ESCUELA COLOMBIANA DE INGENIERÍA
JULIO GARAVITO

VIGILADA MINEDUCACIÓN

SYSTEMS ENGINEERING

Arquitecturas Empresariales

Laboratory 1

Luis Daniel Benavides Navarro

Author:
Nicolás Aguilera Contreras

Contents

1 Prerequisites

These are the necessary installations to run the application on your computer:

Maven - Dependency Management
Java 8 - Development Environment
Git - Version Control System

2 Introduction

In this application a program is developed to calculate the mean and standard deviation of a set of n real numbers. The program reads the n real numbers from a file.

For this calculation, an own implementation of a LinkedList that is compatible with the Java api will be used

3 Concepts

3.1 Linked List

The LinkedList class is a collection which can contain many objects of the same type, just like the ArrayList.

The LinkedList class has all of the same methods as the ArrayList class because they both implement the List interface. This means that you can add items, change items, remove items and clear the list in the same way.
[1]

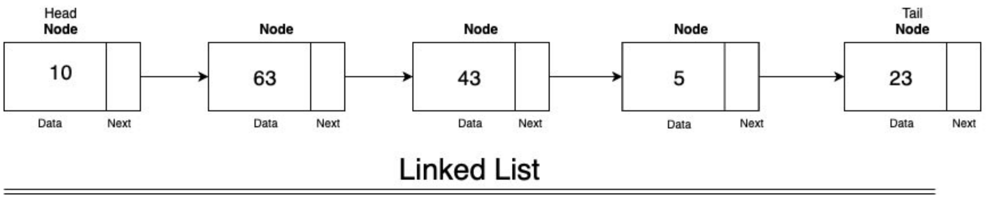


Figure 1: Linked list, *taken from* [2]

3.2 Mean

The mean is the average of a set of data. The average is the most common measure of location for a set of numbers. The average locates the center of the data.

The formula proposed in this laboratory for the calculation of the mean is the following

$$x_{avg} = \frac{\sum_{i=1}^{n} x_i}{n}$$

3.3 Standard Deviation

Standard deviation is a measure of the spread or dispersion of a set of data. Themore widely the values are spread out, the larger the standard deviation.
The formula for calculating the standard deviation is:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_{avg})^2}{n-1}}$$

4 Architecture

4.1 Class diagram of LinkeList

An own implementation of LinkedList is made. The class extends AbstractSecuentialList and implements the List, Deque, Clonable and Serializable interfaces to be compatible with the Java api. It also makes its own implementation of the Linked List Iterator, implementing from Iterator. Finally, it has its own implementation of the nodes that are part of the LinkeList, each of them stores the value and the reference to the next node.

Consequently, we have the following classes:

1. MyCustomLinkedList: Own implementation of a LinkeList in Java.
2. MyCustomNode: Represents a node of the LinkedList. It has value and reference to the next node.
3. MyCustomListIterator: Allows to iterate the LinkeList based on an index. Implements the Iterator interface.

