Program Structures and Algorithms
Fall 2022(SEC 06)

NAME: Mahathi Siddavatam
NUID: 002198134

Task:

Please see the presentation on *Assignment on Parallel Sorting* under the *Exams. etc.* module.

Your task is to implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. You will consider two different schemes for deciding whether to sort in parallel.

1. **A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running. It's your job to experiment and come up with a good value for this cutoff. If there are fewer elements to sort than the cutoff, then you should use the system sort instead.**
2. **Recursion depth or the number of available threads. Using this determination, you might decide on an ideal number (*t*) of separate threads (stick to powers of 2) and arrange for that number of partitions to be parallelized (by preventing recursion after the depth of *lg t* is reached).**
3. **An appropriate combination of these.**

There is a *Main* class and the *ParSort* class in the *sort.par* package of the INFO6205 repository. The *Main* class can be used as is but the *ParSort* class needs to be implemented where you see "TODO..." [it turns out that these TODOs are already implemented].

Unless you have a good reason not to, you should just go along with the Java8-style future implementations provided for you in the class repository.

You must prepare a report that shows the results of your experiments and draws a conclusion (or more) about the efficacy of this method of parallelizing sort. Your experiments should involve sorting arrays of sufficient size for the parallel sort to make a difference. You should run with many different array sizes (they must be sufficiently large to make parallel sorting worthwhile, obviously) and different cutoff schemes.

For varying the number of threads available, you might want to consult the following resources:

- **https://www.callicoder.com/java-8-completablefuture-tutorial/#a-note-about-executor-and-thread-pool**
- **Links to an external site.**

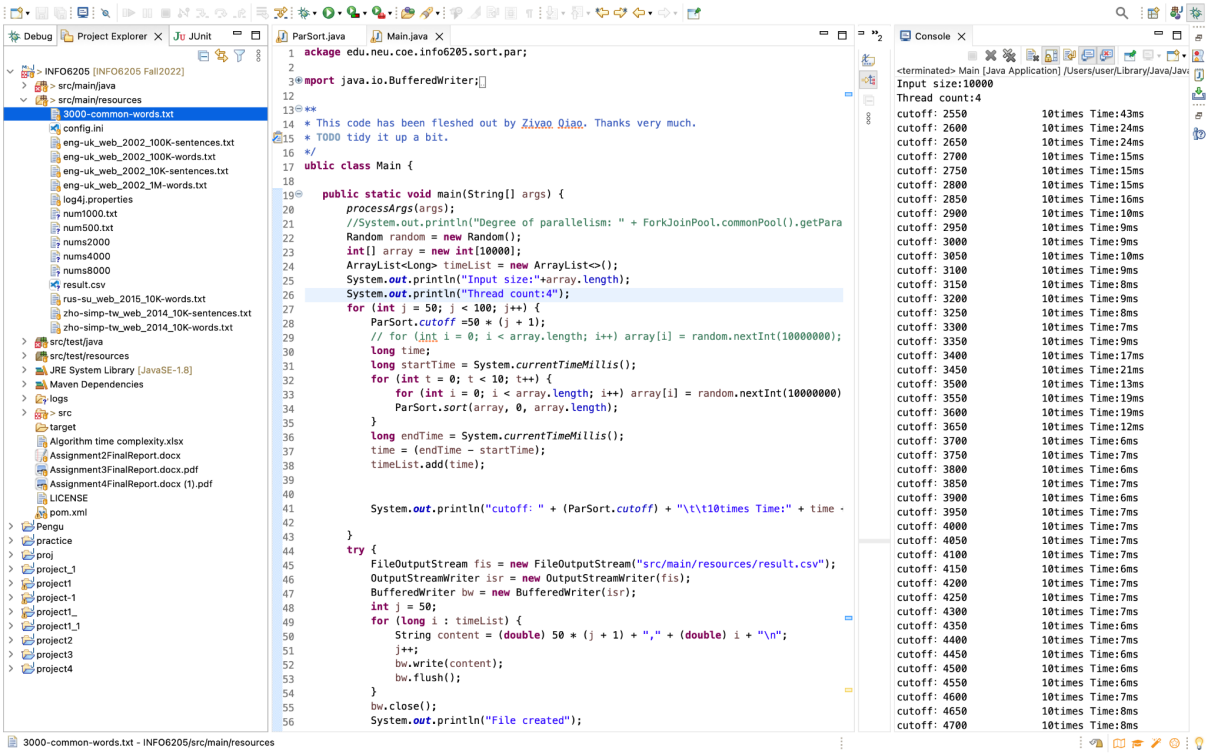
-
- https://stackoverflow.com/questions/36569775/how-to-set-forkjoinpool-with-the-desired-number-of-worker-threads-in-completable
- Links to an external site.
-

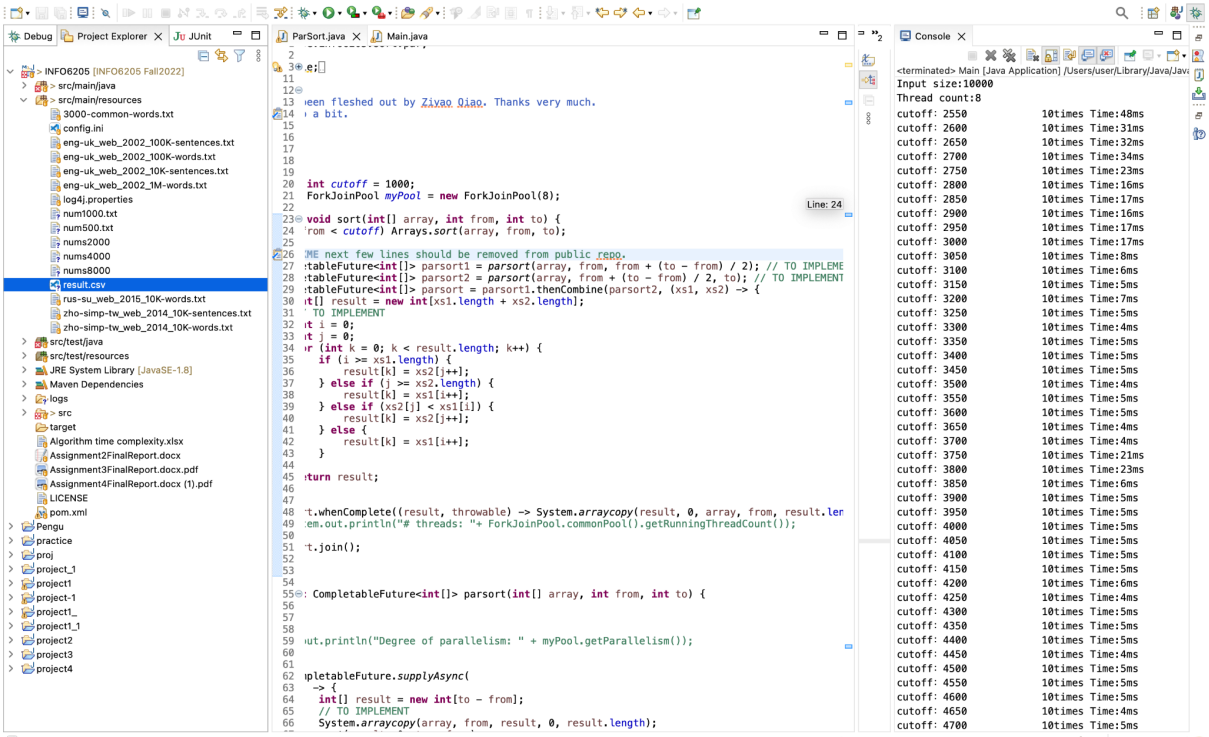
Good luck and enjoy.

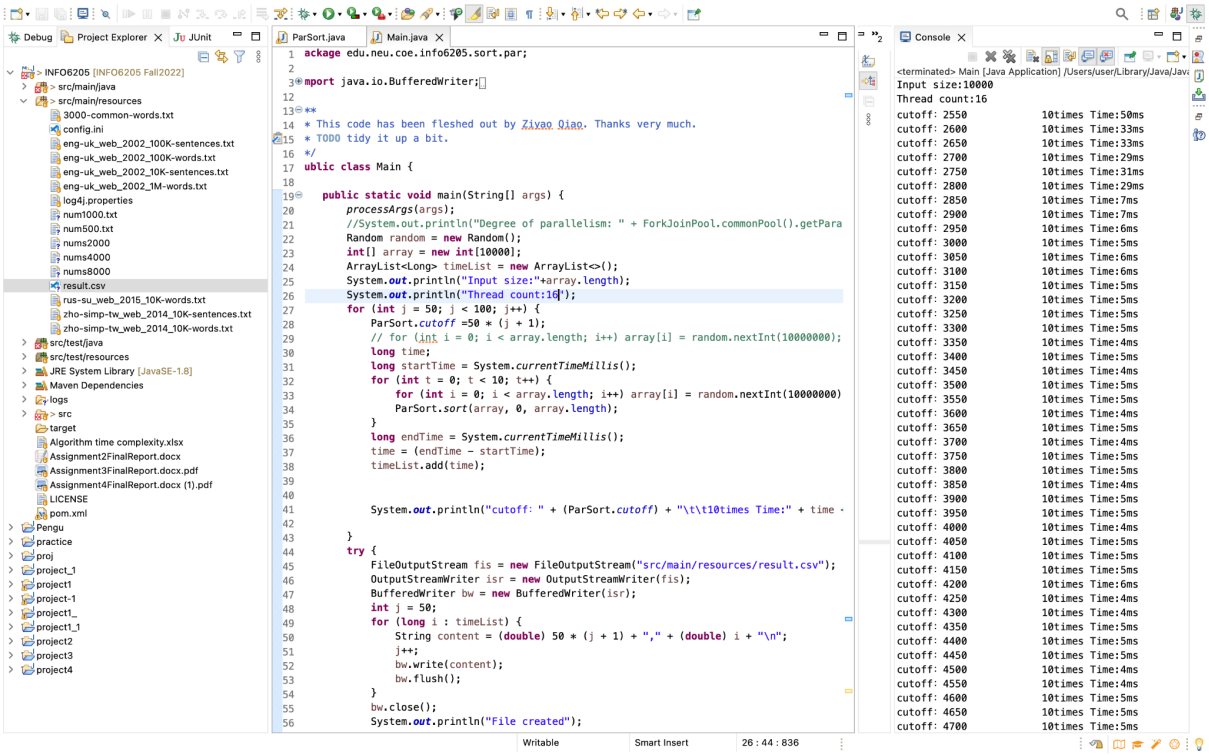
Relationship Conclusion: The conclusion regarding the efficiency of parallel sorting is that it is better to have one thread doing multiple tasks at a time than have multiple threads synchronizing and doing multiple tasks. This is evident in the graphs that suggest in most cases that, the more the threads the more time taken to complete the task.

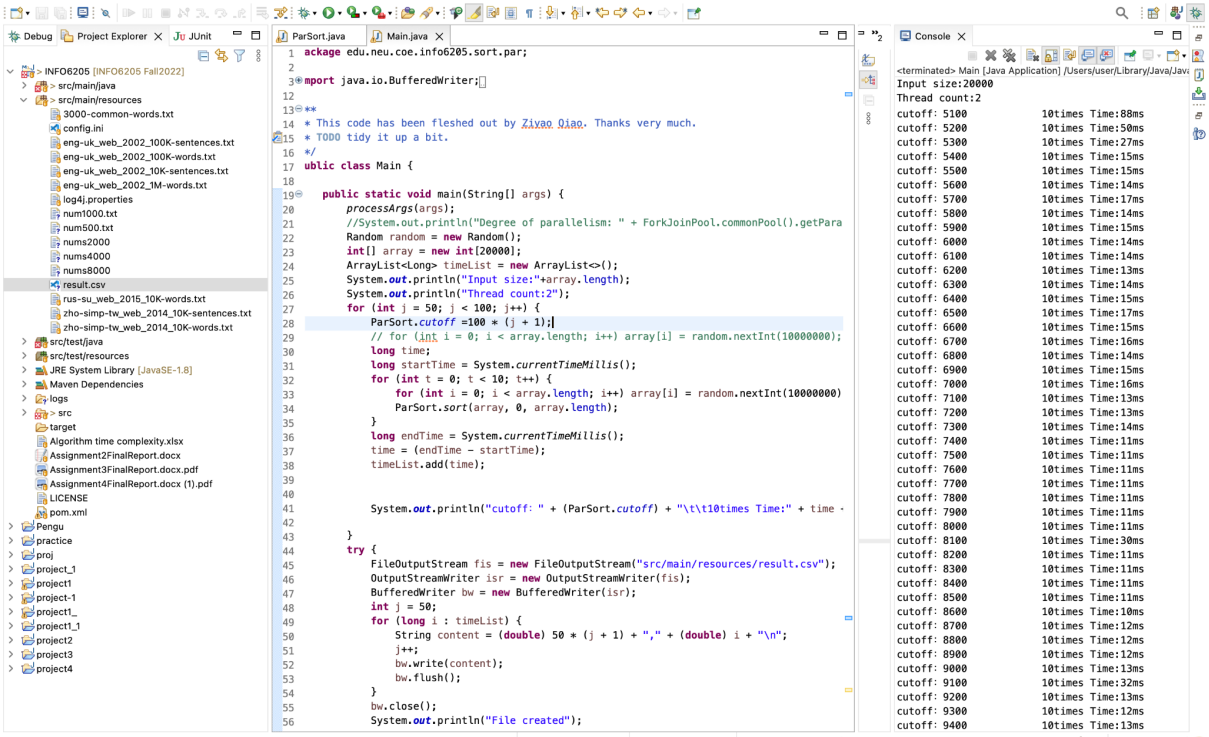
The cut-off value is most optimal when it is 25%-30% of the input size. N/8 is also an efficient cut-off value for many cases.

