Exercise 02

This exercise will give you a chance to try out the scanf() and printf(), for reading formatted input and writing formatted output to the terminal. You'll write a program that reads a sequence of real values (as doubles) and reports the minimum value in the sequence, the maximum value and the average of all the values.

You'll find a partial implementation of a program, stats.c on the course homepage. You can download it from there, or, if you're already on a common platform system, you can use the following curl commands to download copies of the files right to your current directory.

```
curl -O https://www.csc2.ncsu.edu/courses/csc230/exercise/exercise02/stats.c
curl -O https://www.csc2.ncsu.edu/courses/csc230/exercise/exercise02/input.txt
curl -O https://www.csc2.ncsu.edu/courses/csc230/exercise/exercise02/expected.txt
```

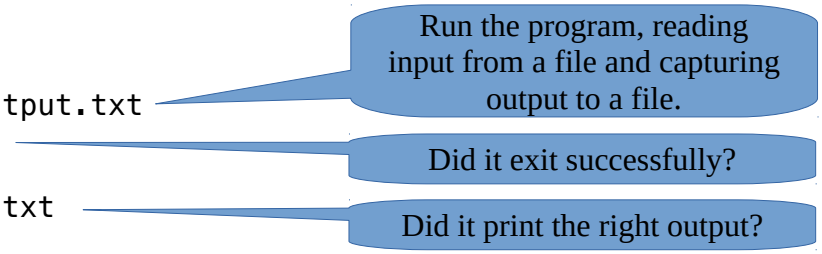
The stats.c program expects the first input value to be an integer, **n**, telling it how many real values to read. This is followed by a sequence of **n** real values. Input values may be separated by any amount of whitespace (which is easy with scanf(), since that's what it automatically does). After reading all the real values, the program will report the minimum value (of the real values, not counting the **n** at the start), the maximum value and the average in the format shown below. Each of these output lines starts with a word (Minimum, Maximum or Average), followed by a colon, then a space, then the value being reported. These values are printed with an 8-character field width, rounded to two fractional digits of precision.

The stats.c program you're getting is a partial implementation. You'll need to fill in some of the missing code to get it working. Once it's complete, you should be able to compile and run it as follows:

```
$ gcc -Wall -std=c99 stats.c -o stats
$ ./stats
7
98.6
5.18
3.14
25.39126
7.2934
16.384
15.1413
Minimum:     3.14
Maximum:    98.60
Average:    24.45
```

There are sample input and expected output provided with this exercise. These will make it easy to check that your program is producing exactly the right output. The sample is just like the example above, except with some spacing differences in the input. You can use shell commands like the following to try out your program on this sample.

```
$ ./stats < input.txt >| output.txt
$ echo $?
0
$ diff output.txt expected.txt
```



Once you're happy with how your program is working, submit your stats.c source file to the exercise_02 assignment in Moodle.