HOMEWORK 1 AREM (INTRODUCTION TO COMPLEX SYSTEMS, JAVA, MVN, AND GIT)

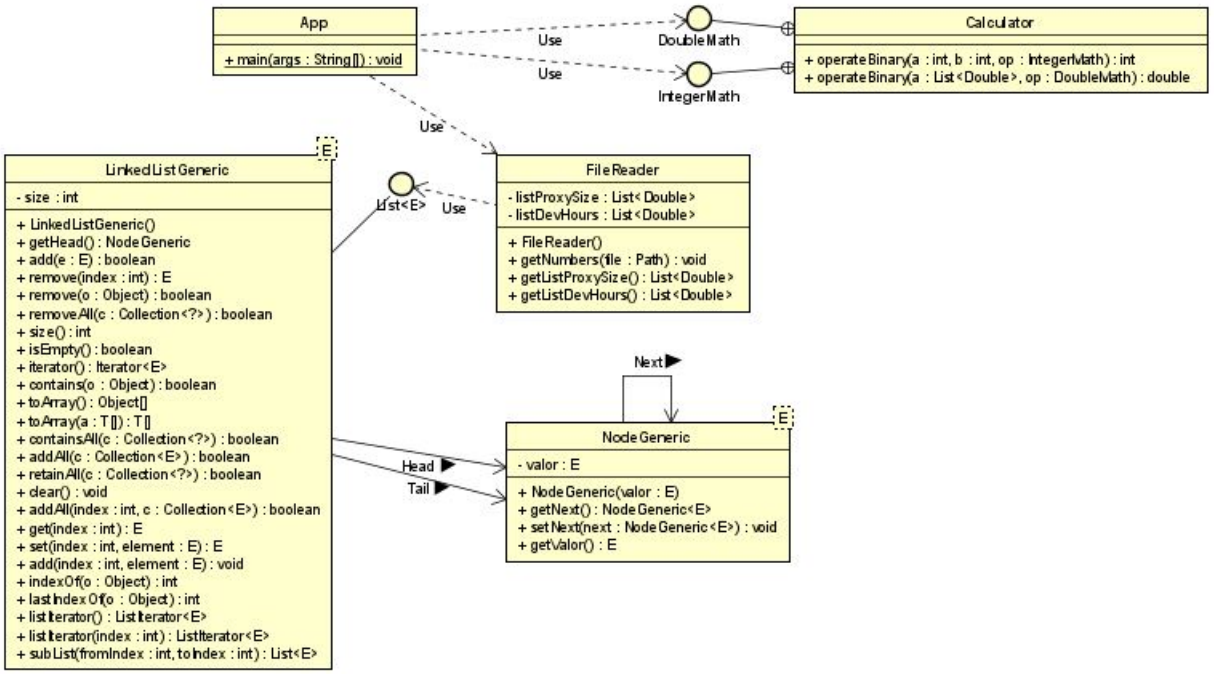
By : Juan Camilo Ortiz Medina

04 August 2020

1 Introduction

Write a program to calculate the mean and standard deviation of a set of n real numbers.

2 Design



3 Start

Copy the project through git clone in any direction to start working:

```
git clone https://github.com/Juaco9502/INTRODUCTION-TO-COMPLEX-SYSTEMS-JAVA-MVN-AND-GIT—AREP.git
```

4 Pre-Requisites

- Java jdk 7 or more.
- Maven (Apache Maven)
- Git

5 Install

5.1 Run in terminal:

This command works to compile the program:

```
mvn package
```

5.2 Documentation(Optional):

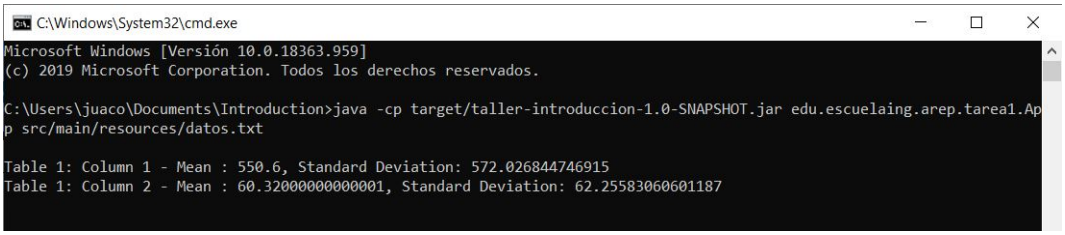
If you want to view the documentation of the application, execute the following command and check the following path: root / target:

```
mvn javadoc:javadoc
```