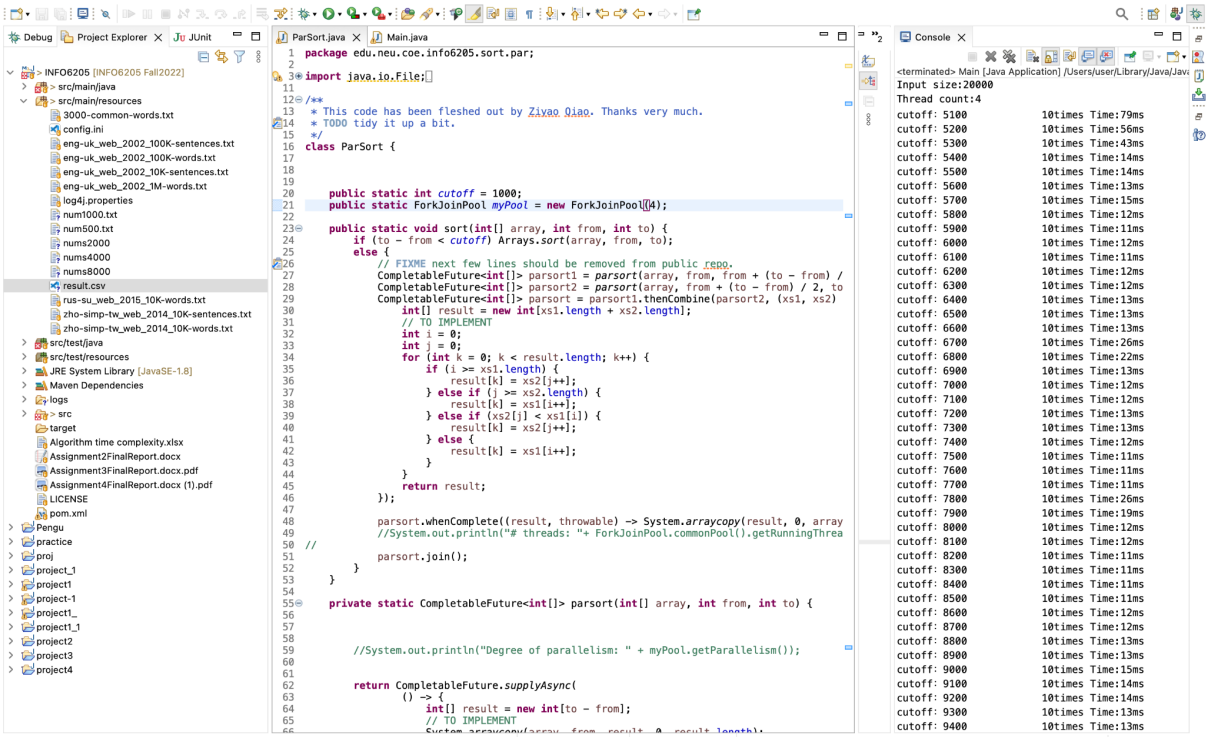
Evidence to support that conclusion:

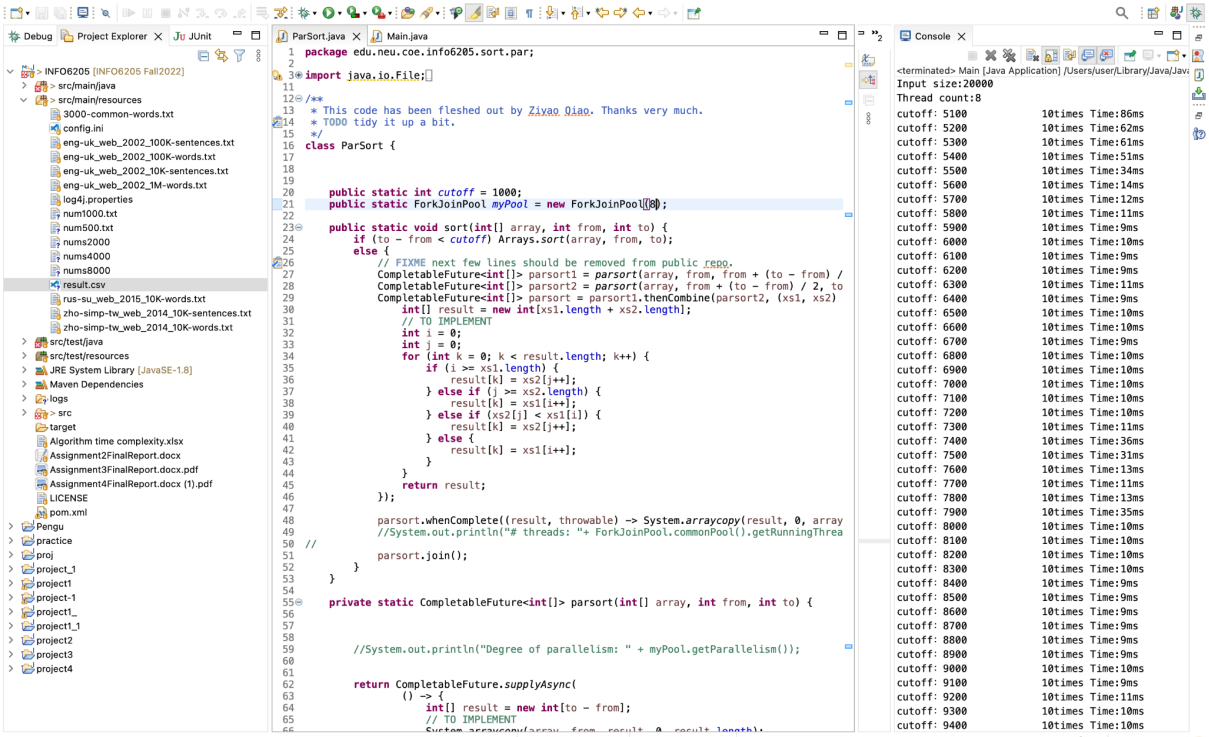


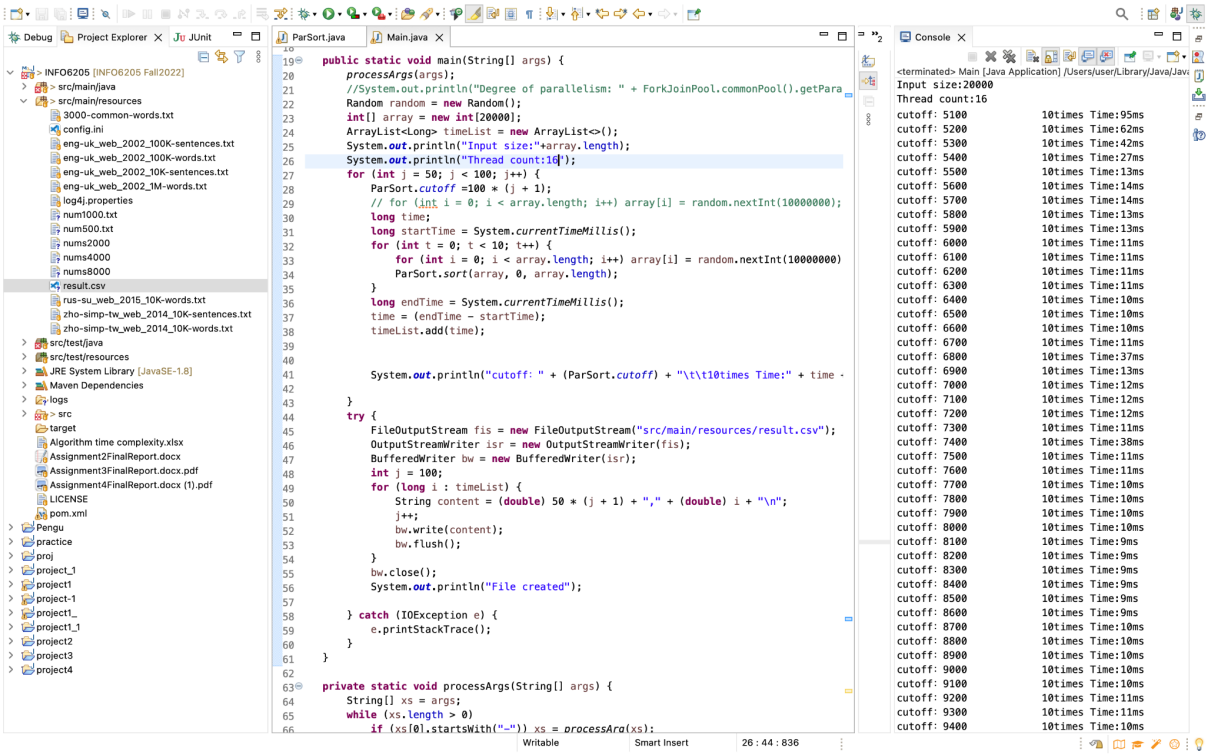

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
Line: 24
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:10000
Thread count:8
cutoff: 2550 10times Time:48ms
cutoff: 2600 10times Time:31ms
cutoff: 2650 10times Time:32ms
cutoff: 2700 10times Time:34ms
cutoff: 2750 10times Time:23ms
cutoff: 2800 10times Time:16ms
cutoff: 2850 10times Time:17ms
cutoff: 2900 10times Time:16ms
cutoff: 2950 10times Time:17ms
cutoff: 3000 10times Time:17ms
cutoff: 3050 10times Time:8ms
cutoff: 3100 10times Time:6ms
cutoff: 3150 10times Time:5ms
cutoff: 3200 10times Time:7ms
cutoff: 3250 10times Time:5ms
cutoff: 3300 10times Time:4ms
cutoff: 3350 10times Time:5ms
cutoff: 3400 10times Time:5ms
cutoff: 3450 10times Time:5ms
cutoff: 3500 10times Time:4ms
cutoff: 3550 10times Time:5ms
cutoff: 3600 10times Time:5ms
cutoff: 3650 10times Time:4ms
cutoff: 3700 10times Time:4ms
cutoff: 3750 10times Time:21ms
cutoff: 3800 10times Time:23ms
cutoff: 3850 10times Time:6ms
cutoff: 3900 10times Time:5ms
cutoff: 3950 10times Time:5ms
cutoff: 4000 10times Time:5ms
cutoff: 4050 10times Time:5ms
cutoff: 4100 10times Time:5ms
cutoff: 4150 10times Time:5ms
cutoff: 4200 10times Time:6ms
cutoff: 4250 10times Time:4ms
cutoff: 4300 10times Time:5ms
cutoff: 4350 10times Time:5ms
cutoff: 4400 10times Time:5ms
cutoff: 4450 10times Time:4ms
cutoff: 4500 10times Time:5ms
cutoff: 4550 10times Time:5ms
cutoff: 4600 10times Time:5ms
cutoff: 4650 10times Time:4ms
cutoff: 4700 10times Time:5ms
result.csv - INFO6205/src/main/resources

