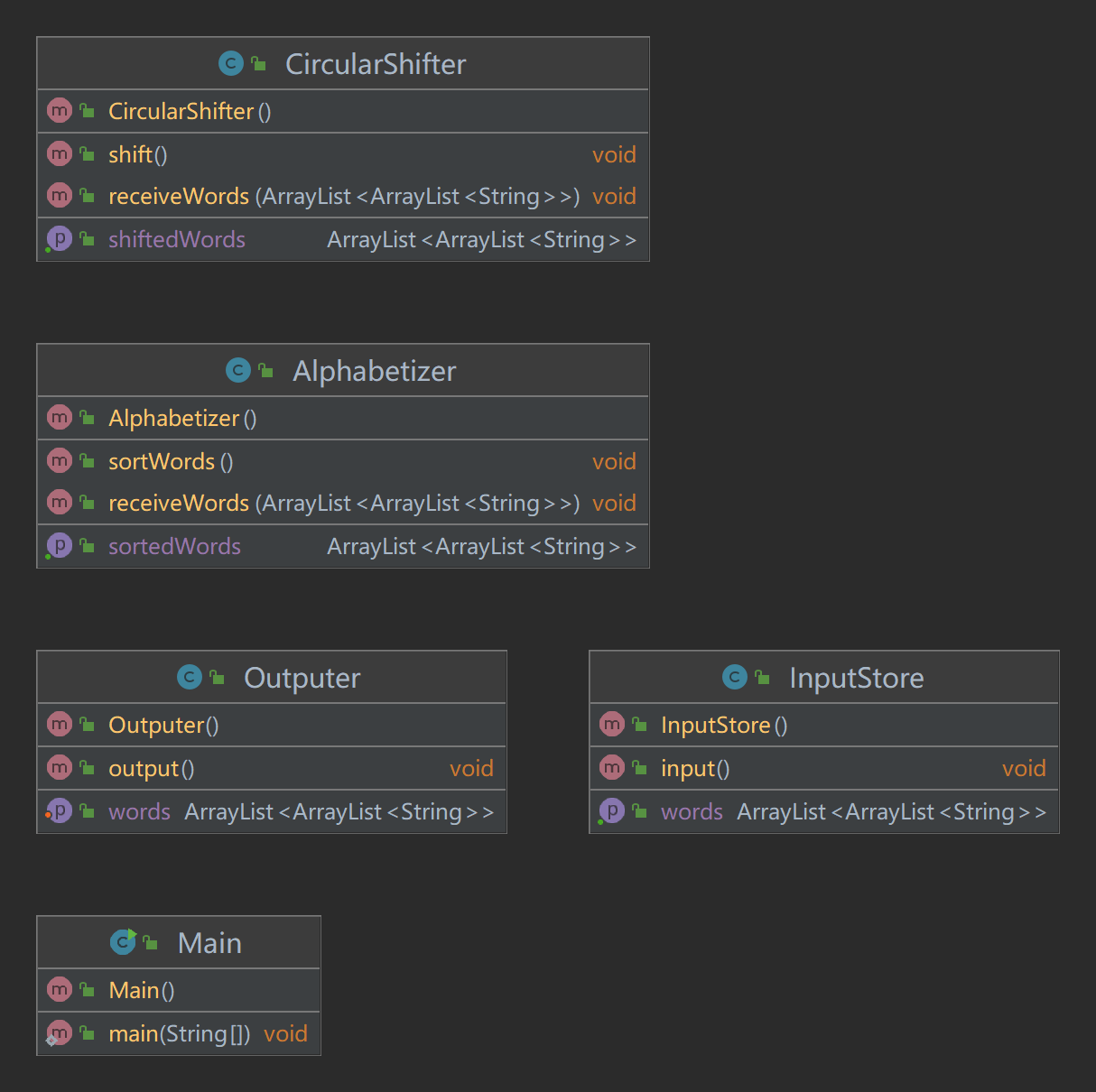
3、使用一个较大的输入文本，测试每一种风格程序的性能。

4、结合程序，总结每一种架构风格的特点。

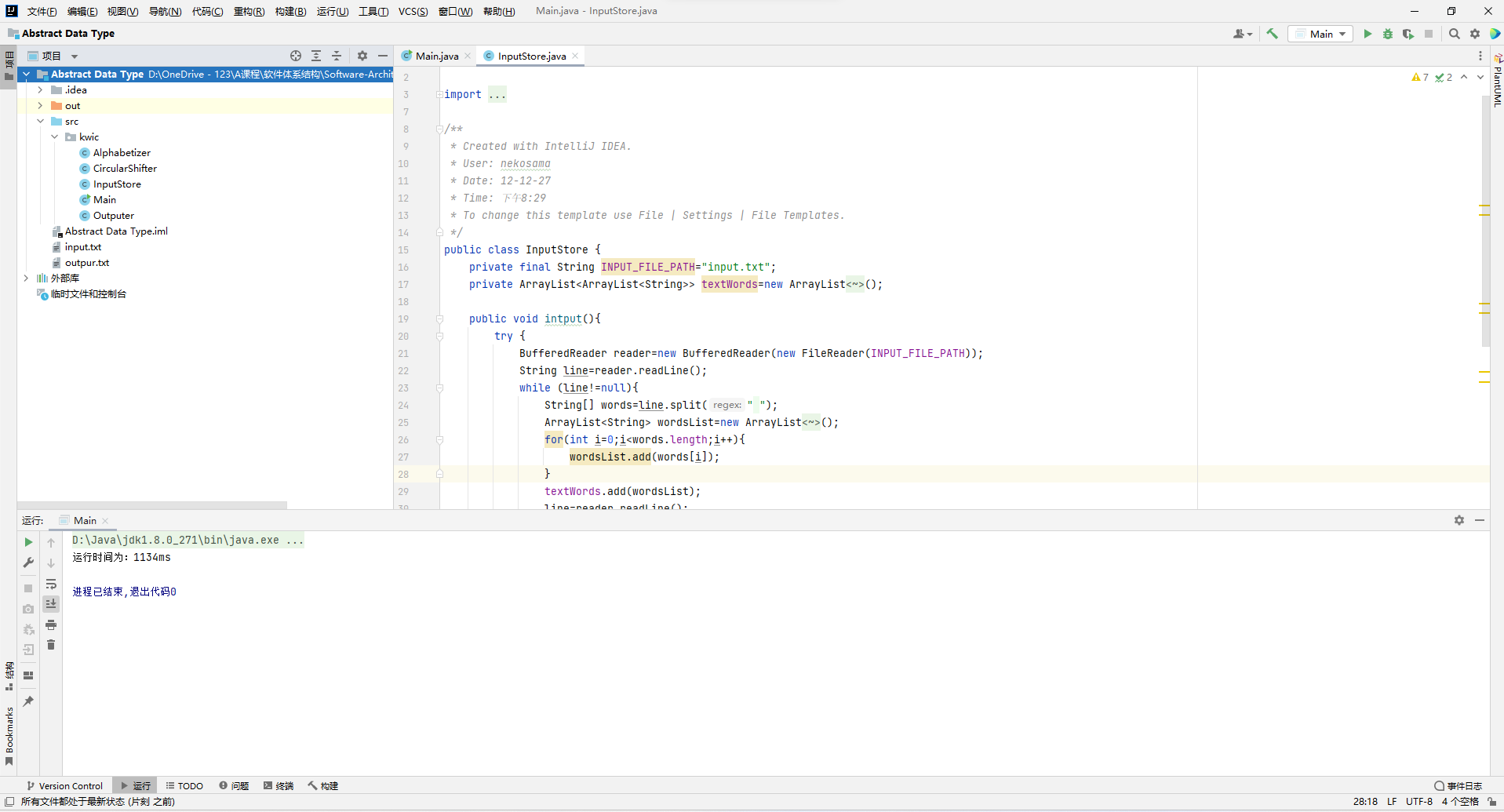
第1、2、3部分

测试数据：1459行