5.3 Run:

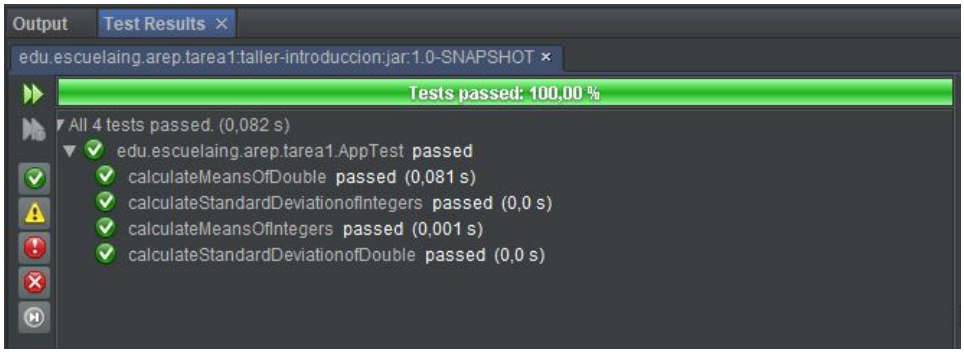
The following command must be used in the project root folder:

```
java -cp target/taller-introduccion-1.0-SNAPSHOT.jar edu.escuelaing.arep.tarea1.App src/main/resources/datos.txt
```



6 Test Report

6.1 Test Result



6.2 Test Cases

6.2.1 Test 1

En esta prueba se ingresaron los datos de la columna 1 , calculandole su media.

```
@Test
public void calculateMeansOfIntegers() throws Exception {

    Path file = Paths.get("src/test/resources/datos.txt");
    FileReader fileReader = new FileReader();
    //leer archivo y guardar las linkedList
    fileReader.getNumbers(file);

    //obtener LinkedList de ProxySize
    List<Double> listProxySize = fileReader.getListProxySize();

    //operar columna 1
    Double m1 = myApp.operateBinary(listProxySize, mean);

    assertEquals(550.6, m1,0.0001);
}
```

6.2.2 Test 2

En esta prueba se ingresaron los datos de la columna 1 , calculandole su desviación estandar.

```
@Test
public void calculateStandardDeviationofIntegers() throws Exception {

    Path file = Paths.get("src/test/resources/datos.txt");
    FileReader fileReader = new FileReader();
    //leer archivo y guardar las linkedList
    fileReader.getNumbers(file);

    //obtener LinkedList de ProxySize
    List<Double> listProxySize = fileReader.getListProxySize();

    //operar columna 1
    Double dl = myApp.operateBinary(listProxySize, sDeviation);

    assertEquals(572.03, dl,0.005);
}
```

6.2.3 Test 3

En esta prueba se ingresaron los datos de la columna 2 , calculandole su media.

```
@Test
public void calculateMeansOfDouble() throws Exception {

    Path file = Paths.get("src/test/resources/datos.txt");
    FileReader fileReader = new FileReader();
    //leer archivo y guardar las linkedList
    fileReader.getNumbers(file);

    //obtener LinkedList de ProxySize
    List<Double> listProxySize = fileReader.getListDevHours();

    //operar columna 1
    Double ml = myApp.operateBinary(listProxySize, mean);

    assertEquals(60.32, ml,0.0001);
}
```

6.2.4 Test 4

En esta prueba se ingresaron los datos de la columna 2 , calculandole su desviación estandar.

```
@Test
public void calculateStandardDeviationofDouble() throws Exception {

    Path file = Paths.get("src/test/resources/datos.txt");
    FileReader fileReader = new FileReader();
    //leer archivo y guardar las linkedList
    fileReader.getNumbers(file);

    //obtener LinkedList de ProxySize
    List<Double> listProxySize = fileReader.getListDevHours();

    //operar columna 1
    Double d1 = myApp.operateBinary(listProxySize, sDeviation);

    assertEquals(62.26, d1,0.005);
}
```

7 Pre-Requisites

- Maven - Dependency Management
- JAVA JDK 8 - Building
- JUnit 3.8.1 - Test

8 License

This project is licensed under the GNU General Public License.