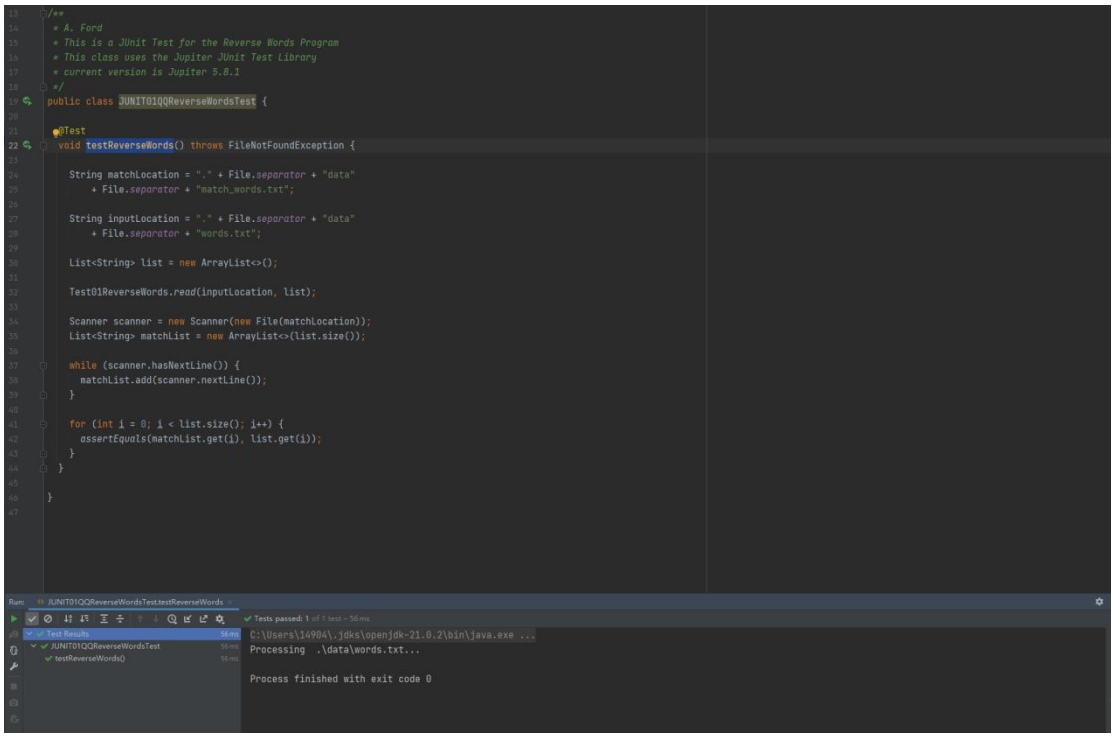
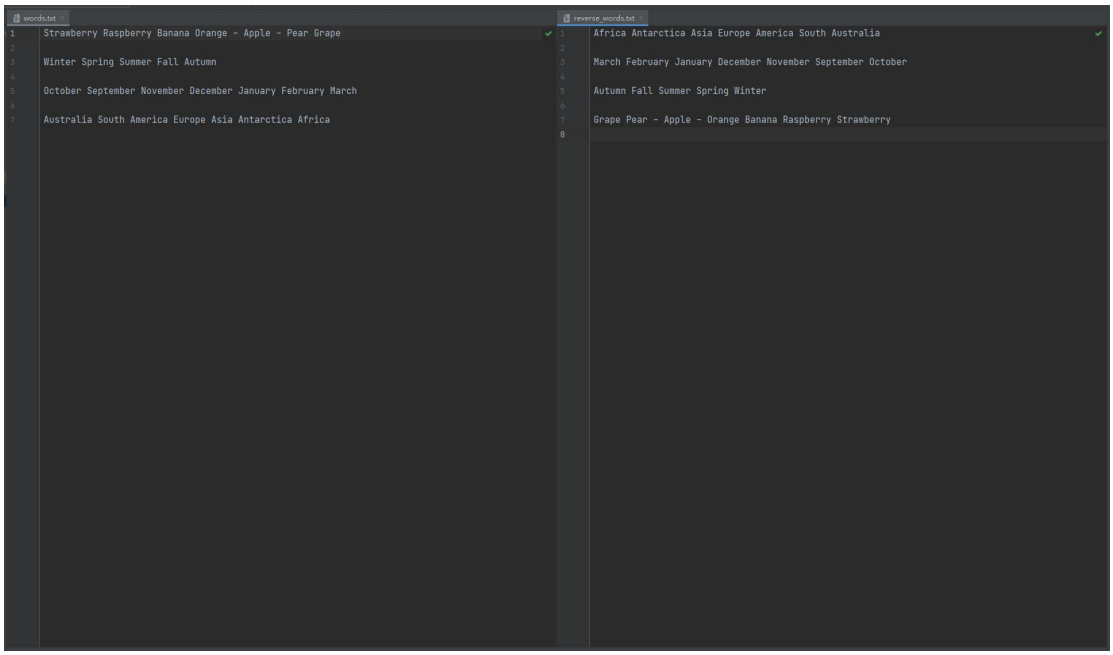
1. Run the junits.JUNIT01QQReverseWordsTest in the junits package



```java
/**
 * A. Ford
 * This is a JUnit Test for the Reverse Words Program
 * This class uses the Jupiter JUnit Test Library
 * current version is Jupiter 5.8.1
 */
public class JUNIT01QQReverseWordsTest {

  @Test
  void testReverseWords() throws FileNotFoundException {

    String matchLocation = "." + File.separator + "data"
        + File.separator + "match_words.txt";

    String inputLocation = "." + File.separator + "data"
        + File.separator + "words.txt";

    List<String> list = new ArrayList<>();

    Test01ReverseWords.read(inputLocation, list);

    Scanner scanner = new Scanner(new File(matchLocation));
    List<String> matchList = new ArrayList<>(list.size());

    while (scanner.hasNextLine()) {
      matchList.add(scanner.nextLine());
    }

    for (int i = 0; i < list.size(); i++) {
      assertEquals(matchList.get(i), list.get(i));
    }
  }

}
```

```
Run: JUNIT01QQReverseWordsTest.testReverseWords
Tests passed: 1 of 1 test – 56 ms
Test Results 56 ms
JUNIT01QQReverseWordsTest 56 ms
testReverseWords() 56 ms
C:\Users\14904\.jdks\openjdk-21.0.2\bin\java.exe ...
Processing  .\data\words.txt...

Process finished with exit code 0
```