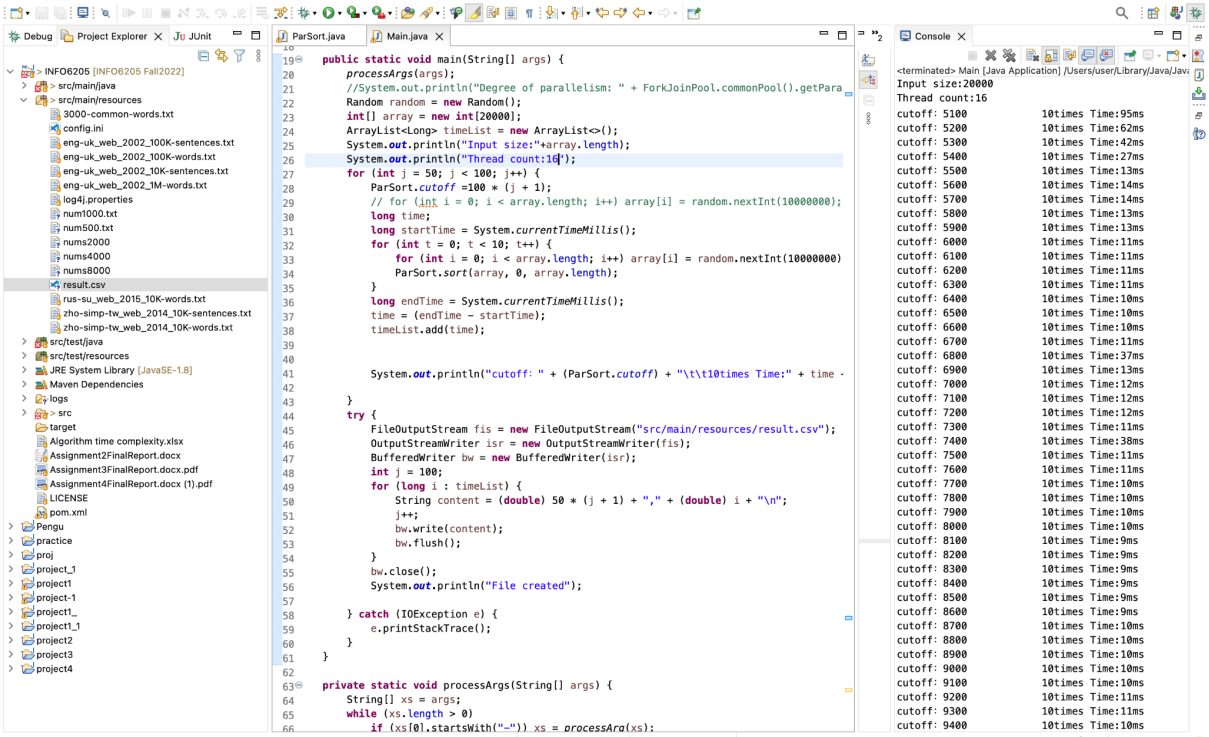

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.BufferedWriter;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
public class Main {
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getPara
Random random = new Random();
int[] array = new int[10000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:16");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =50 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000)
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 50;
for (long i : timeList) {
String content = (double) 50 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
Writable
Smart Insert
26 : 44 : 836
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:10000
Thread count:16
cutoff: 2550 10times Time:50ms
cutoff: 2600 10times Time:33ms
cutoff: 2650 10times Time:33ms
cutoff: 2700 10times Time:29ms
cutoff: 2750 10times Time:31ms
cutoff: 2800 10times Time:29ms
cutoff: 2850 10times Time:7ms
cutoff: 2900 10times Time:7ms
cutoff: 2950 10times Time:6ms
cutoff: 3000 10times Time:5ms
cutoff: 3050 10times Time:6ms
cutoff: 3100 10times Time:6ms
cutoff: 3150 10times Time:5ms
cutoff: 3200 10times Time:5ms
cutoff: 3250 10times Time:5ms
cutoff: 3300 10times Time:5ms
cutoff: 3350 10times Time:4ms
cutoff: 3400 10times Time:5ms
cutoff: 3450 10times Time:4ms
cutoff: 3500 10times Time:5ms
cutoff: 3550 10times Time:5ms
cutoff: 3600 10times Time:4ms
cutoff: 3650 10times Time:5ms
cutoff: 3700 10times Time:4ms
cutoff: 3750 10times Time:5ms
cutoff: 3800 10times Time:5ms
cutoff: 3850 10times Time:4ms
cutoff: 3900 10times Time:5ms
cutoff: 3950 10times Time:5ms
cutoff: 4000 10times Time:4ms
cutoff: 4050 10times Time:5ms
cutoff: 4100 10times Time:5ms
cutoff: 4150 10times Time:5ms
cutoff: 4200 10times Time:6ms
cutoff: 4250 10times Time:4ms
cutoff: 4300 10times Time:4ms
cutoff: 4350 10times Time:5ms
cutoff: 4400 10times Time:5ms
cutoff: 4450 10times Time:5ms
cutoff: 4500 10times Time:4ms
cutoff: 4550 10times Time:4ms
cutoff: 4600 10times Time:5ms
cutoff: 4650 10times Time:5ms
cutoff: 4700 10times Time:5ms

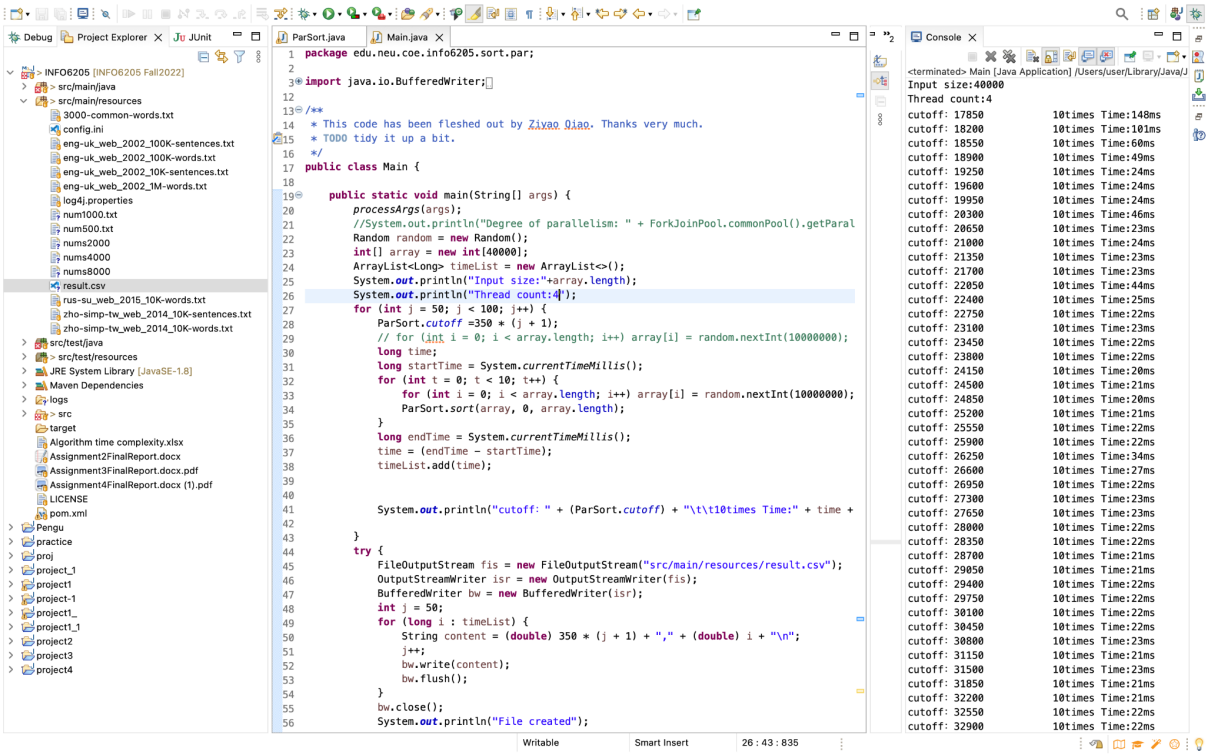

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.BufferedWriter;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
ublic class Main {
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getPara
Random random = new Random();
int[] array = new int[20000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:2");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =100 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000)
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 50;
for (long i : timeList) {
String content = (double) 50 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
Writable
Smart Insert
28 : 43 : 922
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:20000
Thread count:2
cutoff: 5100 10times Time:88ms
cutoff: 5200 10times Time:50ms
cutoff: 5300 10times Time:27ms
cutoff: 5400 10times Time:15ms
cutoff: 5500 10times Time:15ms
cutoff: 5600 10times Time:14ms
cutoff: 5700 10times Time:17ms
cutoff: 5800 10times Time:14ms
cutoff: 5900 10times Time:15ms
cutoff: 6000 10times Time:14ms
cutoff: 6100 10times Time:14ms
cutoff: 6200 10times Time:13ms
cutoff: 6300 10times Time:14ms
cutoff: 6400 10times Time:15ms
cutoff: 6500 10times Time:17ms
cutoff: 6600 10times Time:15ms
cutoff: 6700 10times Time:16ms
cutoff: 6800 10times Time:14ms
cutoff: 6900 10times Time:15ms
cutoff: 7000 10times Time:16ms
cutoff: 7100 10times Time:13ms
cutoff: 7200 10times Time:13ms
cutoff: 7300 10times Time:14ms
cutoff: 7400 10times Time:11ms
cutoff: 7500 10times Time:11ms
cutoff: 7600 10times Time:11ms
cutoff: 7700 10times Time:11ms
cutoff: 7800 10times Time:11ms
cutoff: 7900 10times Time:11ms
cutoff: 8000 10times Time:11ms
cutoff: 8100 10times Time:30ms
cutoff: 8200 10times Time:11ms
cutoff: 8300 10times Time:11ms
cutoff: 8400 10times Time:11ms
cutoff: 8500 10times Time:11ms
cutoff: 8600 10times Time:10ms
cutoff: 8700 10times Time:12ms
cutoff: 8800 10times Time:12ms
cutoff: 8900 10times Time:12ms
cutoff: 9000 10times Time:13ms
cutoff: 9100 10times Time:32ms
cutoff: 9200 10times Time:13ms
cutoff: 9300 10times Time:12ms
cutoff: 9400 10times Time:13ms

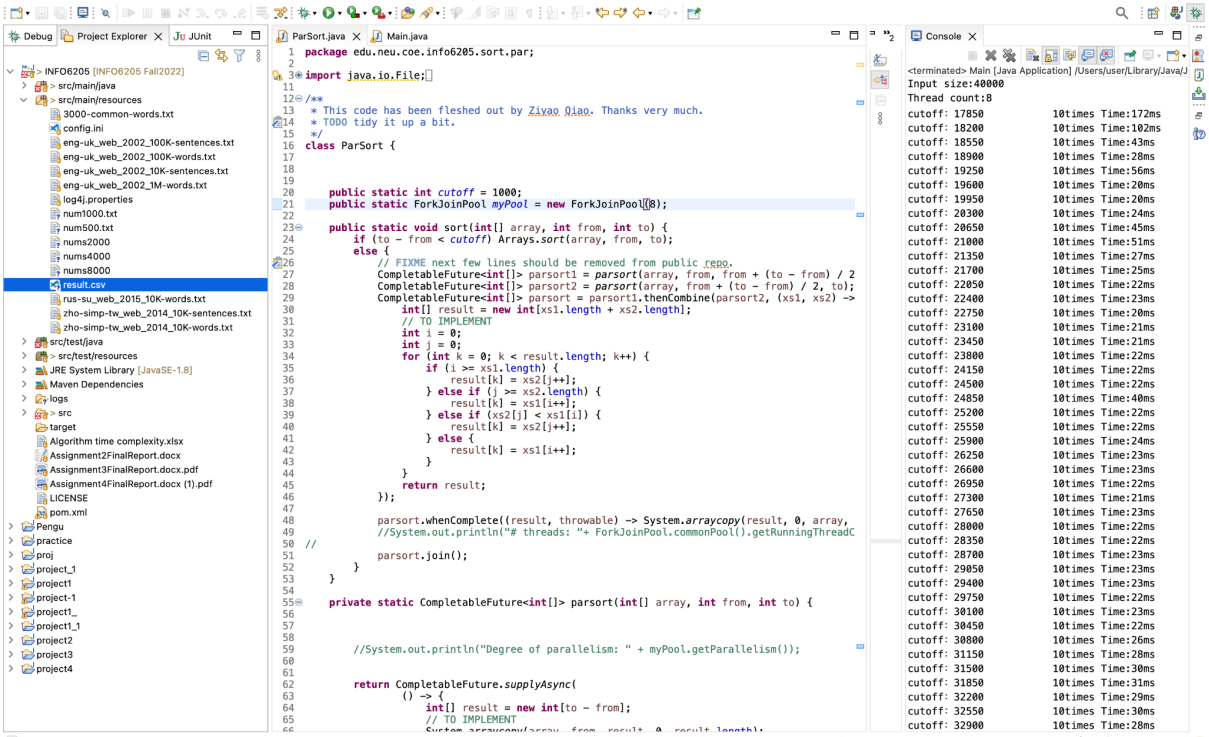

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(4);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) /
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2)
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThrea
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 59 : 534
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:20000
Thread count:4
cutoff: 5100 10times Time:79ms
cutoff: 5200 10times Time:56ms
cutoff: 5300 10times Time:43ms
cutoff: 5400 10times Time:14ms
cutoff: 5500 10times Time:14ms
cutoff: 5600 10times Time:13ms
cutoff: 5700 10times Time:15ms
cutoff: 5800 10times Time:12ms
cutoff: 5900 10times Time:11ms
cutoff: 6000 10times Time:12ms
cutoff: 6100 10times Time:11ms
cutoff: 6200 10times Time:12ms
cutoff: 6300 10times Time:12ms
cutoff: 6400 10times Time:13ms
cutoff: 6500 10times Time:13ms
cutoff: 6600 10times Time:13ms
cutoff: 6700 10times Time:26ms
cutoff: 6800 10times Time:22ms
cutoff: 6900 10times Time:13ms
cutoff: 7000 10times Time:12ms
cutoff: 7100 10times Time:12ms
cutoff: 7200 10times Time:13ms
cutoff: 7300 10times Time:13ms
cutoff: 7400 10times Time:12ms
cutoff: 7500 10times Time:11ms
cutoff: 7600 10times Time:11ms
cutoff: 7700 10times Time:11ms
cutoff: 7800 10times Time:26ms
cutoff: 7900 10times Time:19ms
cutoff: 8000 10times Time:12ms
cutoff: 8100 10times Time:12ms
cutoff: 8200 10times Time:11ms
cutoff: 8300 10times Time:11ms
cutoff: 8400 10times Time:11ms
cutoff: 8500 10times Time:11ms
cutoff: 8600 10times Time:12ms
cutoff: 8700 10times Time:12ms
cutoff: 8800 10times Time:13ms
cutoff: 8900 10times Time:13ms
cutoff: 9000 10times Time:15ms
cutoff: 9100 10times Time:14ms
cutoff: 9200 10times Time:14ms
cutoff: 9300 10times Time:13ms
cutoff: 9400 10times Time:13ms

