Jiao Gong (001561450)

Dou Jin (001098640)

**Program Structures & Algorithms**

**Fall 2021**

**Final Project**

* **Task (List down the tasks performed in the Assignment)**

1. Your task is to implement MSD radix sort for a natural language which uses Unicode characters.
2. You may choose your own language or (Simplified) Chinese.
3. Additionally, you will complete a literature survey of relevant papers and you will compare your method with Timsort, Dual-pivot Quicksort, Huskysort, and LSD radix sort.

* **Solution**

Since we already improve the MSD radix sort to allow it to sort English strings, we need to find a way to convert Chinese strings to Pinyin, and then using MSD radix string sort to get the conventional order for Chinese. The flow of above algorithm is shown in Figure 1.

Figure1

Work flow for the solution

Figure1

We invoked a utility package named pinyin4j to help us convert Chinese strings to Pinyin strings. Then with the MSDCoderFactory class, we can easily transfer Pinyin to integer arrays. After implementing different sort algorithms like Timsort, Dual-pivot Quick sort, and Husky sort through the Coder Factory. We can compare the speed for sorting Chinese strings among different types of sorting algorithm.

* **Relationship Conclusion:**

1. The performance from the fastest to the slowest when sort the Chinese strings using different sorting algorithms is as following:

Husky Mergesort > MSD Radix Sort > MSD Exchange Sort >= Timsort > LSD Radix Sort > Dual-pivot Quicksort

1. All the sorting algorithms have a linear relationship between the number of words and time.
2. A cutoff does helps to improve the speed for MSD radix sort when sort Chinese strings. Before we add the function to convert to Insertionsort when the size of array is small, the speed for MSD radix sort could be 3 times from now.
3. Due to the characteristics of Chinese pinyin, we need to convert the pinyin of each word of each word into equal length before LSD, MSD, MSD Exchange sort.
4. Husky Merge Sort has the best performance when sort large number of elements. But we do recommend using our MSDChineseSort which based on the MSD Radix sort algorithm to sort Chinese strings. It is easier to be implemented than the Husky Mergesort.

* **Evidence to support the conclusion:**

1. **Output (Snapshot of Code output in the terminal)** Text

   Description automatically generated**Text

   Description automatically generated**

A picture containing text, plaque, scoreboard

Description automatically generated

1. **Graphical Representation (Observations from experiments should be tabulated and analyzed by plotting graphs (usually in excel) to arrive on the relationship conclusion)**

