Name of the Course : Ultimate Java Masterclass

Level : Medium

Tool Stack : Java8 and Junit5

Problem Statement : Provide a code solution to find out highest scorer student name from students details.

Description : Create three classes one StudentInformation class with information[] field and with a parameterized constructor, one custom exception class InvalidMarkException to handle invalid mark if any student’s score is less than 0 or greater than 100 in any subject and one MainClass with two static methods

1. public static String highestScorer(StudentInformation studentInformation) throws InvalidMarkException, which accepts StudentInformation object and returns highest scorer student name. Assume the result will have only one student with max mark.
2. public static void main method, for reading the students details from input devices and call the highestScorer method to find out highest scorer student name. Read n strings as input and stores them in a string array. The string consists of student information like name and obtained marks of three subjects. Example: name-mark1-mark2-mark3 [suresh-70-47-12] The mark would range between 0 – 100 (inclusive).

Code:

**package** yaksha;

**public** **class** StudentInformation {

**private** String information[];

**public** StudentInformation(String[] information) {

**super**();

**this**.information = information;

}

**public** String[] getInformation() {

**return** information;

}

**public** **void** setInformation(String[] information) {

**this**.information = information;

}

}

**package** yaksha;

**public** **class** InvalidMarkException **extends** Exception {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**public** InvalidMarkException(String s) {

**super**(s);

}

}

**package** yaksha;

**import** java.util.\*;

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

**public** **static** String highestScorer(StudentInformation studentInformation) **throws** InvalidMarkException {

String name = "";

**int** score = 0;

String information[] = studentInformation.getInformation();

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

StringTokenizer st = **new** StringTokenizer(information[i], "-");

String tname = st.nextToken();

**int** tscore = 0;

**for** (**int** j = 0; j < 3; j++) {

**int** cscore = Integer.*parseInt*(st.nextToken());

**if** (cscore < 0 || cscore > 100) {

**throw** **new** InvalidMarkException("Score in a subject must be between 0 to 100");

} **else** {

tscore = tscore + cscore;

}

}

**if** (tscore > score) {

score = tscore;

name = tname;

}

}

**return** name;

}

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

// **TODO** Auto-generated method stub

System.***out***.println("Enter number of students");

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

**int** size = Integer.*parseInt*(scanner.nextLine());

String information[] = **new** String[size];

System.***out***.println("Enter students details");

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

information[i] = scanner.nextLine();

StudentInformation studentInformation = **new** StudentInformation(information);

System.***out***.print("Highest scorer student is : ");

**try** {

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

} **catch** (InvalidMarkException m) {

System.***out***.println("Exception occured: " + m);

}

scanner.close();

}

}

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.mqone</groupId>

<artifactId>JavaMasterClassMq1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>JavaMasterClassMq1</name>

<description>JavaMasterClassMq1</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>

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

**class** MainClassTest {

@Test

**void** testHighestScorer() **throws** Exception {

// Test will pass

String information1[] = { "sunil-56-88-23", "bindul-88-70-10", "john-70-49-65" };

StudentInformation studentInformation1 = **new** StudentInformation(information1);

**try** {

*yakshaAssert*(*currentTest*(),

(MainClass.*highestScorer*(studentInformation1).equals("john") ? "true" : "false"), *businessTestFile*);

} **catch** (InvalidMarkException e) {

e.printStackTrace();

}

}

}

output\_revised.txt

testHighestScorer=true

testing-JavaMasterClassMq1.xml

<test-cases>

<cases xsi:type="java:com.assessment.data.TestCase">

<test-case-type>Functional</test-case-type>

<expected-ouput>true</expected-ouput>

<name>testHighestScorer</name>

<weight>10</weight>

<mandatory>true</mandatory>

<desc>Test to findout highest scores from given student details</desc>

</cases>

</test-cases>

Test Data1

Enter number of students

3

Enter students details

sunil-56-88-23

bindul-88-70-10

john-70-49-65

Highest scorer student is : john

Learning outcome: Participant could able to learn string array with StringTokenizer and custom exception.