**Abstract Data Type**



1：Inputstore类读取目标文件，将数据存放在textwords属性中；

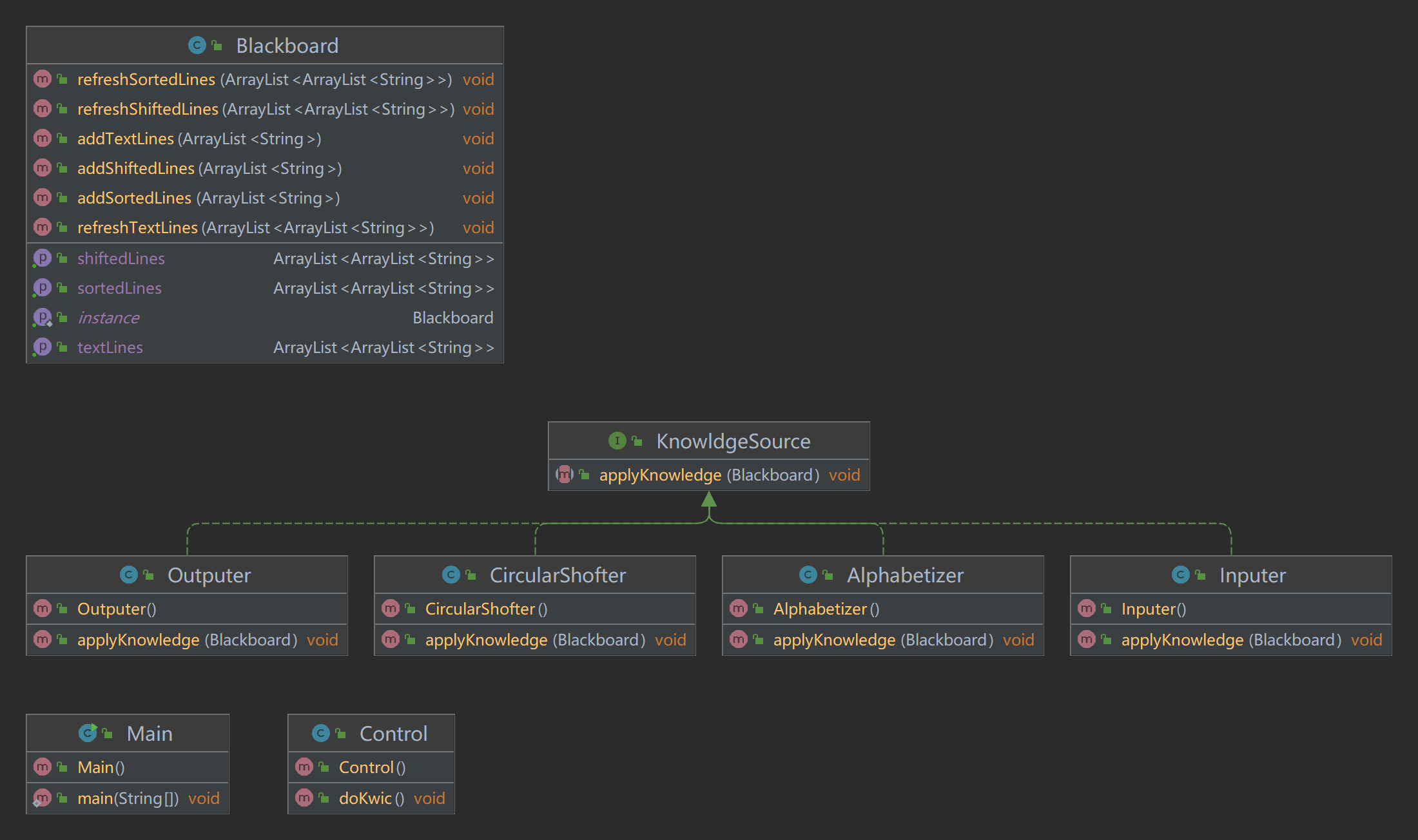


2：CircularShifter类，调用Inputstore类中textwords属性的get方法获取其中的值，执行shitf循环移位方法，结果存在shiftedwords属性中；