**Chart, scatter chart

Description automatically generated**

Figure2

* **Unit tests result:(Snapshot of successful unit test run)**

Text

Description automatically generated with low confidence

MSD Chinese Radix Sort Test

A screenshot of a computer

Description automatically generated with medium confidence

MSD Exchange Radix sort Test

Text

Description automatically generated

LSD Chinese Radix Sort Test

Text

Description automatically generated

Husky Merge sort Test

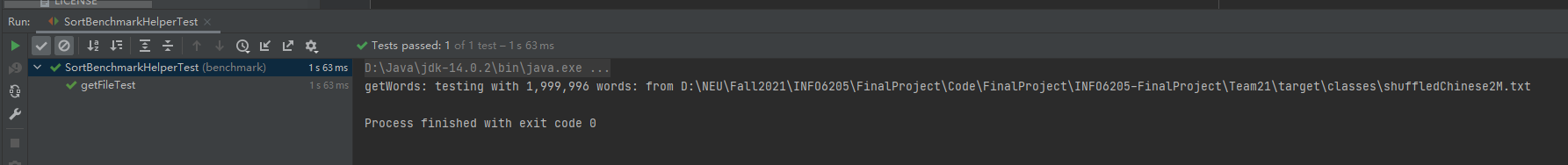
Graphical user interface

Description automatically generated with medium confidence

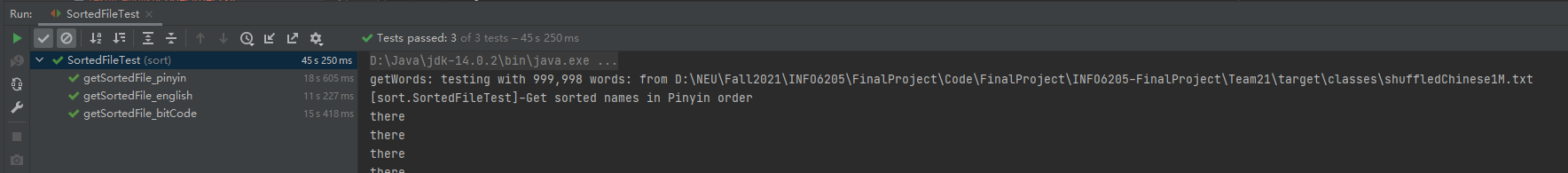
Dual-pivot Quick sort Test

Text

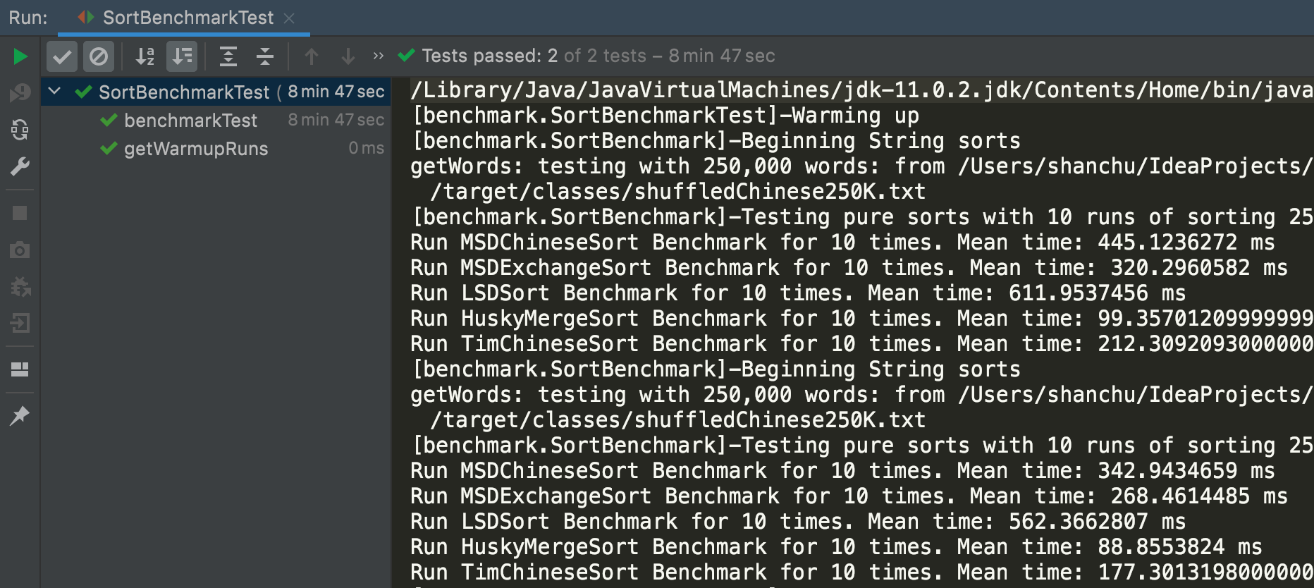
Description automatically generated Timsort Test

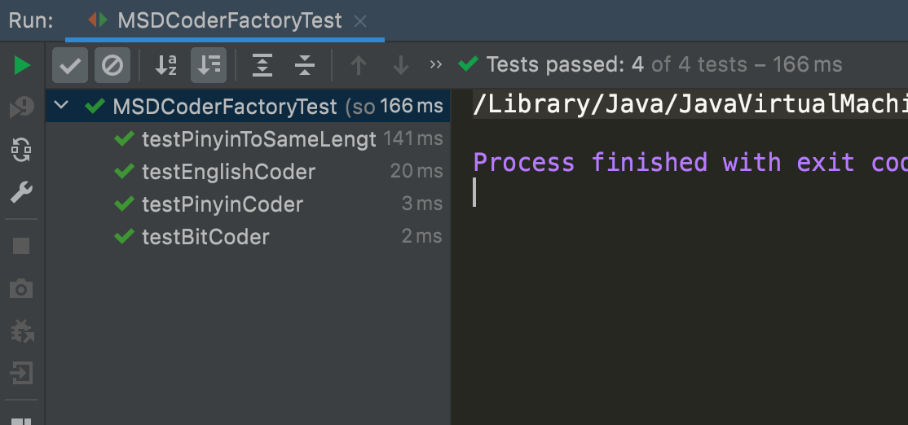


Sort Benchmark Helper Test



Get sorted file via different encoding types Test

Sort Benchmark Test



MSDCoderFactory Test