2. Create a snapshot of your "words" and "reverse_words" side by side



words.txt

```
Strawberry Raspberry Banana Orange - Apple - Pear Grape

Winter Spring Summer Fall Autumn

October September November December January February March

Australia South America Europe Asia Antarctica Africa
```

reverse_words.txt

```
Africa Antarctica Asia Europe America South Australia

March February January December November September October

Autumn Fall Summer Spring Winter

Grape Pear - Apple - Orange Banana Raspberry Strawberry

```

3. Summary

3.1 reads in an empty list and file location and populates list with the lines and words reversed.

3.1.1 Create a file object based on the file path;
3.1.2 Check whether the file object exists. If no, an error message is displayed and the program is terminated.
3.1.3 Use the Scanner class to scan file objects and use the while loop to read the contents of files;
3.1.4 First, the original string is separated by commas to obtain an array of strings. Iterate through the array and always insert each item into the starting position of the StringBuilder object. When the loop is over, the inverted string can be obtained;
3.1.5 Insert the inversion string to the start bit of the list;
3.1.6 Clearing the StringBuilder object;

3.2 writes to an output file

3.2.1 Open a FileOutputStream file stream and generate a PrintStream object;
3.2.2 Write the contents to the file in a loop;
3.2.3 Close the file stream;

3.3 main method

3.3.1 Declaring the source file address;
3.3.2 Declaring the destination folder address;
3.3.3 Declare the destination file address;
3.3.4 Initializing an empty list;
3.3.5 The contents of the file are read from the target file and placed in the inversion list;
3.3.6 Check whether the target address exists. If the target address does not exist, an error message is displayed.
3.3.7 Writing the content to the destination file address;

3.4 CodeStyle

3.4.1 The indentation of the code should be 4 characters;
3.4.2 JavaDoc for every class and method;
3.4.3 When using the for loop, try to use the enhanced for loop;
3.4.4 There should be Spaces on both sides of the operator;
3.4.5 Import needs to be imported in sequence.
3.4.6 Classes, methods, variables should follow the hump naming method, in line with the specification;