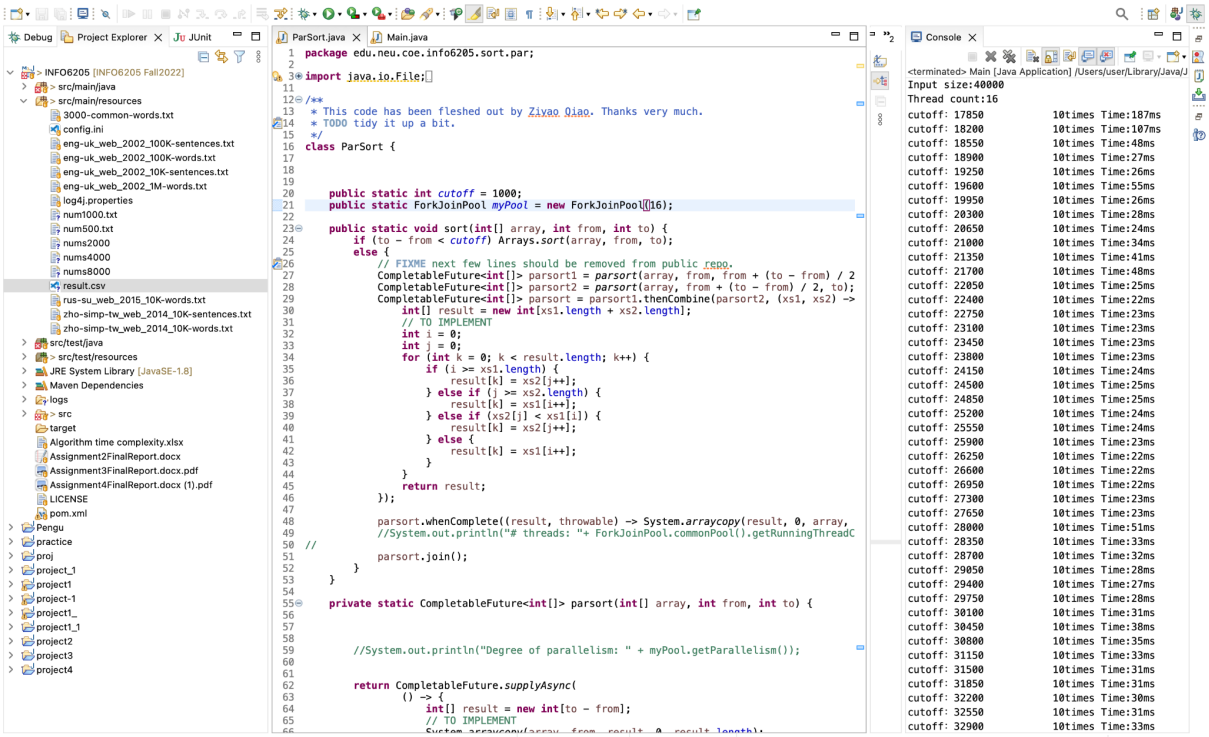

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(8);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) /
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2)
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThrea
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 59 : 534
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:20000
Thread count:8
cutoff: 5100 10times Time:86ms
cutoff: 5200 10times Time:62ms
cutoff: 5300 10times Time:61ms
cutoff: 5400 10times Time:51ms
cutoff: 5500 10times Time:34ms
cutoff: 5600 10times Time:14ms
cutoff: 5700 10times Time:12ms
cutoff: 5800 10times Time:11ms
cutoff: 5900 10times Time:9ms
cutoff: 6000 10times Time:10ms
cutoff: 6100 10times Time:9ms
cutoff: 6200 10times Time:9ms
cutoff: 6300 10times Time:11ms
cutoff: 6400 10times Time:9ms
cutoff: 6500 10times Time:10ms
cutoff: 6600 10times Time:10ms
cutoff: 6700 10times Time:9ms
cutoff: 6800 10times Time:10ms
cutoff: 6900 10times Time:10ms
cutoff: 7000 10times Time:10ms
cutoff: 7100 10times Time:10ms
cutoff: 7200 10times Time:10ms
cutoff: 7300 10times Time:11ms
cutoff: 7400 10times Time:36ms
cutoff: 7500 10times Time:31ms
cutoff: 7600 10times Time:13ms
cutoff: 7700 10times Time:11ms
cutoff: 7800 10times Time:13ms
cutoff: 7900 10times Time:35ms
cutoff: 8000 10times Time:10ms
cutoff: 8100 10times Time:10ms
cutoff: 8200 10times Time:10ms
cutoff: 8300 10times Time:10ms
cutoff: 8400 10times Time:9ms
cutoff: 8500 10times Time:9ms
cutoff: 8600 10times Time:9ms
cutoff: 8700 10times Time:9ms
cutoff: 8800 10times Time:9ms
cutoff: 8900 10times Time:9ms
cutoff: 9000 10times Time:10ms
cutoff: 9100 10times Time:9ms
cutoff: 9200 10times Time:11ms
cutoff: 9300 10times Time:10ms
cutoff: 9400 10times Time:10ms

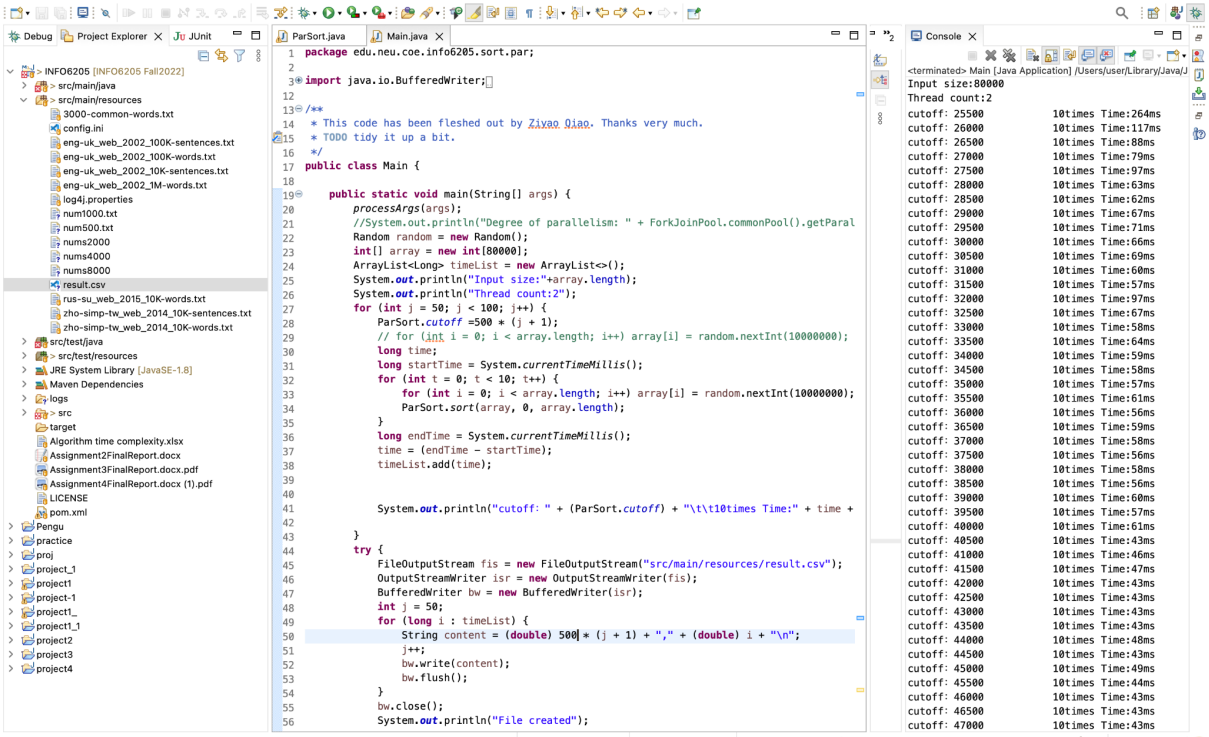

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getPara
Random random = new Random();
int[] array = new int[20000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:16");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =100 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000)
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 100;
for (long i : timeList) {
String content = (double) 50 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
} catch (IOException e) {
e.printStackTrace();
}
}
private static void processArgs(String[] args) {
String[] xs = args;
while (xs.length > 0)
if (xs[0].startsWith("-")) xs = processArg(xs);
Writable
Smart Insert
26 : 44 : 836
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:20000
Thread count:16
cutoff: 5100 10times Time:95ms
cutoff: 5200 10times Time:62ms
cutoff: 5300 10times Time:42ms
cutoff: 5400 10times Time:27ms
cutoff: 5500 10times Time:13ms
cutoff: 5600 10times Time:14ms
cutoff: 5700 10times Time:14ms
cutoff: 5800 10times Time:13ms
cutoff: 5900 10times Time:13ms
cutoff: 6000 10times Time:11ms
cutoff: 6100 10times Time:11ms
cutoff: 6200 10times Time:11ms
cutoff: 6300 10times Time:11ms
cutoff: 6400 10times Time:10ms
cutoff: 6500 10times Time:10ms
cutoff: 6600 10times Time:10ms
cutoff: 6700 10times Time:11ms
cutoff: 6800 10times Time:37ms
cutoff: 6900 10times Time:13ms
cutoff: 7000 10times Time:12ms
cutoff: 7100 10times Time:12ms
cutoff: 7200 10times Time:12ms
cutoff: 7300 10times Time:11ms
cutoff: 7400 10times Time:38ms
cutoff: 7500 10times Time:11ms
cutoff: 7600 10times Time:11ms
cutoff: 7700 10times Time:10ms
cutoff: 7800 10times Time:10ms
cutoff: 7900 10times Time:10ms
cutoff: 8000 10times Time:10ms
cutoff: 8100 10times Time:9ms
cutoff: 8200 10times Time:9ms
cutoff: 8300 10times Time:9ms
cutoff: 8400 10times Time:9ms
cutoff: 8500 10times Time:9ms
cutoff: 8600 10times Time:9ms
cutoff: 8700 10times Time:10ms
cutoff: 8800 10times Time:10ms
cutoff: 8900 10times Time:10ms
cutoff: 9000 10times Time:10ms
cutoff: 9100 10times Time:10ms
cutoff: 9200 10times Time:11ms
cutoff: 9300 10times Time:11ms
cutoff: 9400 10times Time:10ms

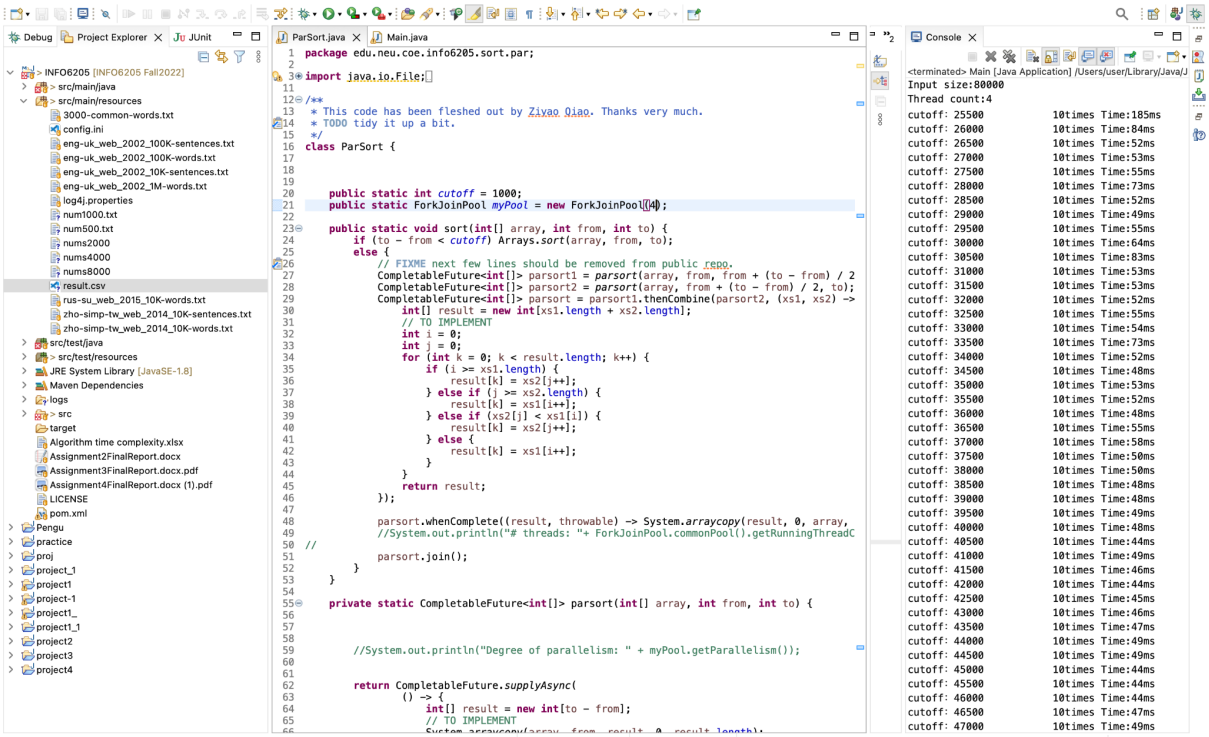

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getPara
Random random = new Random();
int[] array = new int[20000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:16");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =100 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000)
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 100;
for (long i : timeList) {
String content = (double) 50 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
} catch (IOException e) {
e.printStackTrace();
}
}
private static void processArgs(String[] args) {
String[] xs = args;
while (xs.length > 0)
if (xs[0].startsWith("-")) xs = processArg(xs);
Writable
Smart Insert
26 : 44 : 836
Console
<terminated> Main [Java Application] /Users/user/Library/Java/Java
Input size:20000
Thread count:16
cutoff: 5100 10times Time:95ms
cutoff: 5200 10times Time:62ms
cutoff: 5300 10times Time:42ms
cutoff: 5400 10times Time:27ms
cutoff: 5500 10times Time:13ms
cutoff: 5600 10times Time:14ms
cutoff: 5700 10times Time:14ms
cutoff: 5800 10times Time:13ms
cutoff: 5900 10times Time:13ms
cutoff: 6000 10times Time:11ms
cutoff: 6100 10times Time:11ms
cutoff: 6200 10times Time:11ms
cutoff: 6300 10times Time:11ms
cutoff: 6400 10times Time:10ms
cutoff: 6500 10times Time:10ms
cutoff: 6600 10times Time:10ms
cutoff: 6700 10times Time:11ms
cutoff: 6800 10times Time:37ms
cutoff: 6900 10times Time:13ms
cutoff: 7000 10times Time:12ms
cutoff: 7100 10times Time:12ms
cutoff: 7200 10times Time:12ms
cutoff: 7300 10times Time:11ms
cutoff: 7400 10times Time:38ms
cutoff: 7500 10times Time:11ms
cutoff: 7600 10times Time:11ms
cutoff: 7700 10times Time:10ms
cutoff: 7800 10times Time:10ms
cutoff: 7900 10times Time:10ms
cutoff: 8000 10times Time:10ms
cutoff: 8100 10times Time:9ms
cutoff: 8200 10times Time:9ms
cutoff: 8300 10times Time:9ms
cutoff: 8400 10times Time:9ms
cutoff: 8500 10times Time:9ms
cutoff: 8600 10times Time:9ms
cutoff: 8700 10times Time:10ms
cutoff: 8800 10times Time:10ms
cutoff: 8900 10times Time:10ms
cutoff: 9000 10times Time:10ms
cutoff: 9100 10times Time:10ms
cutoff: 9200 10times Time:11ms
cutoff: 9300 10times Time:11ms
cutoff: 9400 10times Time:10ms