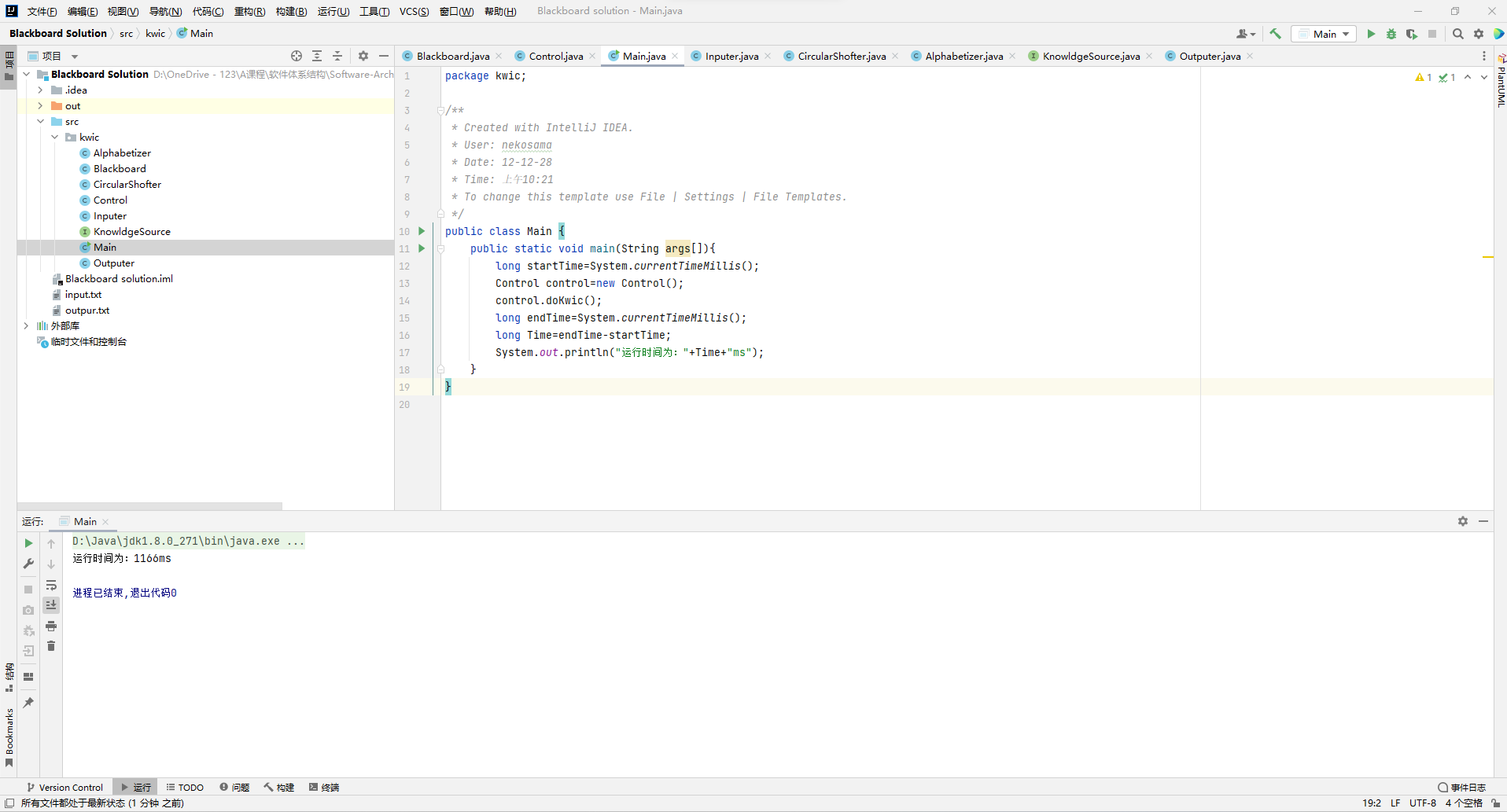
3：Alphavetizer类，通过receriveWords方法接受circularShifter中shiftedwords属性的get方法的值并作为自己的参数执行sortwords方法进行排序，结果存入words属性中。

