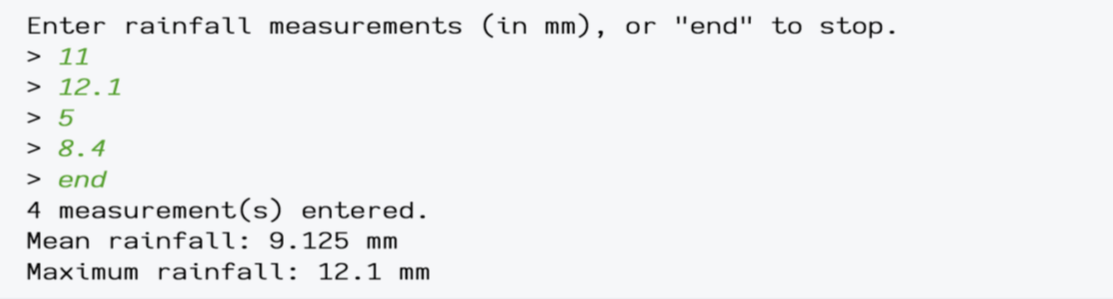
EXERCISES TO COMPLETE

Exercise 1 – Rainfall statistics

You are required to write a Java program which calculates statistics about rainfall measurements. The user enters rainfall measurements (in millimetres), and then enters the command (*end)* once all measurements have been entered. After this, the program prints:

- The number of measurements entered.
- The mean of the rainfall measurements.
- The maximum rainfall measurement.

For example, the following listing shows a typical example of using this program:



```
Enter rainfall measurements (in mm), or "end" to stop.
> 11
> 12.1
> 5
> 8.4
> end
4 measurement(s) entered.
Mean rainfall: 9.125 mm
Maximum rainfall: 12.1 mm
```

This project will include the following classes:

- RainfallStats — a class which records the measurements and computes statistics.
- RainfallProgram — the main program which handles user input/output.
- RainfallStatsTest — a unit test class for the model.
- InvalidRainfallException — an exception thrown when a measurement is negative. This class extends the (java.lang.Exception).

You are provided with partial versions of the RainfallProgram and RainfallStatsTest classes, which can be downloaded from the lab link on Moodle.

The 'RainfallStats' class

The RainfallStats class is the model for this program. It declares three variables `count`, `total` and `max` and initialise their values to 0. Also, it should have four methods:

- addMeasurement — adds a measurement (as a double).
- getCount — returns the number of measurements (as an int).
- getMean — returns the mean of the measurements (as a double).
- getMax — returns the largest measurement (as a double).

The addMeasurement Method

When a measurement is added:

- the count of measurements is incremented by 1
- the measurement value is added to the total measurements
- the measurement value entered is checked to determine the maximum measurement recorded

The addMeasurement method should throw an InvalidRainfallException if the measurement is negative. This is a custom **checked exception** which you must declare.

The getMean and getMax methods should each throw an IllegalStateException if no measurements have been added yet (the count of the measurements is equal to zero).

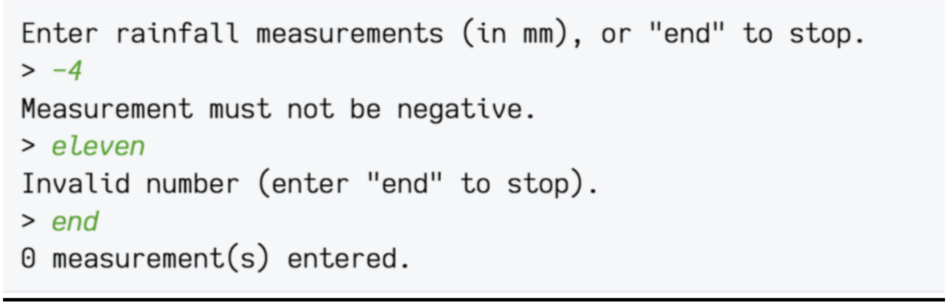
The RainfallProgram class reads measurements and returns a String value. However, when measurements are added to the system using the addMeasurement method they need to be of type double. Accordingly, measurement values need to be parsed from String to Double. Check the parseDouble class method of the Double class to help you convert a String to double. This method throws a NumberFormatException.

The 'RainfallProgram' class

The RainfallProgram class contains the program's main method. This method has a loop for the user to enter measurements, which stops when they enter the *end* command.

You are required to complete the main method by using a scanner object to read the measurements (double) and adding them to the stats object, and catching or preventing exceptions as appropriate.

The following listing shows examples of how error cases should be handled:



```
Enter rainfall measurements (in mm), or "end" to stop.
> -4
Measurement must not be negative.
> eleven
Invalid number (enter "end" to stop).
> end
0 measurement(s) entered.
```

Note that since no measurements were entered, the mean and maximum cannot be computed, so they should not be displayed.

The ‘RainfallStatsTest’ class

This is a unit test class for the RainfallStats class. There are three unwritten unit tests which you should complete by testing that the correct type of exception is thrown in each circumstance, as specified above.

When testing if an exception is thrown use the following syntax:

```
@Test (expected = ExceptionClassName.class)
    public void testName() {
        // TODO: write unit test
        Call the method that throws the test


    }
```

<u>Hint:</u> Check the Testing for exceptions section ***6-3 - specifying exceptions*** (https://moodle.bcu.ac.uk/mod/resource/view.php?id=7407948) on Moodle.