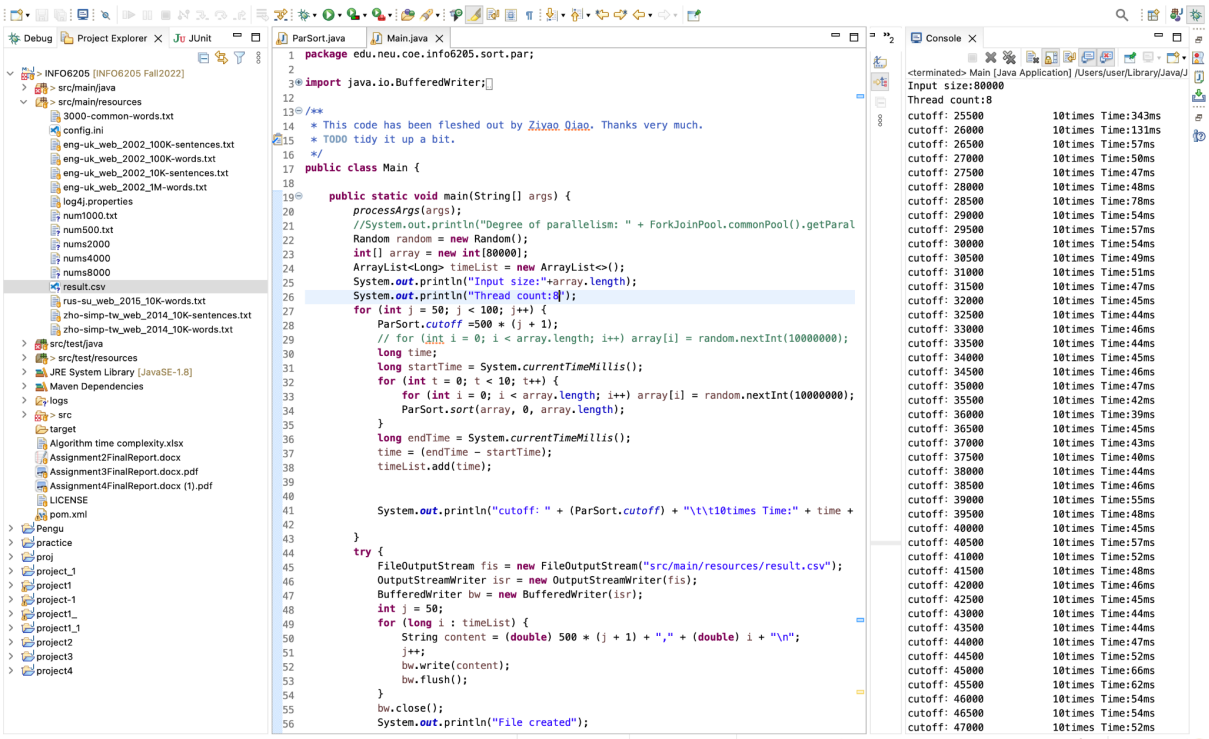

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.BufferedWriter;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
public class Main {
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getParal
Random random = new Random();
int[] array = new int[40000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:4");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =350 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time +
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 50;
for (long i : timeList) {
String content = (double) 350 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
Writable
Smart Insert
26 : 43 : 835
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:40000
Thread count:4
cutoff: 17850 10times Time:148ms
cutoff: 18200 10times Time:101ms
cutoff: 18550 10times Time:60ms
cutoff: 18900 10times Time:49ms
cutoff: 19250 10times Time:24ms
cutoff: 19600 10times Time:24ms
cutoff: 19950 10times Time:24ms
cutoff: 20300 10times Time:46ms
cutoff: 20650 10times Time:23ms
cutoff: 21000 10times Time:24ms
cutoff: 21350 10times Time:23ms
cutoff: 21700 10times Time:23ms
cutoff: 22050 10times Time:44ms
cutoff: 22400 10times Time:25ms
cutoff: 22750 10times Time:22ms
cutoff: 23100 10times Time:23ms
cutoff: 23450 10times Time:22ms
cutoff: 23800 10times Time:22ms
cutoff: 24150 10times Time:20ms
cutoff: 24500 10times Time:21ms
cutoff: 24850 10times Time:20ms
cutoff: 25200 10times Time:21ms
cutoff: 25550 10times Time:22ms
cutoff: 25900 10times Time:22ms
cutoff: 26250 10times Time:34ms
cutoff: 26600 10times Time:27ms
cutoff: 26950 10times Time:22ms
cutoff: 27300 10times Time:23ms
cutoff: 27650 10times Time:23ms
cutoff: 28000 10times Time:22ms
cutoff: 28350 10times Time:22ms
cutoff: 28700 10times Time:21ms
cutoff: 29050 10times Time:21ms
cutoff: 29400 10times Time:22ms
cutoff: 29750 10times Time:22ms
cutoff: 30100 10times Time:22ms
cutoff: 30450 10times Time:22ms
cutoff: 30800 10times Time:23ms
cutoff: 31150 10times Time:21ms
cutoff: 31500 10times Time:23ms
cutoff: 31850 10times Time:21ms
cutoff: 32200 10times Time:21ms
cutoff: 32550 10times Time:22ms
cutoff: 32900 10times Time:22ms

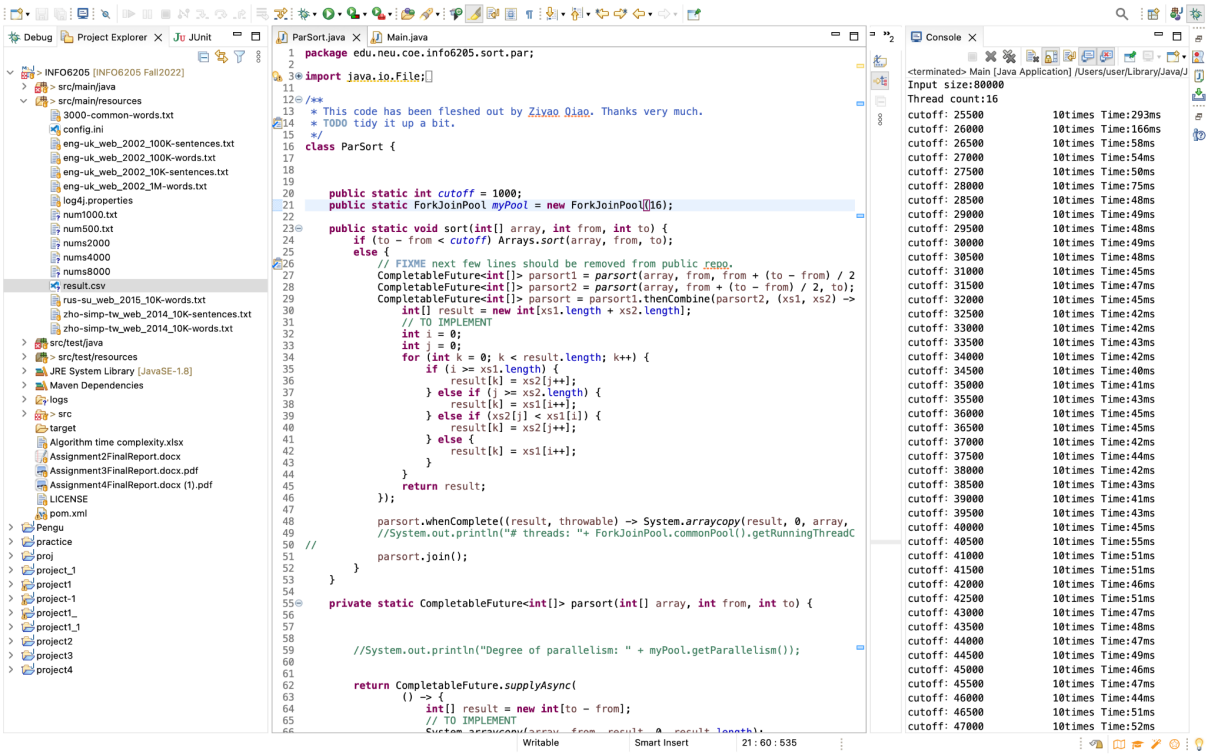

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(8);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:40000
Thread count:8
cutoff: 17850 10times Time:172ms
cutoff: 18200 10times Time:102ms
cutoff: 18550 10times Time:43ms
cutoff: 18900 10times Time:28ms
cutoff: 19250 10times Time:56ms
cutoff: 19600 10times Time:20ms
cutoff: 19950 10times Time:20ms
cutoff: 20300 10times Time:24ms
cutoff: 20650 10times Time:45ms
cutoff: 21000 10times Time:51ms
cutoff: 21350 10times Time:27ms
cutoff: 21700 10times Time:25ms
cutoff: 22050 10times Time:22ms
cutoff: 22400 10times Time:23ms
cutoff: 22750 10times Time:20ms
cutoff: 23100 10times Time:21ms
cutoff: 23450 10times Time:21ms
cutoff: 23800 10times Time:22ms
cutoff: 24150 10times Time:22ms
cutoff: 24500 10times Time:22ms
cutoff: 24850 10times Time:40ms
cutoff: 25200 10times Time:22ms
cutoff: 25550 10times Time:22ms
cutoff: 25900 10times Time:24ms
cutoff: 26250 10times Time:23ms
cutoff: 26600 10times Time:23ms
cutoff: 26950 10times Time:22ms
cutoff: 27300 10times Time:21ms
cutoff: 27650 10times Time:23ms
cutoff: 28000 10times Time:22ms
cutoff: 28350 10times Time:22ms
cutoff: 28700 10times Time:23ms
cutoff: 29050 10times Time:23ms
cutoff: 29400 10times Time:23ms
cutoff: 29750 10times Time:22ms
cutoff: 30100 10times Time:23ms
cutoff: 30450 10times Time:22ms
cutoff: 30800 10times Time:26ms
cutoff: 31150 10times Time:28ms
cutoff: 31500 10times Time:30ms
cutoff: 31850 10times Time:31ms
cutoff: 32200 10times Time:29ms
cutoff: 32550 10times Time:30ms
cutoff: 32900 10times Time:28ms
result.csv - INFO6205/src/main/resources

