Name of the Course : Advanced Data Structures and Algorithms in Java 9

Level : Medium

Tool Stack : Java9 and Junit5

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

Description : Create three classes one Array class with arr[] field and with a parameterized constructor and another HeapSort class with four static methods

1. public static void swap(int arr[], int i, int j), which accepts three parameters and do swapping.
2. public static void heap(int arr[], int i), which accepts two parameters and create MAX heap.
3. public static void heapMethod(int arr[]), which accepts one parameter and calls heap method.
4. public static int[] heapSort(Array array), which accepts Array object and returns arr[] as an int[] array.

and one MainClass with one static method

1. public static void main method, for reading the number of elements of the array and int arr[] values from input devices and call the heapSort method to do Heap Sort.

Code:

**package** yaksha;

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

**private** **int** arr[];

**public** Array(**int**[] arr) {

**super**();

**this**.arr = arr;

}

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

**return** arr;

}

**public** **void** setArr(**int**[] arr) {

**this**.arr = arr;

}

}

**package** yaksha;

**public** **class** HeapSort {

**private** **static** **int** *N*;

**public** **static** **int**[] heapSort(Array array) {

**int** arr[] = array.getArr();

*heapMethod*(arr);

**for** (**int** i = *N*; i > 0; i--) {

*swap*(arr, 0, i);

*N* = *N* - 1;

*heap*(arr, 0);

}

**return** arr;

}

**public** **static** **void** heapMethod(**int** arr[]) {

*N* = arr.length - 1;

**for** (**int** i = *N* / 2; i >= 0; i--)

*heap*(arr, i);

}

**public** **static** **void** heap(**int** arr[], **int** i) {

**int** left = 2 \* i;

**int** right = 2 \* i + 1;

**int** max = i;

**if** (left <= *N* && arr[left] > arr[i])

max = left;

**if** (right <= *N* && arr[right] > arr[max])

max = right;

**if** (max != i) {

*swap*(arr, i, max);

*heap*(arr, max);

}

}

**public** **static** **void** swap(**int** arr[], **int** i, **int** j) {

**int** tmp = arr[i];

arr[i] = arr[j];

arr[j] = tmp;

}

}

**package** yaksha;

**import** java.util.Scanner;

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

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

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

**int** n;

System.***out***.println("Enter the number of elements to be sorted:");

n = in.nextInt();

**int** arr[] = **new** **int**[n];

System.***out***.println("Enter " + n + " integer elements");

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

arr[i] = in.nextInt();

Array array = **new** Array(arr);

**int** result[] = HeapSort.*heapSort*(array);

System.***out***.println("After sorting ");

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

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

System.***out***.println();

in.close();

}

}

Junit Testing

**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.*boundaryTestFile*);

}

@Test

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

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

}

@Test

**void** testHeapSort() **throws** Exception {

Integer expectedResult[] = { 1, 32, 54, 67, 78, 99 };

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

**int**[] arr = { 99, 54, 67, 32, 1, 78 };

// 99 54 67 32 1 78

Array array = **new** Array(arr);

**int** result[] = HeapSort.*heapSort*(array);

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

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

resultList.add(i);

}

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

}

}

**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();

}

}

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.hs</groupId>

<artifactId>HeapSortADSnAlgJ9MQTwo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>HeapSortADSnAlgJ9MQTwo</name>

<description>HeapSortADSnAlgJ9MQTwo</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>

</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 the number of elements to be sorted:

6

Enter 6 integer elements

99 54 67 32 1 78

After sorting

1

32

54

67

78

99

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