Name of the Course : Begining Java Data Structures and Algorithms

Module : Sorting Algo and Fundamental Data Structures

Tool Stack : Java8 and Junit5

Problem Statement : Provide a code solution to sort the array in ascending order using Merge Sort.

Description : Create three classes one Array class with array[], beg, mid and end fields and with two parameterized constructors and another MergeSort class with two methods

1. void merge(Array array5), which accepts Array object and do merging.
2. int[] mergeSort(Array array1), which accepts Array object and returns array[] as an int[] array.

and one MainClass with one static method

1. public static void main method, for reading the size of the array and int array[] values from input devices and call the mergeSort method to do Merge Sort.

Code:

**package** yaksha;

**public** **class** Array {

**private** **int** array[];

**private** **int** beg;

**private** **int** mid;

**private** **int** end;

**public** Array(**int**[] array, **int** beg, **int** mid, **int** end) {

**super**();

**this**.array = array;

**this**.beg = beg;

**this**.mid = mid;

**this**.end = end;

}

**public** Array(**int**[] array, **int** beg, **int** end) {

**super**();

**this**.array = array;

**this**.beg = beg;

**this**.end = end;

}

**public** **int**[] getArray() {

**return** array;

}

**public** **void** setArray(**int**[] array) {

**this**.array = array;

}

**public** **int** getBeg() {

**return** beg;

}

**public** **void** setBeg(**int** beg) {

**this**.beg = beg;

}

**public** **int** getMid() {

**return** mid;

}

**public** **void** setMid(**int** mid) {

**this**.mid = mid;

}

**public** **int** getEnd() {

**return** end;

}

**public** **void** setEnd(**int** end) {

**this**.end = end;

}

}

**package** yaksha;

**public** **class** MergeSort {

**void** merge(Array array5) {

**int** arr[] = array5.getArray();

**int** beg = array5.getBeg();

**int** mid = array5.getMid();

**int** end = array5.getEnd();

**int** l = mid - beg + 1;

**int** r = end - mid;

**int** LeftArray[] = **new** **int**[l];

**int** RightArray[] = **new** **int**[r];

**for** (**int** i = 0; i < l; ++i)

LeftArray[i] = arr[beg + i];

**for** (**int** j = 0; j < r; ++j)

RightArray[j] = arr[mid + 1 + j];

**int** i = 0, j = 0;

**int** k = beg;

**while** (i < l && j < r) {

**if** (LeftArray[i] <= RightArray[j]) {

arr[k] = LeftArray[i];

i++;

} **else** {

arr[k] = RightArray[j];

j++;

}

k++;

}

**while** (i < l) {

arr[k] = LeftArray[i];

i++;

k++;

}

**while** (j < r) {

arr[k] = RightArray[j];

j++;

k++;

}

}

**int**[] mergeSort(Array array1) {

**int** array[] = array1.getArray();

**int** beg = array1.getBeg();

**int** end = array1.getEnd();

**if** (beg < end) {

**int** mid = (beg + end) / 2;

Array array2 = **new** Array(array, beg, mid);

mergeSort(array2);

Array array3 = **new** Array(array, mid + 1, end);

mergeSort(array3);

Array array4 = **new** Array(array, beg, mid, end);

merge(array4);

}

**return** array;

}

}

**package** yaksha;

**import** java.util.Scanner;

**public** **class** MainClass {

**public** **static** **void** main(String args[]) {

**int** i;

Scanner scanner = **new** Scanner(System.***in***);

**int**[] array;

**int** sizeOfArray;

System.***out***.println("Enter size of the Array :");

sizeOfArray = scanner.nextInt();

array = **new** **int**[sizeOfArray];

System.***out***.println("Enter numbers :");

**for** (i = 0; i < array.length; i++) {

array[i] = scanner.nextInt();

}

MergeSort ob = **new** MergeSort();

Array array1 = **new** Array(array, 0, array.length - 1);

**int** result[] = ob.mergeSort(array1);

System.***out***.println("\nSorted array");

**for** (i = 0; i < array.length; i++) {

System.***out***.println(result[i] + "");

}

scanner.close();

}

}

Junit Testing

**package** yaksha;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

// boiler-plate code

**public** **class** TestUtils {

**public** **static** File *businessTestFile*;

**public** **static** File *boundaryTestFile*;

**public** **static** File *exceptionTestFile*;

**static** {

*businessTestFile* = **new** File("./output\_revised.txt");

*businessTestFile*.delete();

*boundaryTestFile* = **new** File("./output\_boundary\_revised.txt");

*boundaryTestFile*.delete();

*exceptionTestFile* = **new** File("./output\_exception\_revised.txt");

*exceptionTestFile*.delete();

}

**public** **static** **void** yakshaAssert(String testName, Object result, File file) **throws** IOException {

System.***out***.println("\n" + testName + "=" + result);

FileWriter writer = **new** FileWriter(file, **true**);

writer.append("\n" + testName + "=" + result);

writer.flush();

writer.close();

}

**public** **static** String currentTest() {

**return** Thread.*currentThread*().getStackTrace()[2].getMethodName();

}

}

**package** yaksha;

**import** **static** yaksha.TestUtils.*businessTestFile*;

**import** **static** yaksha.TestUtils.*currentTest*;

**import** **static** yaksha.TestUtils.*yakshaAssert*;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.junit.jupiter.api.Test;

**public** **class** TestMainClass {

@Test

**public** **void** testExceptionConditon() **throws** Exception {

TestUtils.*yakshaAssert*(TestUtils.*currentTest*(), **true**, TestUtils.*exceptionTestFile*);

}

@Test

**public** **void** testBoundaryCondition() **throws** Exception {

TestUtils.*yakshaAssert*(TestUtils.*currentTest*(), **true**, TestUtils.*boundaryTestFile*);

}

@Test

**void** testMergeSort() **throws** Exception {

MergeSort mergeSort1 = **new** MergeSort();

Integer expectedResult[] = { 213, 219, 415, 615, 617, 819, 910, 1101 };

List<Integer> expectedResultList = Arrays.*asList*(expectedResult);

**int**[] array = { 910, 213, 1101, 415, 615, 219, 617, 819 };

// 910 213 1101 415 615 219 617 819

Array array1 = **new** Array(array, 0, array.length - 1);

**int** result[] = mergeSort1.mergeSort(array1);

List<Integer> resultList = **new** ArrayList<Integer>(result.length);

**for** (**int** i : result) {

resultList.add(i);

}

*yakshaAssert*(*currentTest*(), (expectedResultList.containsAll(resultList) ? "true" : "false"), *businessTestFile*);

}

}

pom.xml :

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>iiht.yaksha.mergesrt</groupId>

<artifactId>MrgSortBegJavaDSnAlgoSortAlgonFDS</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>MrgSortBegJavaDSnAlgoSortAlgonFDS</name>

<description>MrgSortBegJavaDSnAlgoSortAlgonFDS</description>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>${maven.compiler.source}</maven.compiler.target>

<junit.jupiter.version>5.5.2</junit.jupiter.version>

<junit.platform.version>1.5.2</junit.platform.version>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.12</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>${junit.jupiter.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-runner</artifactId>

<version>${junit.platform.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.11</version>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-all</artifactId>

<version>1.10.19</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.2</version>

</plugin>

</plugins>

</build>

</project>

Test Data1

Enter size of the Array :

8

Enter numbers :

910 213 1101 415 615 219 617 819

Sorted array

213

219

415

615

617

819

910

1101

Learning outcome: Participant could able to learn algorithm and implementation of Merge Sort.