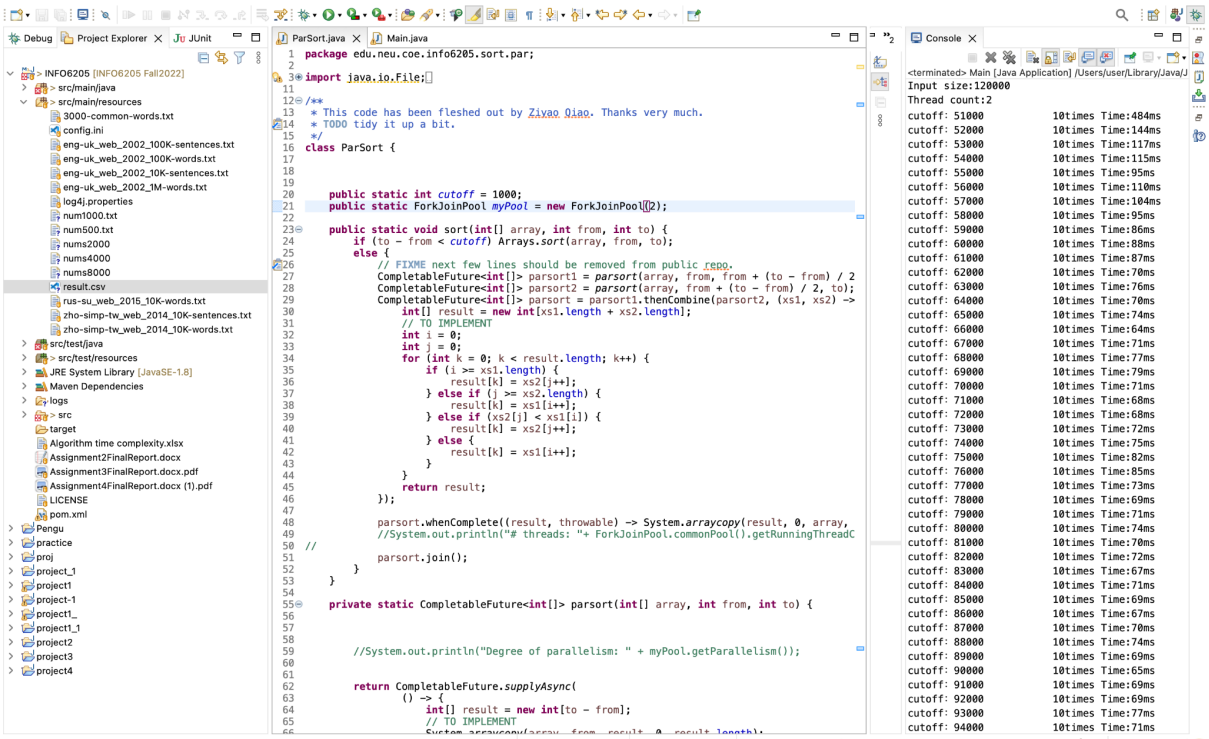

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(16);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:40000
Thread count:16
cutoff: 17850 10times Time:187ms
cutoff: 18200 10times Time:107ms
cutoff: 18550 10times Time:48ms
cutoff: 18900 10times Time:27ms
cutoff: 19250 10times Time:26ms
cutoff: 19600 10times Time:55ms
cutoff: 19950 10times Time:26ms
cutoff: 20300 10times Time:28ms
cutoff: 20650 10times Time:24ms
cutoff: 21000 10times Time:34ms
cutoff: 21350 10times Time:41ms
cutoff: 21700 10times Time:48ms
cutoff: 22050 10times Time:25ms
cutoff: 22400 10times Time:22ms
cutoff: 22750 10times Time:23ms
cutoff: 23100 10times Time:23ms
cutoff: 23450 10times Time:23ms
cutoff: 23800 10times Time:23ms
cutoff: 24150 10times Time:24ms
cutoff: 24500 10times Time:25ms
cutoff: 24850 10times Time:25ms
cutoff: 25200 10times Time:24ms
cutoff: 25550 10times Time:24ms
cutoff: 25900 10times Time:23ms
cutoff: 26250 10times Time:22ms
cutoff: 26600 10times Time:22ms
cutoff: 26950 10times Time:22ms
cutoff: 27300 10times Time:23ms
cutoff: 27650 10times Time:23ms
cutoff: 28000 10times Time:51ms
cutoff: 28350 10times Time:33ms
cutoff: 28700 10times Time:32ms
cutoff: 29050 10times Time:28ms
cutoff: 29400 10times Time:27ms
cutoff: 29750 10times Time:28ms
cutoff: 30100 10times Time:31ms
cutoff: 30450 10times Time:38ms
cutoff: 30800 10times Time:35ms
cutoff: 31150 10times Time:33ms
cutoff: 31500 10times Time:31ms
cutoff: 31850 10times Time:31ms
cutoff: 32200 10times Time:30ms
cutoff: 32550 10times Time:31ms
cutoff: 32900 10times Time:33ms
Writable
Smart Insert
21 : 60 : 535

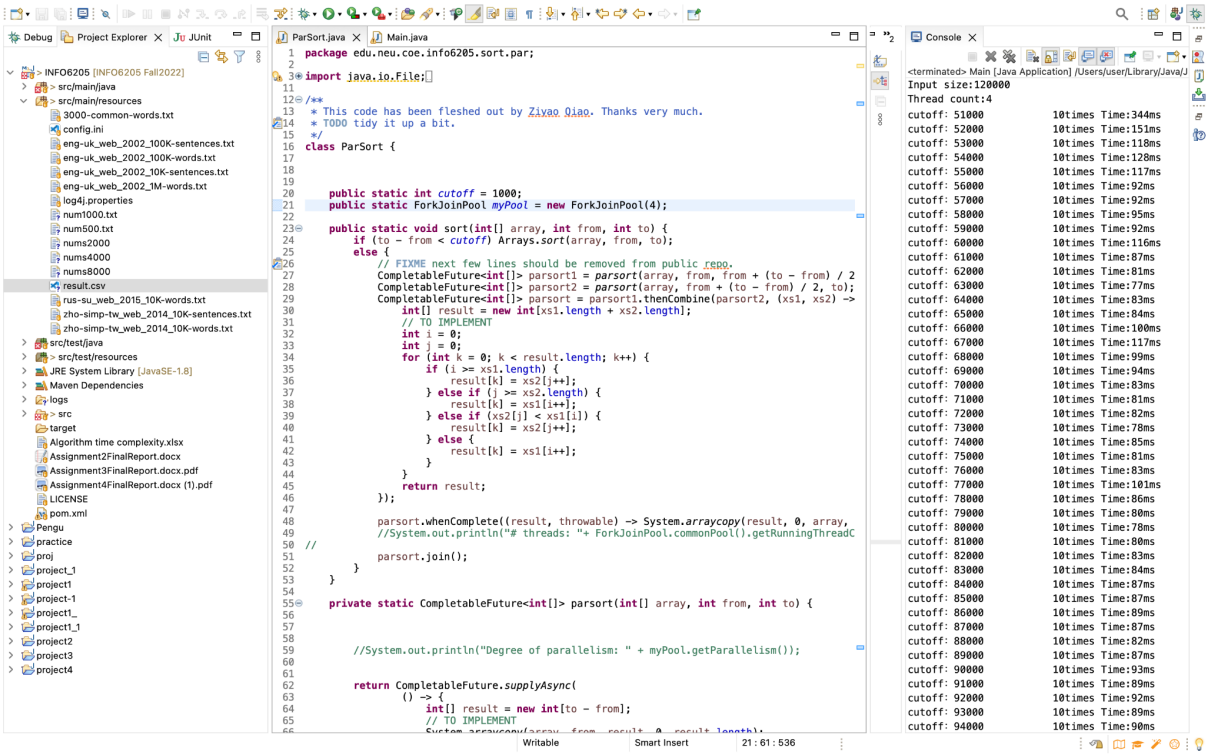

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.BufferedWriter;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
public class Main {
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getParal
Random random = new Random();
int[] array = new int[80000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:2");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =500 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time +
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 50;
for (long i : timeList) {
String content = (double) 500 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
Writable
Smart Insert
50 : 46 : 1871
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:80000
Thread count:2
cutoff: 25500 10times Time:264ms
cutoff: 26000 10times Time:117ms
cutoff: 26500 10times Time:88ms
cutoff: 27000 10times Time:79ms
cutoff: 27500 10times Time:97ms
cutoff: 28000 10times Time:63ms
cutoff: 28500 10times Time:62ms
cutoff: 29000 10times Time:67ms
cutoff: 29500 10times Time:71ms
cutoff: 30000 10times Time:66ms
cutoff: 30500 10times Time:69ms
cutoff: 31000 10times Time:60ms
cutoff: 31500 10times Time:57ms
cutoff: 32000 10times Time:97ms
cutoff: 32500 10times Time:67ms
cutoff: 33000 10times Time:58ms
cutoff: 33500 10times Time:64ms
cutoff: 34000 10times Time:59ms
cutoff: 34500 10times Time:58ms
cutoff: 35000 10times Time:57ms
cutoff: 35500 10times Time:61ms
cutoff: 36000 10times Time:56ms
cutoff: 36500 10times Time:59ms
cutoff: 37000 10times Time:58ms
cutoff: 37500 10times Time:56ms
cutoff: 38000 10times Time:58ms
cutoff: 38500 10times Time:56ms
cutoff: 39000 10times Time:60ms
cutoff: 39500 10times Time:57ms
cutoff: 40000 10times Time:61ms
cutoff: 40500 10times Time:43ms
cutoff: 41000 10times Time:46ms
cutoff: 41500 10times Time:47ms
cutoff: 42000 10times Time:43ms
cutoff: 42500 10times Time:43ms
cutoff: 43000 10times Time:43ms
cutoff: 43500 10times Time:43ms
cutoff: 44000 10times Time:48ms
cutoff: 44500 10times Time:43ms
cutoff: 45000 10times Time:49ms
cutoff: 45500 10times Time:44ms
cutoff: 46000 10times Time:43ms
cutoff: 46500 10times Time:43ms
cutoff: 47000 10times Time:43ms

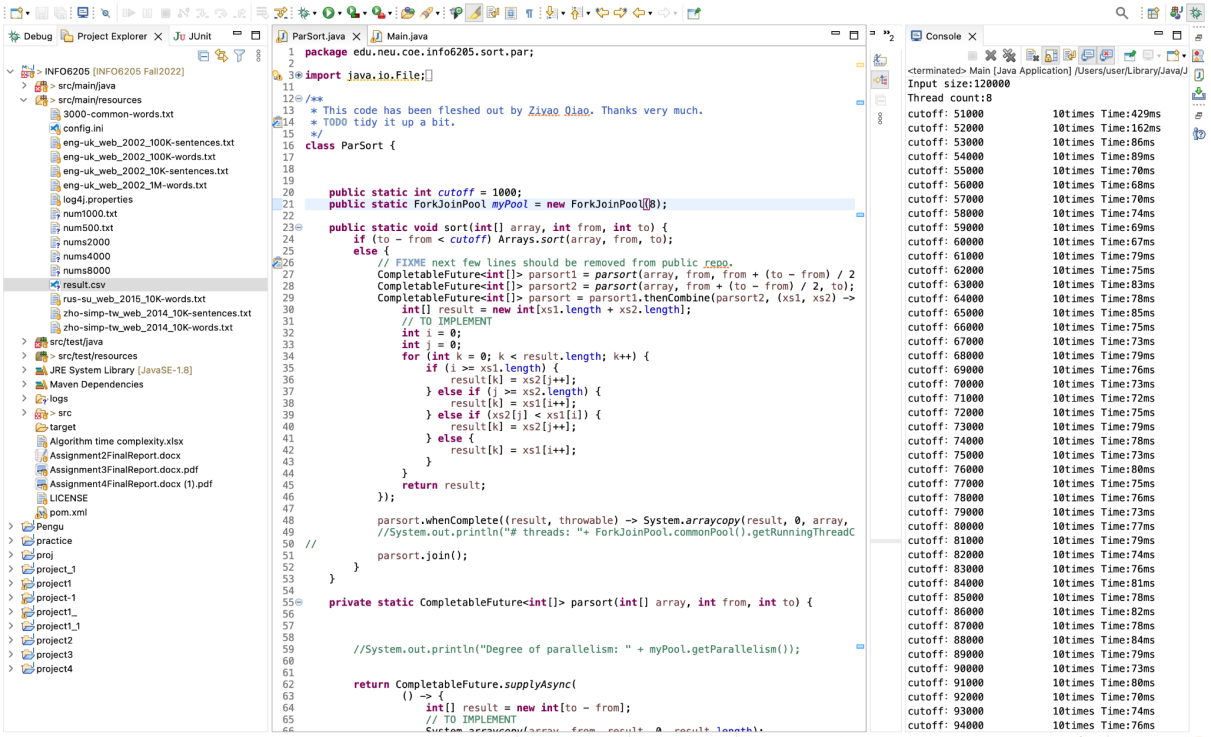

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(4);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 59 : 534
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:80000
Thread count:4
cutoff: 25500 10times Time:185ms
cutoff: 26000 10times Time:84ms
cutoff: 26500 10times Time:52ms
cutoff: 27000 10times Time:53ms
cutoff: 27500 10times Time:55ms
cutoff: 28000 10times Time:73ms
cutoff: 28500 10times Time:52ms
cutoff: 29000 10times Time:49ms
cutoff: 29500 10times Time:55ms
cutoff: 30000 10times Time:64ms
cutoff: 30500 10times Time:83ms
cutoff: 31000 10times Time:53ms
cutoff: 31500 10times Time:53ms
cutoff: 32000 10times Time:52ms
cutoff: 32500 10times Time:55ms
cutoff: 33000 10times Time:54ms
cutoff: 33500 10times Time:73ms
cutoff: 34000 10times Time:52ms
cutoff: 34500 10times Time:48ms
cutoff: 35000 10times Time:53ms
cutoff: 35500 10times Time:52ms
cutoff: 36000 10times Time:48ms
cutoff: 36500 10times Time:55ms
cutoff: 37000 10times Time:58ms
cutoff: 37500 10times Time:50ms
cutoff: 38000 10times Time:50ms
cutoff: 38500 10times Time:48ms
cutoff: 39000 10times Time:48ms
cutoff: 39500 10times Time:49ms
cutoff: 40000 10times Time:48ms
cutoff: 40500 10times Time:44ms
cutoff: 41000 10times Time:49ms
cutoff: 41500 10times Time:46ms
cutoff: 42000 10times Time:44ms
cutoff: 42500 10times Time:45ms
cutoff: 43000 10times Time:46ms
cutoff: 43500 10times Time:47ms
cutoff: 44000 10times Time:49ms
cutoff: 44500 10times Time:49ms
cutoff: 45000 10times Time:44ms
cutoff: 45500 10times Time:44ms
cutoff: 46000 10times Time:44ms
cutoff: 46500 10times Time:47ms
cutoff: 47000 10times Time:49ms

