/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package week2;

/\*\*

\*

\* @author 09092543

\*/

import java.util.Scanner;

public class Main {

private static Stats myStats = new ArrayStats();

private static TrendBoundableStats myArrayStats = new BStats(new ArrayStats());

private static TrendBoundableStats mySimpleStats = new BStats(new SimpleStats());

private static Scanner scan = new Scanner(System.in);

public static void main(String[] args) {

String option;

do {

System.out.println("Options are:");

System.out.println("AS: Add a value to the SimpleStats Collection");

System.out.println("PS: Print out SimpleStats statistics");

System.out.println("AA: Add a value to the ArrayStats Collection");

System.out.println("PA: Print out ArrayStats statistics");

System.out.print("Enter your option > ");

option = scan.nextLine();

if (option.equalsIgnoreCase("AS")) {

addValue(mySimpleStats);

}

if (option.equalsIgnoreCase("PS")) {

printStats(mySimpleStats);

}

if (option.equalsIgnoreCase("AA")) {

addValue(myArrayStats);

}

if (option.equalsIgnoreCase("PA")) {

printStats(myArrayStats);

}

System.out.println();

} while (!option.equalsIgnoreCase("Q"));

}

private static void addValue(Addable stats) {

System.out.print("Enter value to be added > ");

double d = scan.nextDouble();

stats.add(d);

}

private static void printStats(TrendBoundableStats stats) {

System.out.println("Min = " + stats.getMin());

System.out.println("Max = " + stats.getMax());

System.out.println("Count = " + stats.getCount());

System.out.println("Sum = " + stats.getSum());

System.out.println("Trend = " + stats.getTrend());

if (stats.getCount() > 0) {

System.out.println("Average = " + stats.getAverage());

}

}

}

package week2;

/\*\*

\*

\* @author 09092543

\*/

public class SimpleStats extends AbstractStats {

private double sum, lastElement,penultimateElement;

private int count;

public double getSum() {

return sum;

}

public int getCount() {

return count;

}

public void add(double element) {

//Added for ex7, stores the second to last value

penultimateElement = lastElement;

lastElement = element;

count++;

sum += element;

}

public double getLastElement() {

return lastElement;

}

//Added to ex7, returns the second to last element.

public double getPenultimateElement(){

return penultimateElement;

}

}

package week2;

/\*\*

\*

\* @author 09092543

\*/

public class ArrayStats extends AbstractStats {

public static final int MAX\_ELEMENTS = 1000;

protected double[] elements = new double[MAX\_ELEMENTS];

private int count;

public double getElement(int i) {

return elements[i];

}

public double getSum() {

double sum = 0;

for (int i = 0; i < count; i++) {

sum += getElement(i);

}

return sum;

}

public int getCount() {

return count;

}

public void add(double element) {

elements[count] = element;

count++;

}

//Added for ex7, returns ,last element in the array

public double getLastElement(){

if(count >=1){

return getElement(count-1);

}else{

return 0;

}

}

//Added for ex7, returns the second to last element in the array

public double getPenultimateElement(){

if(count >= 2){

return getElement(count - 2);

}else{

return 0;

}

}

}

public interface Stats extends Addable {

public double getSum();

public int getCount();

public double getAverage();

//Added for ex7, already implemented in SimpleStats, added to ArrayStats

public double getLastElement();

//Added for ex7, added to both ArrayStats and SimpleStats

public double getPenultimateElement();

}

public abstract class AbstractStats implements Stats {

public abstract double getSum();

public abstract int getCount();

public double getAverage() {return getSum()/getCount();};

public abstract void add(double element);

}

public interface Bounds extends Addable {

public double getMin();

public double getMax();

}

public class SimpleBounds implements Bounds {

private double min = Double.MAX\_VALUE;

private double max = Double.MIN\_VALUE;

public double getMin() {

return min;

}

public double getMax() {

return max;

}

public void add(double element) {

if (element > max) {

max = element;

}

if (element < min) {

min = element;

}

}

}

public class BStats implements TrendBoundableStats{

private Stats stats;

private double min = Double.MAX\_VALUE;

private double max = Double.MIN\_VALUE;

public BStats(Stats stats) {

this.stats = stats;

}

public double getMin() {

return min;

}

public double getMax() {

return max;

}

public void add(double element) {

if (element > max) {

max = element;

}

if (element < min) {

min = element;

}

stats.add(element);

}

public double getSum() {

return stats.getSum();

}

public int getCount() {

return stats.getCount();

}

public double getAverage() {

return stats.getAverage();

}

//Added for ex7

public double getLastElement(){

return stats.getLastElement();

}

//Added for ex7

public double getPenultimateElement(){

return stats.getPenultimateElement();

}

//Added for ex7

public double getTrend(){

double penultimateValue = stats.getPenultimateElement();

double lastValue = stats.getLastElement();

int count = stats.getCount();

if(count < 2){

return -2;

}

if(lastValue > penultimateValue){

return 1;

}

if(lastValue < penultimateValue){

return -1;

}

if(lastValue == penultimateValue){

return 0;

}

return -5;

}

}

public class TestBounds {

public static void main(String[] args) {

Bounds simple = new SimpleBounds();

Bounds bstats = new BStats(new ArrayStats());

simple.add(-1);

simple.add(450);

simple.add(1);

bstats.add(56);

bstats.add(34);

bstats.add(1);

System.out.println("SimpleBounds");

showBounds(simple);

System.out.println();

System.out.println("BStats");

showBounds(bstats);

}

private static void showBounds(Bounds bounds) {

System.out.println("Min = " + bounds.getMin());

System.out.println("Max = " + bounds.getMax());

}

}

public interface Addable {

public void add(double element);

}

public interface BoundableStats extends Bounds,Stats {

}

public interface Trend extends Addable {

public double getTrend();

}

public interface TrendBoundableStats extends Trend,BoundableStats {

}