4：Outputer类，通过setWords方法接收alphabetizer中words属性的get方法的值走位自己的属性，结果输出到目标文件中。

**Blackboard solution：**



1：创建Blackboard对象

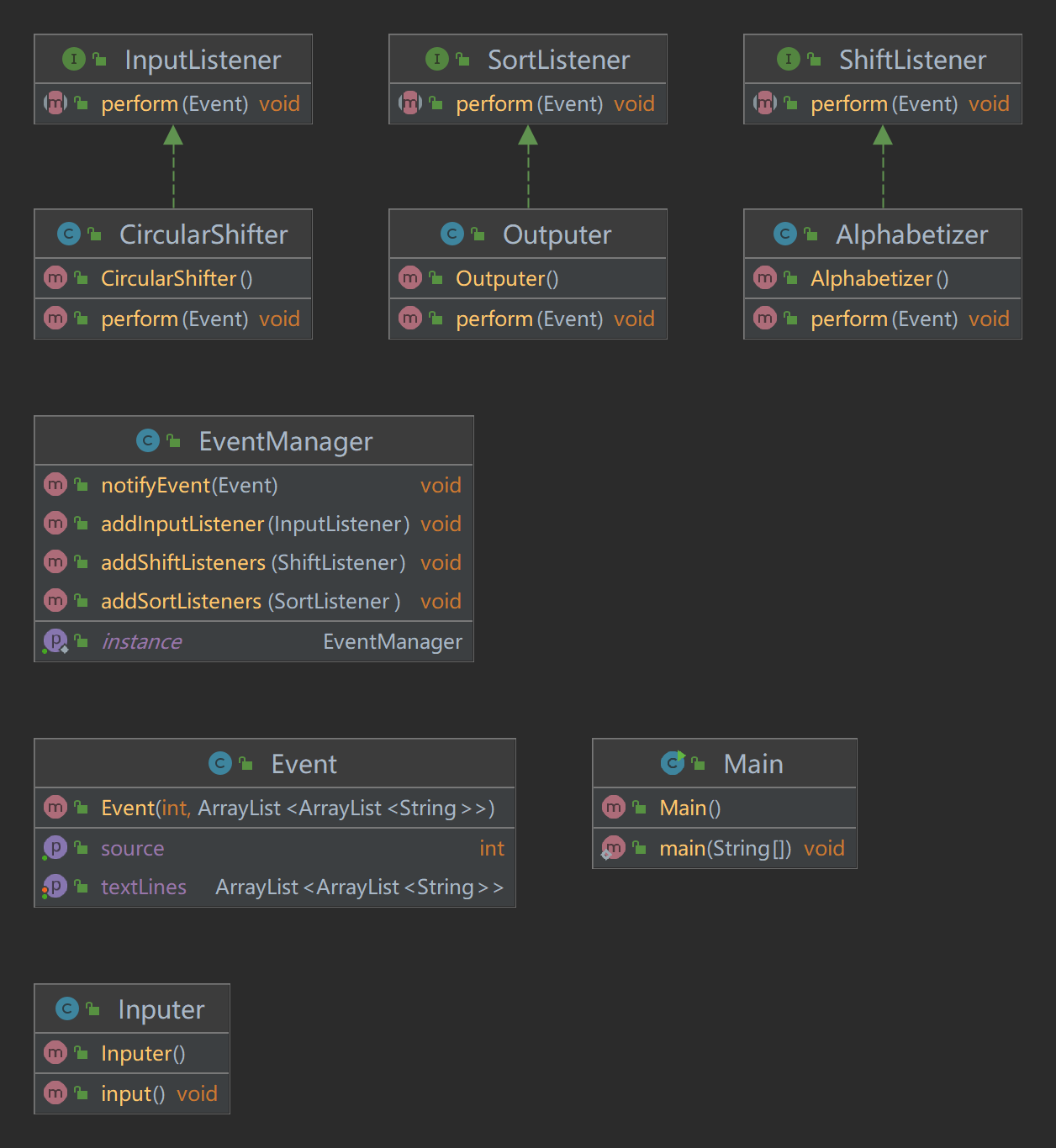
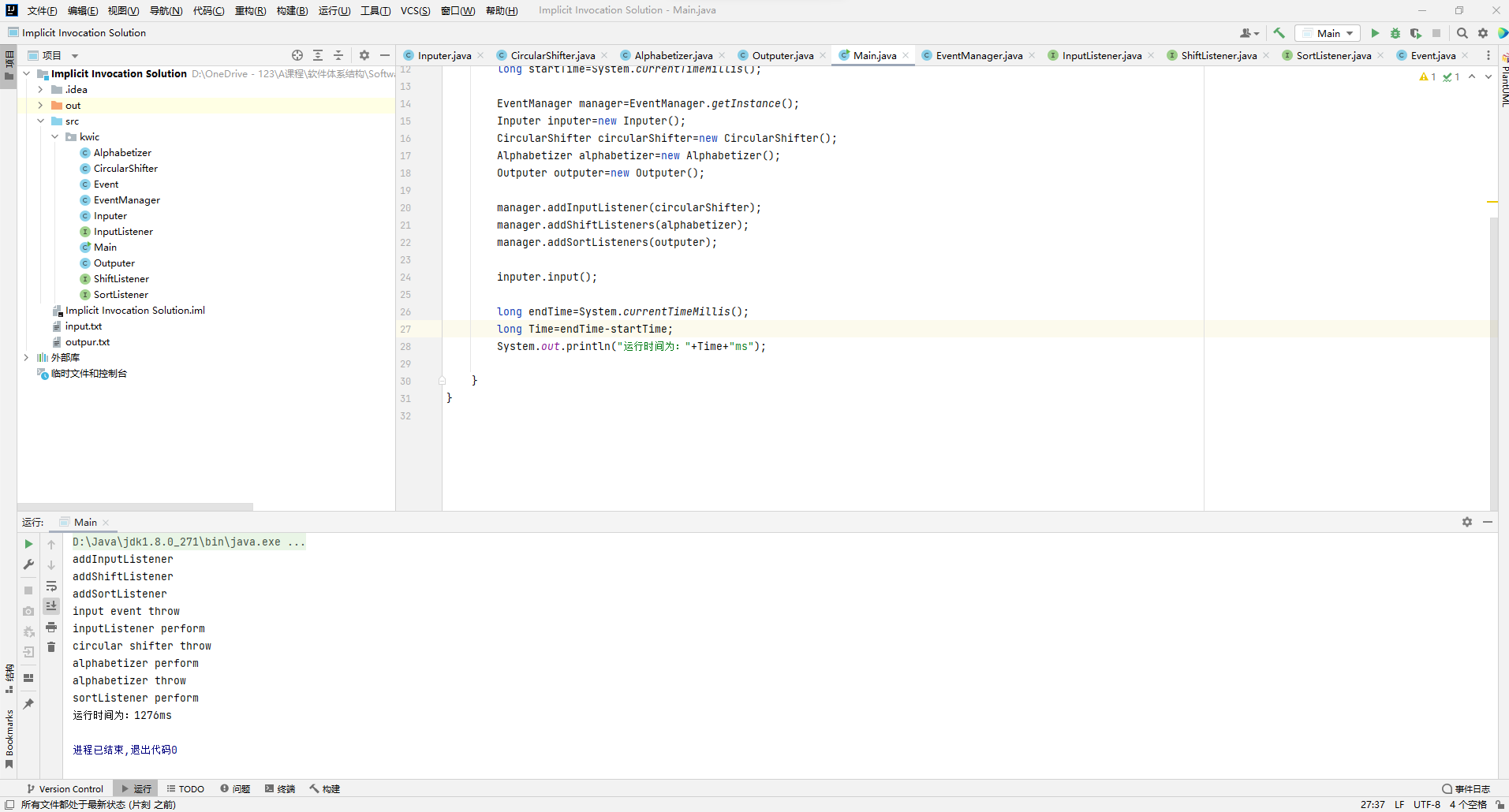