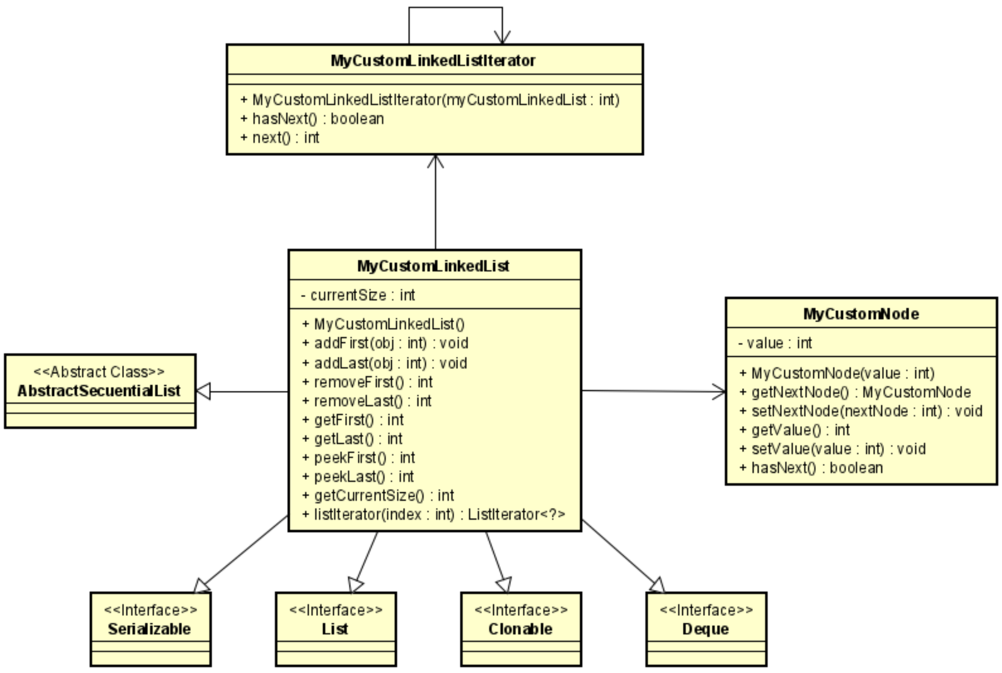


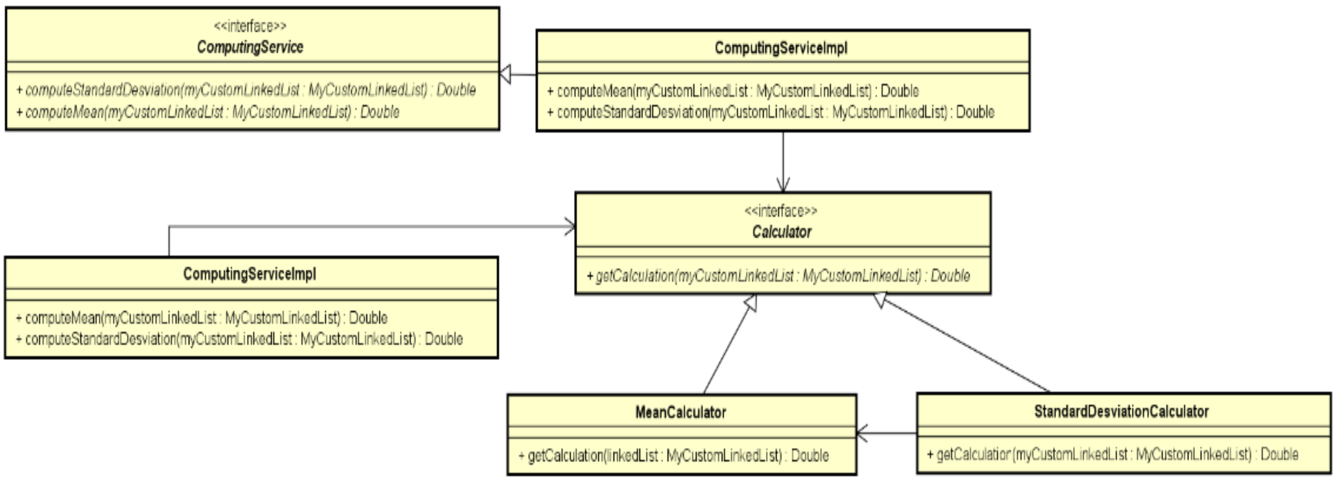
Figure 2: Class diagram

4.2 Class diagram of the computing of mean and standard deviation

Here is a class called ComputingService that will allow the user to access the application services. In this case, this class will connect with an interface called Calculator that will allow to perform an operation on a LinkedList. In this case, the standard deviation or the mean of the values in the LinkedList can be calculated. For that there are two classes called MeanCalculator and StandardDesviationCalculator that extend from the Calculator interface. This allows us to add more operations easily in case the user requires it.

In this case we are going to have the following classes:

1. Calculator (Interface): Interface that represents the calculations done on my Linked List
2. MeanCalculator: Class that computes the mean of a set of numbers on a Linked List
3. StandardDesviationCalculator: Class that computes the Standard Deviation of a set of numbers on a Linked List
4. Computing Service: Computes the Standard Deviation of a set of numbers on a Linked List
5. ComputingServiceImpl: Service that implements the computation of mean and standard deviation on a LinkedList

