**6 kyu**

**Rainfall**

3313284% of 490371of 2,478[g964](https://www.codewars.com/users/g964)

Java

* [TRAIN AGAIN](https://www.codewars.com/kata/rainfall/train/java)
* [NEXT KATA](https://www.codewars.com/trainer/java)

Details

[Solutions](https://www.codewars.com/kata/rainfall/solutions/java)

[Discourse (117)](https://www.codewars.com/kata/rainfall/discuss/java)

* Add to Collection
* |
* Share this kata:

dataand data1 are two strings with rainfall records of a few cities for months from January to December. The records of towns are separated by \n. The name of each town is followed by :.

data and towns can be seen in "Your Test Cases:".

**Task:**

* function: mean(town, strng) should return the average of rainfall for the city town and the strng data or data1 (In R and Julia this function is called avg).
* function: variance(town, strng) should return the variance of rainfall for the city town and the strng data or data1.

**Examples:**

mean("London", data), 51.19(9999999999996)

variance("London", data), 57.42(833333333374)

**Notes:**

* if functions mean or variance have as parameter town a city which has no records return -1 or -1.0 (depending on the language)
* Don't truncate or round: the tests will pass if abs(your\_result - test\_result) <= 1e-2 or abs((your\_result - test\_result) / test\_result) <= 1e-6 depending on the language.
* Shell tests only variance
* A ref: <http://www.mathsisfun.com/data/standard-deviation.html>
* data and data1 (can be named d0 and d1 depending on the language; see "Sample Tests:") are adapted from: [http://www.worldclimate.com](http://www.worldclimate.com/)

<https://www.codewars.com/kata/rainfall/java>

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication11;

import java.util.HashMap;

/\*\*

\*

\* @author Usuario

\*/

public class JavaApplication11 {

/\*\*

\* @param town

\* @param strng

\* @param args the command line arguments

\* @return

\*/

public static double mean(String town, String strng) {

// your code

String[] ciudades = strng.split("\n");

HashMap<String, String> diccio =

new HashMap();

for (String ciudade : ciudades) {

String[] sc = ciudade.split(":");

String ciudad = sc[0];

String cadena = sc[1];

// diccio[ciudad] = cadena;

diccio.put(ciudad, cadena);

}

if(!diccio.containsKey(town)) {

return -1.0;

}

String[] por\_meses = diccio.get(town).split(",");

double sum = 0.0;

for (String por\_mese : por\_meses) {

String[] ab = por\_mese.split(" ");

//Console.WriteLine(por\_meses[j]);

if(ab.length > 1) {

if (ab[1].length() > 0)

{

sum += Double.parseDouble(ab[1]);

} else {

return -1.0;

}

}

else {

return -1.0;

}

}

return sum / (por\_meses.length);

}

public static double variance(String town, String strng) {

// your code

String[] ciudades = strng.split("\n");

HashMap<String, String> diccio =

new HashMap();

for (String ciudade : ciudades) {

String[] sc = ciudade.split(":");

String ciudad = sc[0];

String cadena = sc[1];

diccio.put(ciudad, cadena);

}

if(!diccio.containsKey(town)) {

return -1.0;

}

String[] por\_meses = diccio.get(town).split(",");

double media = mean(town, strng);

double sum = 0.0;

for (String por\_mese : por\_meses) {

String[] ab = por\_mese.split(" ");

if(ab.length > 1) {

if (ab[1].length() > 0)

{

sum += (Double.parseDouble(ab[1]) - media)\* (Double.parseDouble(ab[1]) - media);

}

}

else {

return -1.0;

}

}

// Console.WriteLine(sum);

return sum / (por\_meses.length);

}

public static void main(String[] args) {

// TODO code application logic here

String data =

"Rome:Jan 81.2,Feb 63.2,Mar 70.3,Apr 55.7,May 53.0,Jun 36.4,Jul 17.5,Aug 27.5,Sep 60.9,Oct 117.7,Nov 111.0,Dec 97.9" +

"\n" +

"London:Jan 48.0,Feb 38.9,Mar 39.9,Apr 42.2,May 47.3,Jun 52.1,Jul 59.5,Aug 57.2,Sep 55.4,Oct 62.0,Nov 59.0,Dec 52.9" +

"\n" +

"Paris:Jan 182.3,Feb 120.6,Mar 158.1,Apr 204.9,May 323.1,Jun 300.5,Jul 236.8,Aug 192.9,Sep 66.3,Oct 63.3,Nov 83.2,Dec 154.7" +

"\n" +

"NY:Jan 108.7,Feb 101.8,Mar 131.9,Apr 93.5,May 98.8,Jun 93.6,Jul 102.2,Aug 131.8,Sep 92.0,Oct 82.3,Nov 107.8,Dec 94.2" +

"\n" +

"Vancouver:Jan 145.7,Feb 121.4,Mar 102.3,Apr 69.2,May 55.8,Jun 47.1,Jul 31.3,Aug 37.0,Sep 59.6,Oct 116.3,Nov 154.6,Dec 171.5" +

"\n" +

"Sydney:Jan 103.4,Feb 111.0,Mar 131.3,Apr 129.7,May 123.0,Jun 129.2,Jul 102.8,Aug 80.3,Sep 69.3,Oct 82.6,Nov 81.4,Dec 78.2" +

"\n" +

"Bangkok:Jan 10.6,Feb 28.2,Mar 30.7,Apr 71.8,May 189.4,Jun 151.7,Jul 158.2,Aug 187.0,Sep 319.9,Oct 230.8,Nov 57.3,Dec 9.4" +

"\n" +

"Tokyo:Jan 49.9,Feb 71.5,Mar 106.4,Apr 129.2,May 144.0,Jun 176.0,Jul 135.6,Aug 148.5,Sep 216.4,Oct 194.1,Nov 95.6,Dec 54.4" +

"\n" +

"Beijing:Jan 3.9,Feb 4.7,Mar 8.2,Apr 18.4,May 33.0,Jun 78.1,Jul 224.3,Aug 170.0,Sep 58.4,Oct 18.0,Nov 9.3,Dec 2.7" +

"\n" +

"Lima:Jan 1.2,Feb 0.9,Mar 0.7,Apr 0.4,May 0.6,Jun 1.8,Jul 4.4,Aug 3.1,Sep 3.3,Oct 1.7,Nov 0.5,Dec 0.7";

System.out.print( mean("London", data));

System.out.println();

System.out.print( variance("London", data));

}

}