<?xml version="1.0" encoding="UTF-8"?>  
<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>ru.netology</groupId>  
 <artifactId>Pr2Maven</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>11</maven.compiler.source>  
 <maven.compiler.target>11</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
 <dependencies>  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>5.4.2</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>RELEASE</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>RELEASE</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-surefire-plugin</artifactId>  
 <version>2.22.2</version>  
 </plugin>  
 </plugins>  
 </build>  
</project>

package ru.netology.statistic;  
  
public class StatisticsService {  
  
 public long calculateSum(long[] sales) {  
 long sum = 0;  
 for (long sale : sales) {  
 sum += sale;  
 }  
 return sum;  
 }  
  
 public long calculateAverage(long[] sales) {  
 long sum = 0;  
 long average = 0;  
 for (long sale : sales) {  
 sum += sale;  
 }  
 average = sum / sales.length;  
 return average;  
 }  
  
 public int maxSales(long[] sales) {  
 int maxMonth = 0;  
 int month = 0;  
 for (long sale : sales) {  
 if (sale >= sales[maxMonth]) {  
 maxMonth = month;  
 }  
 month = month + 1;  
 }  
 return maxMonth + 1;  
 }  
  
 public int minSales(long[] sales) {  
 int minMonth = 0;  
 int month = 0; // переменная для индекса рассматриваемого месяца в массиве  
 for (long sale : sales) {  
 // sales[minMonth] - продажи в месяце minMonth  
 // sale - продажи в рассматриваемом месяце  
 if (sale <= sales[minMonth]) {  
 minMonth = month;  
 }  
 month = month + 1; // следующий рассматриваемый месяц имеет номер на 1 больше  
 }  
 return minMonth + 1;  
 }  
  
 public long calculateLowAverageSales(long[] sales) {  
 long sum = 0;  
 long average = 0;  
 for (long sale : sales) {  
 sum += sale;  
 }  
 average = sum / sales.length;  
 int m = 0;  
 for (long sale : sales) {  
 if (sale < average) {  
 m = m + 1;  
 }  
 }  
 return m;  
 }  
  
 public long calculateUpAverageSales(long[] sales) {  
 long sum = 0;  
 long average = 0;  
 for (long sale : sales) {  
 sum += sale;  
 }  
 average = sum / sales.length;  
 int m = 0;  
 for (long sale : sales) {  
 if (sale > average) {  
 m = m + 1;  
 }  
 }  
 return m;  
 }  
}

package ru.netology.statistic;  
  
import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.*assertEquals*;  
  
class StatisticsServiceTest {  
  
 @Test  
 void calculateSum() {  
 StatisticsService service = new StatisticsService();  
  
  
 // вызываем целевой метод:  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 180;  
 long actual = service.calculateSum(sales);  
  
 // производим проверку (сравниваем ожидаемый и фактический):  
 *assertEquals*(expected, actual);  
 }  
  
 @Test  
 void calculateAverage() {  
 StatisticsService service = new StatisticsService();  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 15;  
 long actual = service.calculateAverage(sales);  
 *assertEquals*(expected, actual);  
 }  
  
 @Test  
 void maxSales() {  
 StatisticsService service = new StatisticsService();  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 8;  
 long actual = service.maxSales(sales);  
 *assertEquals*(expected, actual);  
 }  
  
 @Test  
 void minSales() {  
 StatisticsService service = new StatisticsService();  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 9;  
 long actual = service.minSales(sales);  
 *assertEquals*(expected, actual);  
 }  
  
 @Test  
 void calculateLowAverageSales() {  
 StatisticsService service = new StatisticsService();  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 5;  
 long actual = service.calculateLowAverageSales(sales);  
 *assertEquals*(expected, actual);  
 }  
  
 @Test  
 void calculateUpAverageSales() {  
 StatisticsService service = new StatisticsService();  
 long[] sales = {8, 15, 13, 15, 17, 20, 19, 20, 7, 14, 14, 18};  
 long expected = 5;  
 long actual = service.calculateUpAverageSales(sales);  
 *assertEquals*(expected, actual);  
 }  
}