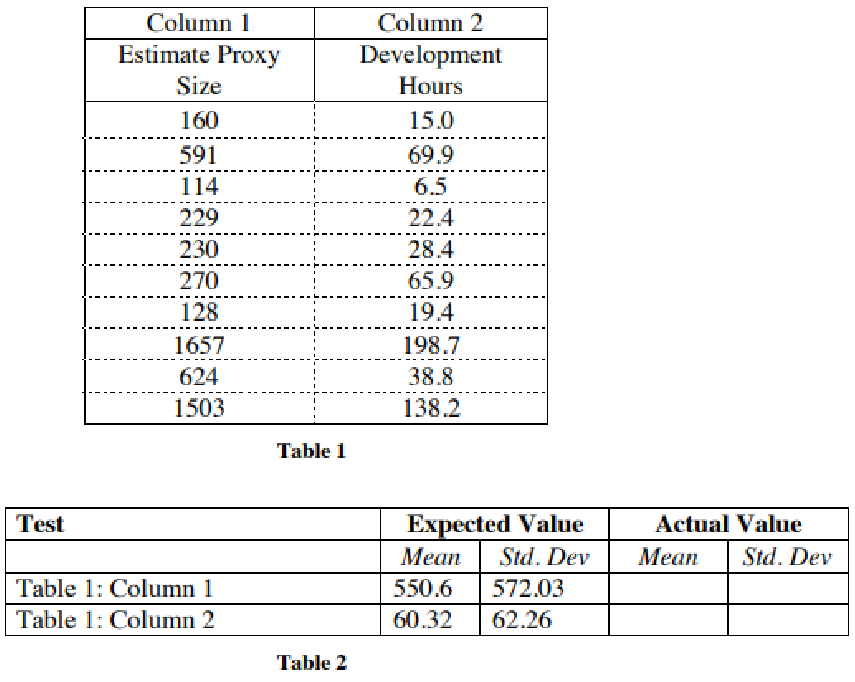


<<interface>>
ComputingService
+ computeStandardDesviation(myCustomLinkedList : MyCustomLinkedList) : Double
+ computeMean(myCustomLinkedList : MyCustomLinkedList) : Double
ComputingServiceImpl
+ computeMean(myCustomLinkedList : MyCustomLinkedList) : Double
+ computeStandardDesviation(myCustomLinkedList : MyCustomLinkedList) : Double
<<interface>>
Calculator
+ getCalculation(myCustomLinkedList : MyCustomLinkedList) : Double
ComputingServiceImpl
+ computeMean(myCustomLinkedList : MyCustomLinkedList) : Double
+ computeStandardDesviation(myCustomLinkedList : MyCustomLinkedList) : Double
MeanCalculator
+ getCalculation(linkedList : MyCustomLinkedList) : Double
StandardDesviationCalculator
+ getCalculation(myCustomLinkedList : MyCustomLinkedList) : Double

Figure 3: Class Diagram 2

5 Tests Cases

In this workshop we have two test cases along with their respective mean and standard deviation. In the following tables you can see the test data and the expected results.



Column 1	Column 2
Estimate Proxy Size	Development Hours
160	15.0
591	69.9
114	6.5
229	22.4
230	28.4
270	65.9
128	19.4
1657	198.7
624	38.8
1503	138.2

Table 1

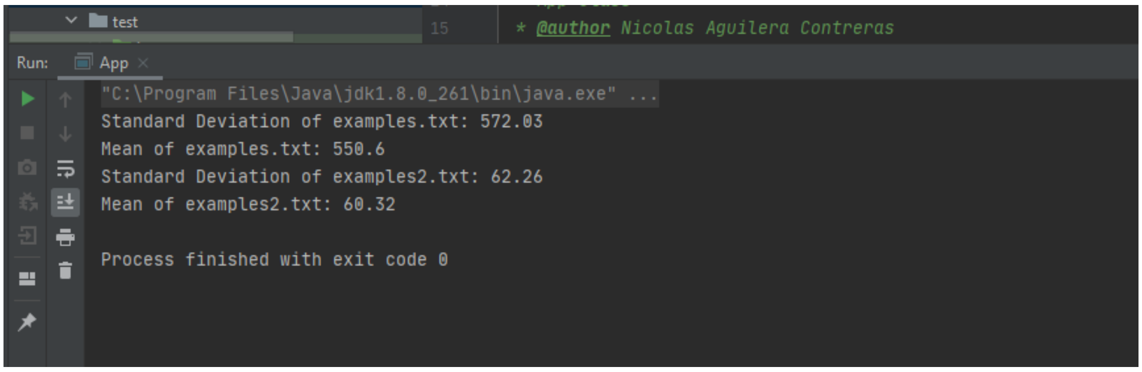
Test	**Expected Value**		**Actual Value**	
	Mean	*Std. Dev*	*Mean*	*Std. Dev*
Table 1: Column 1	550.6	572.03		
Table 1: Column 2	60.32	62.26		

Table 2

Figure 4: Test data

6 Results

Observing the results we can see that these are equal to the expected ones so the program behaves correctly when making the corresponding calculations.



```
* @author Nicolas Aguilera Contreras
```

```
"C:\Program Files\Java\jdk1.8.0_261\bin\java.exe" ...
Standard Deviation of examples.txt: 572.03
Mean of examples.txt: 550.6
Standard Deviation of examples2.txt: 62.26
Mean of examples2.txt: 60.32

Process finished with exit code 0
```

Figure 5: Results obtained

7 References

[1] W3Schools, *Java linkedlist*, `https://www.w3schools.com/java/java_linkedlist.asp`, Accessed on 2021-01-29.

[2] *Linked list, queue and stack-data structure algorithm part i*, https://dev.to/fernandoblima/linked-list-queue-and-stack-data-structure-part-i-1pen, Accessed on 2021-01-29, 2019.