2：依次创建Inputer、CircularShofter、Alphabetizer、Outputer四种类的对象，四者主要功能以Blackboard类对象为参数。

3：按照需求调用这四个类的对象，以更改Blackboard对象的属性值。

4：依次完成读取文件、循环位移、排序和输出目的文件。

**Implicit Invocation Solution**



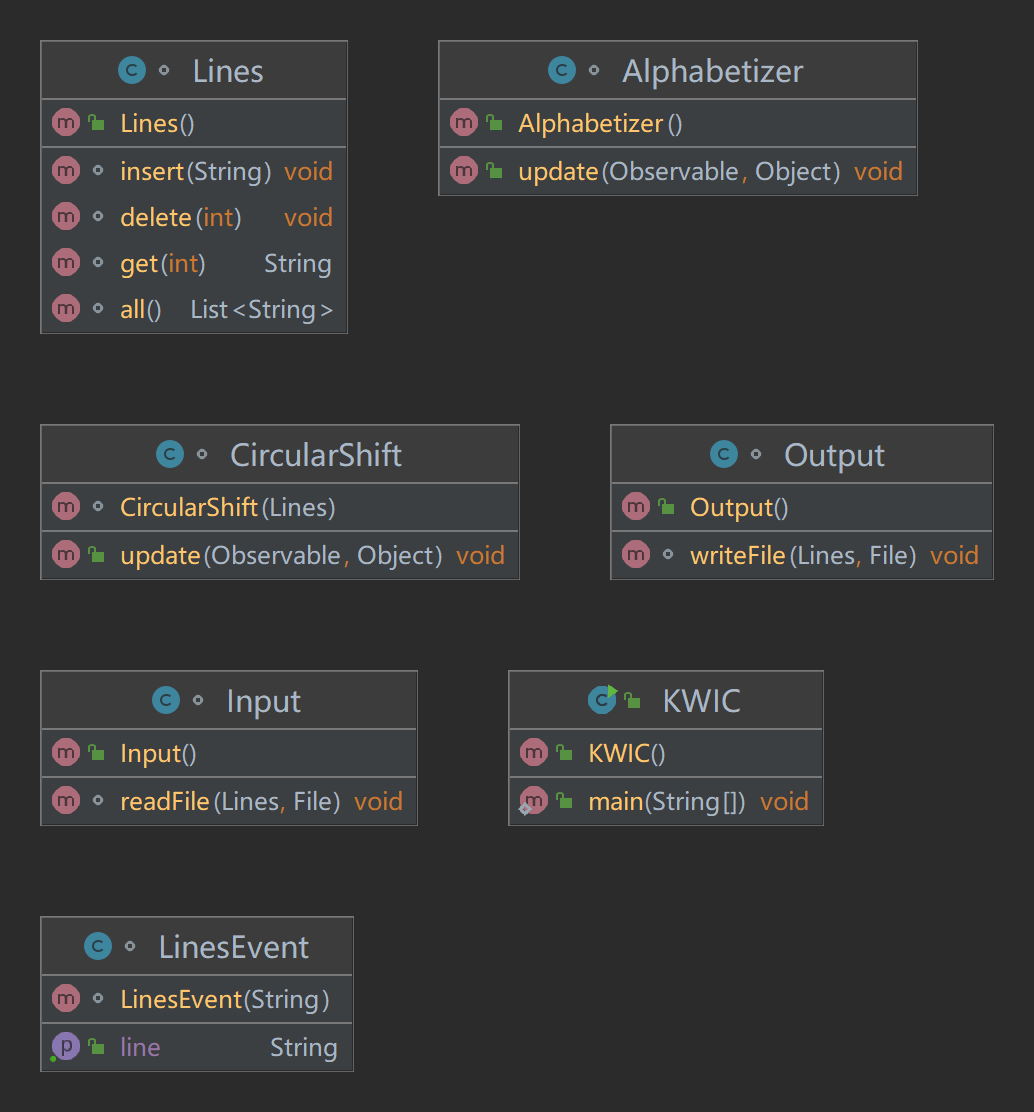