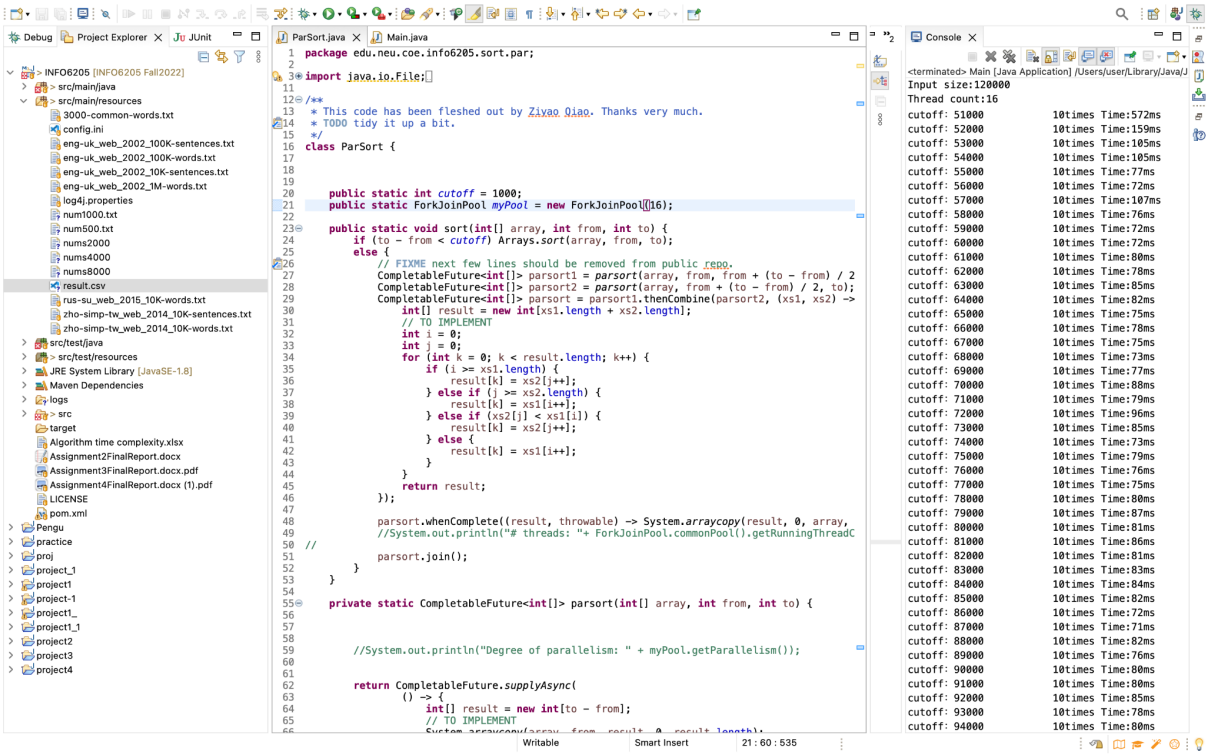

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.BufferedWriter;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
public class Main {
public static void main(String[] args) {
processArgs(args);
//System.out.println("Degree of parallelism: " + ForkJoinPool.commonPool().getParal
Random random = new Random();
int[] array = new int[80000];
ArrayList<Long> timeList = new ArrayList<>();
System.out.println("Input size:"+array.length);
System.out.println("Thread count:8");
for (int j = 50; j < 100; j++) {
ParSort.cutoff =500 * (j + 1);
// for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
long time;
long startTime = System.currentTimeMillis();
for (int t = 0; t < 10; t++) {
for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
ParSort.sort(array, 0, array.length);
}
long endTime = System.currentTimeMillis();
time = (endTime - startTime);
timeList.add(time);
System.out.println("cutoff: " + (ParSort.cutoff) + "\t\t10times Time:" + time +
}
try {
FileOutputStream fis = new FileOutputStream("src/main/resources/result.csv");
OutputStreamWriter isr = new OutputStreamWriter(fis);
BufferedWriter bw = new BufferedWriter(isr);
int j = 50;
for (long i : timeList) {
String content = (double) 500 * (j + 1) + "," + (double) i + "\n";
j++;
bw.write(content);
bw.flush();
}
bw.close();
System.out.println("File created");
Writable
Smart Insert
26 : 43 : 835
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:80000
Thread count:8
cutoff: 25500 10times Time:343ms
cutoff: 26000 10times Time:131ms
cutoff: 26500 10times Time:57ms
cutoff: 27000 10times Time:50ms
cutoff: 27500 10times Time:47ms
cutoff: 28000 10times Time:48ms
cutoff: 28500 10times Time:78ms
cutoff: 29000 10times Time:54ms
cutoff: 29500 10times Time:57ms
cutoff: 30000 10times Time:54ms
cutoff: 30500 10times Time:49ms
cutoff: 31000 10times Time:51ms
cutoff: 31500 10times Time:47ms
cutoff: 32000 10times Time:45ms
cutoff: 32500 10times Time:44ms
cutoff: 33000 10times Time:46ms
cutoff: 33500 10times Time:44ms
cutoff: 34000 10times Time:45ms
cutoff: 34500 10times Time:46ms
cutoff: 35000 10times Time:47ms
cutoff: 35500 10times Time:42ms
cutoff: 36000 10times Time:39ms
cutoff: 36500 10times Time:45ms
cutoff: 37000 10times Time:43ms
cutoff: 37500 10times Time:40ms
cutoff: 38000 10times Time:44ms
cutoff: 38500 10times Time:46ms
cutoff: 39000 10times Time:55ms
cutoff: 39500 10times Time:48ms
cutoff: 40000 10times Time:45ms
cutoff: 40500 10times Time:57ms
cutoff: 41000 10times Time:52ms
cutoff: 41500 10times Time:48ms
cutoff: 42000 10times Time:46ms
cutoff: 42500 10times Time:45ms
cutoff: 43000 10times Time:44ms
cutoff: 43500 10times Time:44ms
cutoff: 44000 10times Time:47ms
cutoff: 44500 10times Time:52ms
cutoff: 45000 10times Time:66ms
cutoff: 45500 10times Time:62ms
cutoff: 46000 10times Time:54ms
cutoff: 46500 10times Time:54ms
cutoff: 47000 10times Time:52ms


Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Zivao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(16);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 60 : 535
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:80000
Thread count:16
cutoff: 25500 10times Time:293ms
cutoff: 26000 10times Time:166ms
cutoff: 26500 10times Time:58ms
cutoff: 27000 10times Time:54ms
cutoff: 27500 10times Time:50ms
cutoff: 28000 10times Time:75ms
cutoff: 28500 10times Time:48ms
cutoff: 29000 10times Time:49ms
cutoff: 29500 10times Time:48ms
cutoff: 30000 10times Time:49ms
cutoff: 30500 10times Time:48ms
cutoff: 31000 10times Time:45ms
cutoff: 31500 10times Time:47ms
cutoff: 32000 10times Time:45ms
cutoff: 32500 10times Time:42ms
cutoff: 33000 10times Time:42ms
cutoff: 33500 10times Time:43ms
cutoff: 34000 10times Time:42ms
cutoff: 34500 10times Time:40ms
cutoff: 35000 10times Time:41ms
cutoff: 35500 10times Time:43ms
cutoff: 36000 10times Time:45ms
cutoff: 36500 10times Time:45ms
cutoff: 37000 10times Time:42ms
cutoff: 37500 10times Time:44ms
cutoff: 38000 10times Time:42ms
cutoff: 38500 10times Time:43ms
cutoff: 39000 10times Time:41ms
cutoff: 39500 10times Time:43ms
cutoff: 40000 10times Time:45ms
cutoff: 40500 10times Time:55ms
cutoff: 41000 10times Time:51ms
cutoff: 41500 10times Time:51ms
cutoff: 42000 10times Time:46ms
cutoff: 42500 10times Time:51ms
cutoff: 43000 10times Time:47ms
cutoff: 43500 10times Time:48ms
cutoff: 44000 10times Time:47ms
cutoff: 44500 10times Time:49ms
cutoff: 45000 10times Time:46ms
cutoff: 45500 10times Time:47ms
cutoff: 46000 10times Time:44ms
cutoff: 46500 10times Time:51ms
cutoff: 47000 10times Time:52ms


Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(2);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 59 : 534
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:120000
Thread count:2
cutoff: 51000 10times Time:484ms
cutoff: 52000 10times Time:144ms
cutoff: 53000 10times Time:117ms
cutoff: 54000 10times Time:115ms
cutoff: 55000 10times Time:95ms
cutoff: 56000 10times Time:110ms
cutoff: 57000 10times Time:104ms
cutoff: 58000 10times Time:95ms
cutoff: 59000 10times Time:86ms
cutoff: 60000 10times Time:88ms
cutoff: 61000 10times Time:87ms
cutoff: 62000 10times Time:70ms
cutoff: 63000 10times Time:76ms
cutoff: 64000 10times Time:70ms
cutoff: 65000 10times Time:74ms
cutoff: 66000 10times Time:64ms
cutoff: 67000 10times Time:71ms
cutoff: 68000 10times Time:77ms
cutoff: 69000 10times Time:79ms
cutoff: 70000 10times Time:71ms
cutoff: 71000 10times Time:68ms
cutoff: 72000 10times Time:68ms
cutoff: 73000 10times Time:72ms
cutoff: 74000 10times Time:75ms
cutoff: 75000 10times Time:82ms
cutoff: 76000 10times Time:85ms
cutoff: 77000 10times Time:73ms
cutoff: 78000 10times Time:69ms
cutoff: 79000 10times Time:71ms
cutoff: 80000 10times Time:74ms
cutoff: 81000 10times Time:70ms
cutoff: 82000 10times Time:72ms
cutoff: 83000 10times Time:67ms
cutoff: 84000 10times Time:71ms
cutoff: 85000 10times Time:69ms
cutoff: 86000 10times Time:67ms
cutoff: 87000 10times Time:70ms
cutoff: 88000 10times Time:74ms
cutoff: 89000 10times Time:69ms
cutoff: 90000 10times Time:65ms
cutoff: 91000 10times Time:69ms
cutoff: 92000 10times Time:69ms
cutoff: 93000 10times Time:77ms
cutoff: 94000 10times Time:71ms