1. 创建一个EventManager的事件类对象

2. 创建分别能实现对应接口方法的类的对象（inputer，circularShifter，alphabetizer，outputer）

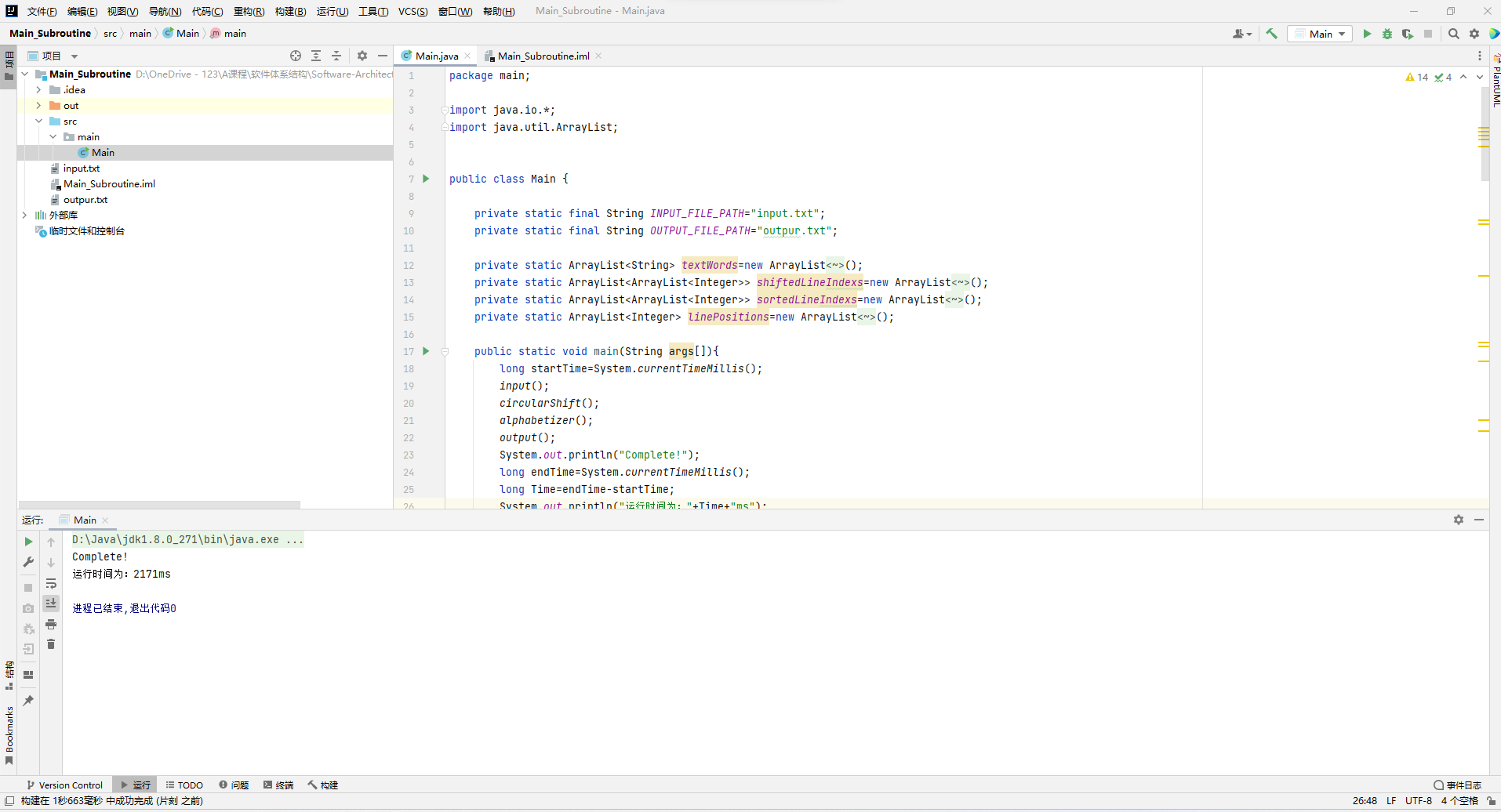
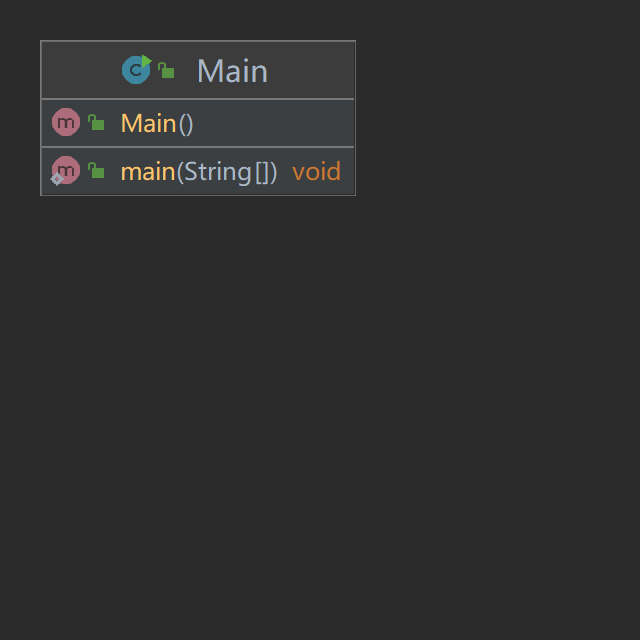
3. 将步骤二中利用构造器创建的对象的构造结果传入作为EventManager类方法的参数，这样可以实现对其的属性的改变并达到完成下一方法的条件

4. EventManager类的对象依次执行对应读取文件、循环移位、排序、输出到目的文件方法，完成功能

**Implicit Invocation Solution2**



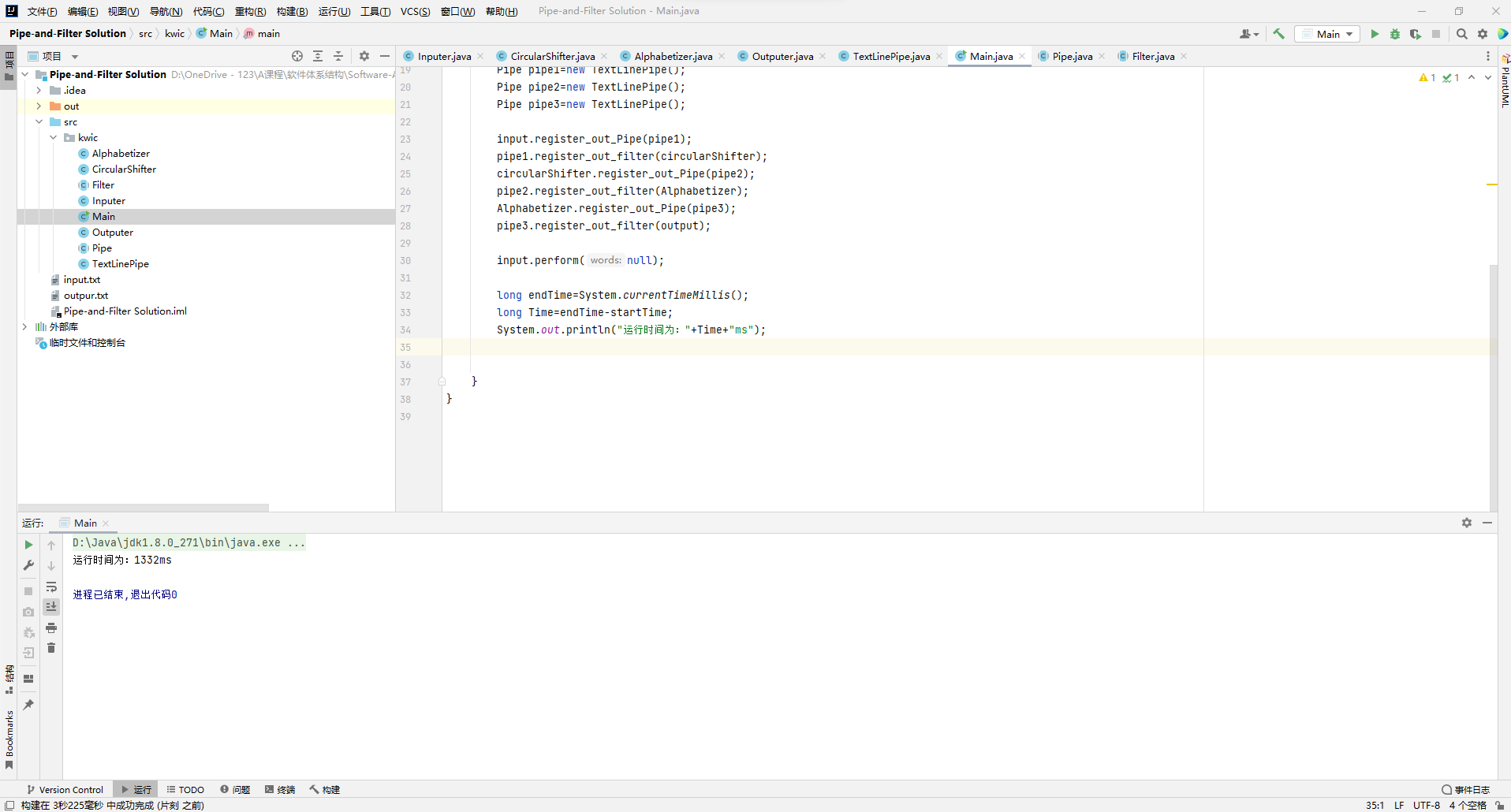
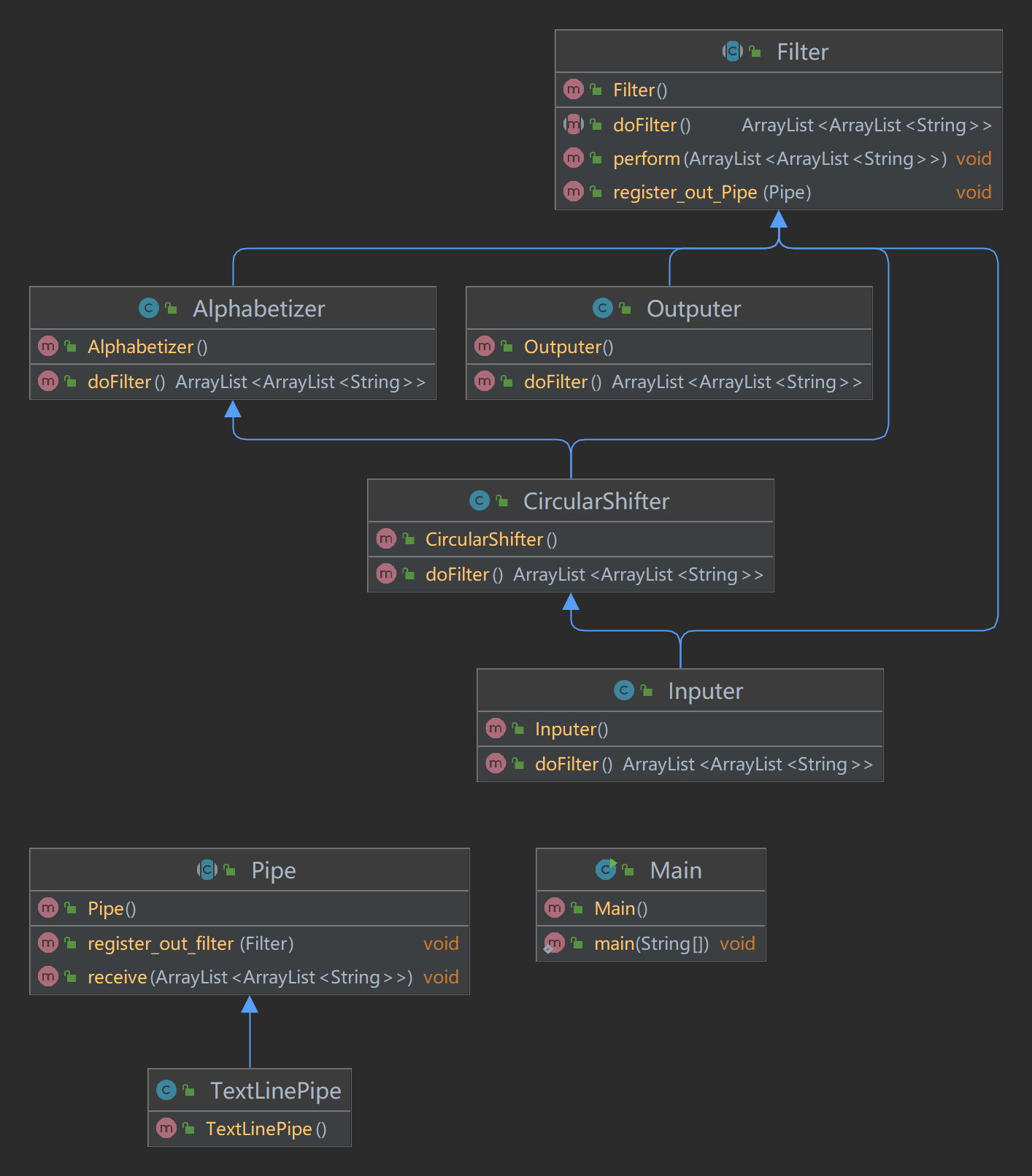
**Main Subroutine**