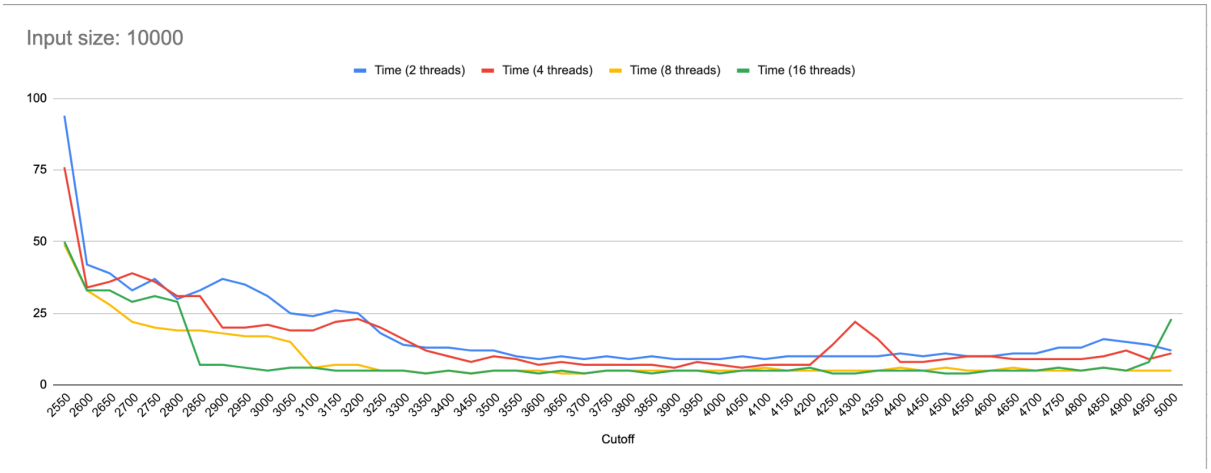

Debug
Project Explorer
JUnit
INFO6205 [INFO6205 Fall2022]
src/main/java
src/main/resources
3000-common-words.txt
config.ini
eng-uk_web_2002_100K-sentences.txt
eng-uk_web_2002_100K-words.txt
eng-uk_web_2002_10K-sentences.txt
eng-uk_web_2002_1M-words.txt
log4j.properties
num1000.txt
num500.txt
nums2000
nums4000
nums8000
result.csv
rus-su_web_2015_10K-words.txt
zho-simp-tw_web_2014_10K-sentences.txt
zho-simp-tw_web_2014_10K-words.txt
src/test/java
src/test/resources
JRE System Library [JavaSE-1.8]
Maven Dependencies
logs
src
target
Algorithm time complexity.xlsx
Assignment2FinalReport.docx
Assignment3FinalReport.docx.pdf
Assignment4FinalReport.docx (1).pdf
LICENSE
pom.xml
Pengu
practice
proj
project_1
project1
project-1
project1_
project1_1
project2
project3
project4
ParSort.java
Main.java
package edu.neu.coe.info6205.sort.par;
import java.io.File;
/**
* This code has been fleshed out by Ziyao Qiao. Thanks very much.
* TODO tidy it up a bit.
*/
class ParSort {
public static int cutoff = 1000;
public static ForkJoinPool myPool = new ForkJoinPool(4);
public static void sort(int[] array, int from, int to) {
if (to - from < cutoff) Arrays.sort(array, from, to);
else {
// FIXME next few lines should be removed from public repo.
CompletableFuture<int[]> parsort1 = parsort(array, from, from + (to - from) / 2
CompletableFuture<int[]> parsort2 = parsort(array, from + (to - from) / 2, to);
CompletableFuture<int[]> parsort = parsort1.thenCombine(parsort2, (xs1, xs2) ->
int[] result = new int[xs1.length + xs2.length];
// TO IMPLEMENT
int i = 0;
int j = 0;
for (int k = 0; k < result.length; k++) {
if (i >= xs1.length) {
result[k] = xs2[j++];
} else if (j >= xs2.length) {
result[k] = xs1[i++];
} else if (xs2[j] < xs1[i]) {
result[k] = xs2[j++];
} else {
result[k] = xs1[i++];
}
}
return result;
});
parsort.whenComplete((result, throwable) -> System.arraycopy(result, 0, array,
//System.out.println("# threads: "+ ForkJoinPool.commonPool().getRunningThreadC
//
parsort.join();
}
}
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
//System.out.println("Degree of parallelism: " + myPool.getParallelism());
return CompletableFuture.supplyAsync(
() -> {
int[] result = new int[to - from];
// TO IMPLEMENT
Writable
Smart Insert
21 : 61 : 536
Console
<terminated> Main [Java Application] /Users/user/Library/Java/J
Input size:120000
Thread count:4
cutoff: 51000 10times Time:344ms
cutoff: 52000 10times Time:151ms
cutoff: 53000 10times Time:118ms
cutoff: 54000 10times Time:128ms
cutoff: 55000 10times Time:117ms
cutoff: 56000 10times Time:92ms
cutoff: 57000 10times Time:92ms
cutoff: 58000 10times Time:95ms
cutoff: 59000 10times Time:92ms
cutoff: 60000 10times Time:116ms
cutoff: 61000 10times Time:87ms
cutoff: 62000 10times Time:81ms
cutoff: 63000 10times Time:77ms
cutoff: 64000 10times Time:83ms
cutoff: 65000 10times Time:84ms
cutoff: 66000 10times Time:100ms
cutoff: 67000 10times Time:117ms
cutoff: 68000 10times Time:99ms
cutoff: 69000 10times Time:94ms
cutoff: 70000 10times Time:83ms
cutoff: 71000 10times Time:81ms
cutoff: 72000 10times Time:82ms
cutoff: 73000 10times Time:78ms
cutoff: 74000 10times Time:85ms
cutoff: 75000 10times Time:81ms
cutoff: 76000 10times Time:83ms
cutoff: 77000 10times Time:101ms
cutoff: 78000 10times Time:86ms
cutoff: 79000 10times Time:80ms
cutoff: 80000 10times Time:78ms
cutoff: 81000 10times Time:80ms
cutoff: 82000 10times Time:83ms
cutoff: 83000 10times Time:84ms
cutoff: 84000 10times Time:87ms
cutoff: 85000 10times Time:87ms
cutoff: 86000 10times Time:89ms
cutoff: 87000 10times Time:87ms
cutoff: 88000 10times Time:82ms
cutoff: 89000 10times Time:87ms
cutoff: 90000 10times Time:93ms
cutoff: 91000 10times Time:89ms
cutoff: 92000 10times Time:92ms
cutoff: 93000 10times Time:89ms
cutoff: 94000 10times Time:90ms

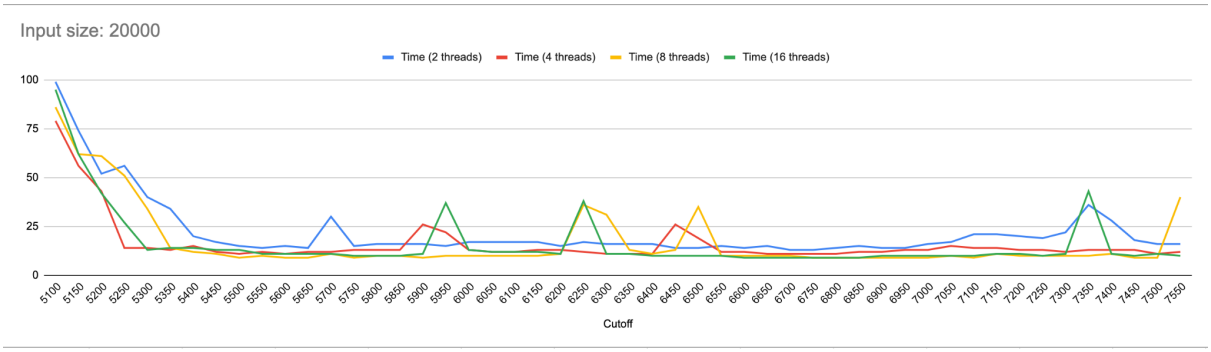


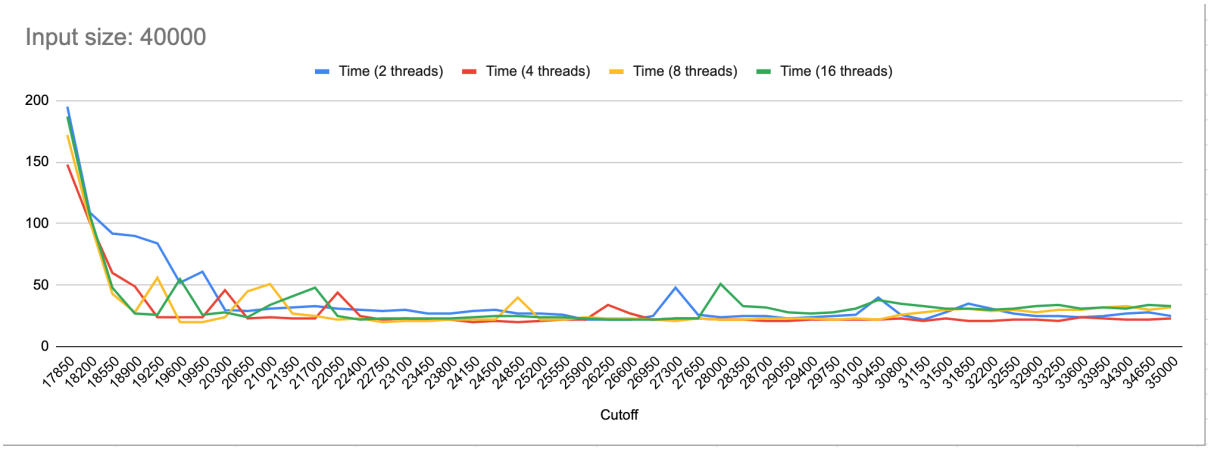


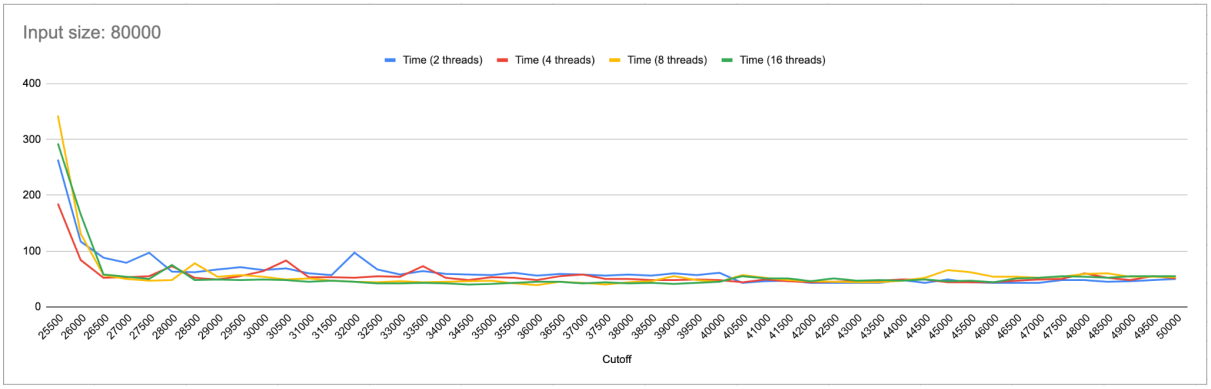
- CSV files attached on github for a better view of the cut-off values and time.

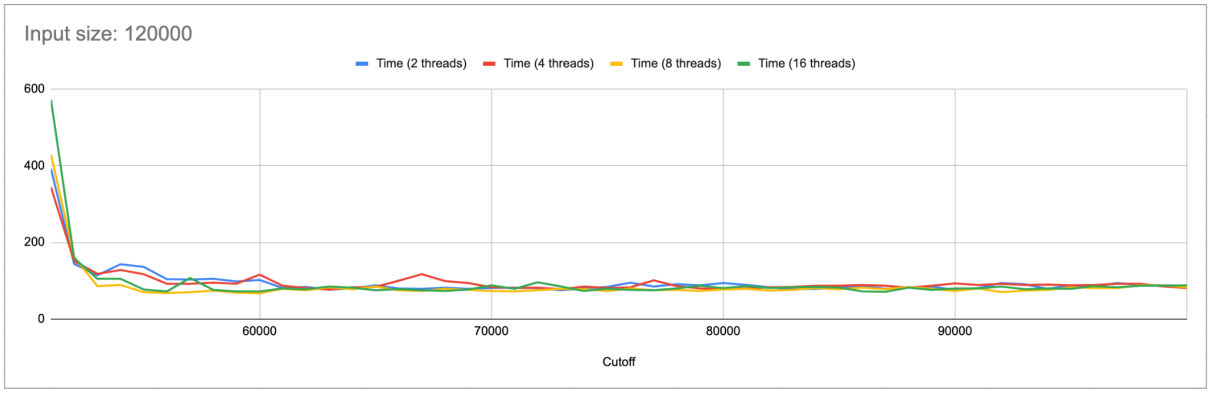
Graphical Representation:

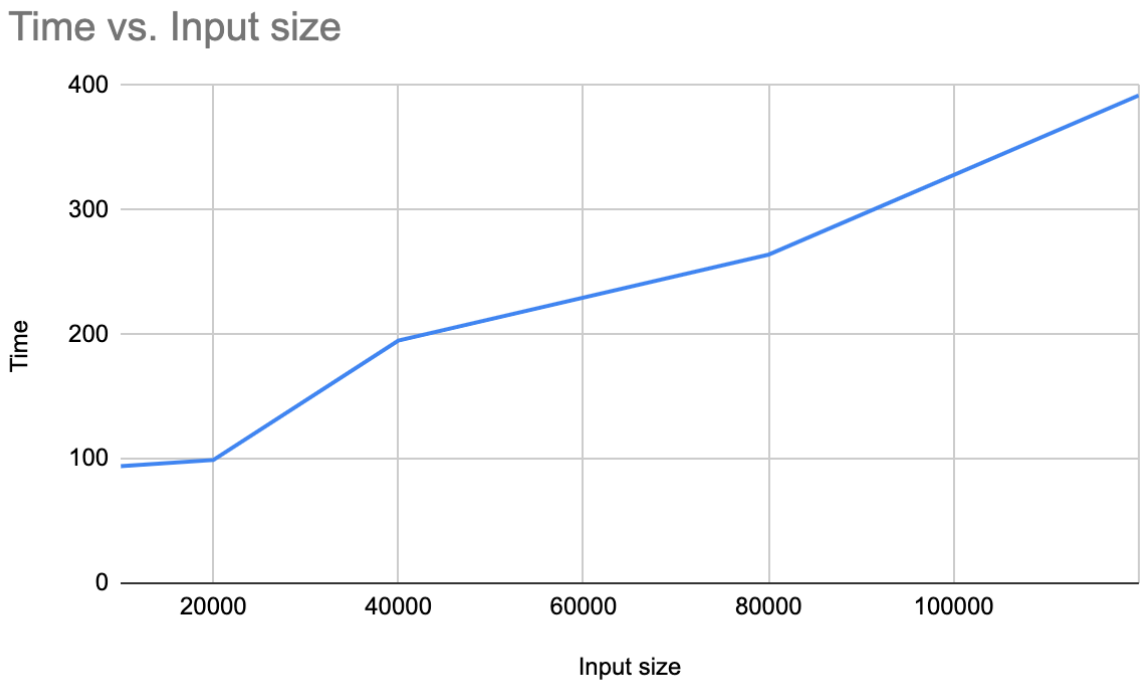







Input size: 80000
Time (2 threads)
Time (4 threads)
Time (8 threads)
Time (16 threads)
400
300
200
100
0
25500 26000 26500 27000 27500 28000 28500 29000 29500 30000 30500 31000 31500 32000 32500 33000 33500 34000 34500 35000 35500 36000 36500 37000 37500 38000 38500 39000 39500 40000 40500 41000 41500 42000 42500 43000 43500 44000 44500 45000 45500 46000 46500 47000 47500 48000 48500 49000 49500 50000
Cutoff


Input size: 120000
Time (2 threads)
Time (4 threads)
Time (8 threads)
Time (16 threads)
600
400
200
0
60000
70000
80000
90000
Cutoff


Time vs. Input size
400
300
200
100
0
Time
20000
40000
60000
80000
100000
Input size