1：在main方法中依次执行input（从指定路径读取文件放入textwords属性）、circularShift（循环移位放入shiftedLineIndexs属性）、alphabetizer（排序放入sortedLineIndexs属性）、output（输出到目的文件）四个方法。

2：NULL

**Pipe-and-Filter Solution**



1：在启动类中通过Filter工厂类创建input、circularShifter，Alphabetizer，output四个字类对象；

2：在启动类里通过Pipe工厂pipe1，pipe2，pipe3三个TextLinePipe子类对象；

3：通过Filter工厂类对象依次调用内在方法并把Pipe工厂类对象作为参数传递过去；

4：依次执行方法，完成四类操作。

第四部分

**Abstract Data Type**

将功能封装到不同的类中，创建类的对象来执行其内部方法。用get方法将执行获得结果传给外界，利用主方法输出结果文件。

**Blackboard solution**

创建一个总的对象类，里面有每个功能类需要的属性，每个功能类都要实现可以接收总的对象类的对象的接口。首先创建总对象类的一个空对象，然后创建其余功能类的对象，把该总对象类的对象传入作为参数依次执行各个功能类的方法，上一个方法执行后对该对象的改变恰好可以使其满足执行下一个功能类的方法。总结就是：所有的类对一个对象进行操作

**Implicit Invocation Solution**

每个执行的类都对应一个接口，对应的类来实现对应的接口中的方法，在执行类中通过选择语句选择需要执行的方法，实现需要的功能。

**Main Subroutine**

把所有方法和属性都写在一个大类中（启动类），依次执行相应的方法来处理得到结果，整个程序只需要一个类。

**Pipe-and-Filter Solution**

创建能完成目的几个工厂类，通过多态和继承来创建具体的操作类和属性，然后创建子类型的父类对象，进而通过其内部方法完成每个类需要完成的工作。