```
Script started on 2024-11-07 13:51:03-06:00 [TERM="xterm-256color" TTY="/dev/pts/6'
e_prevost@ares:~/Portfolio_2/Lab 2-new$ pwd
/home/students/e_prevost/Portfolio_2/Lab 2-new
e_prevost@ares:~/Portfolio_2/Lab 2-new$ cat roundemup.info
***************

Robert Prevost

CSC 122 W01



Round em up Lab



Takes in input and runs through library function to recursively validate

inputs until a correct input is met.



Base Level: Level 1.5

Bonus for input: +1 Level

Total Level: Level 2.5

***************e_prevost@ares:~/Portfolio_2/Lab 2-new$ show-code roundemup.cpp


roundemup.cpp:
```

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <cmath>
#include <limits>

std::string getFilename() {
    std::string filename;
    while (true) {
        std::cout << "Please enter the name of your data file: ";
        std::cin >> filename;
        std::ifstream file(filename);
        if (file.is_open()) {
            file.close();
            return filename;
        }
        std::cout << "I'm sorry, I could not open '" << filename <<
        "' Please enter another name:" << std::endl;
    }
}

class Statistics {
    public:
            int count;
            double minimum;
            double maximum;
            double average;
            double stdDev;
};

Statistics processFile(const std::string& filename) {
    std::ifstream file(filename);
    Statistics stats = {0, std::numeric_limits<double>::max(),
    std::numeric_limits<double>::lowest(), 0.0, 0.0};
    double num, sum = 0.0, sumSquares = 0.0;

    while (file >> num) {
        stats.count++;
        sum += num;
        sumSquares += num * num;
        stats.minimum = std::min(stats.minimum, num);
        stats.maximum = std::max(stats.maximum, num);
    }

    if (stats.count > 0) {
        stats.average = sum / stats.count;
        double variance = (sumSquares / stats.count) - (stats.average *
        stats.average);
        stats.stdDev = std::sqrt(variance);
    }

    return stats;
}

void printResults(const Statistics& stats) {
    if (stats.count == 0) {
        std::cout << "No valid data found in the file." << std::endl;
    } else {
        std::cout << "\nFor your data, the statistics are as follows:\n"
        << std::endl;

        // Set precision and fixed-point notation
        std::cout.precision(5);
        std::cout.setf(std::ios::fixed, std::ios::floatfield);

        std::cout << "     Count:  " << stats.count << std::endl;
        std::cout << "   Minimum:  " << stats.minimum << std::endl;
        std::cout << "   Average:  " << stats.average << std::endl;
        std::cout << "   Maximum:  " << stats.maximum << std::endl;
        std::cout << "    StdDev:  " << stats.stdDev << std::endl;

        // Optionally, reset the stream's formatting
        std::cout.unsetf(std::ios::floatfield);
    }
```

```
    75  }
    76
    77  int main() {
    78      std::cout << "\n                Welcome to your worst nightmare\n" <<
    79      std::endl;
    80
    81      std::string filename = getFilename();
    82      std::cout << "\nfile '" << filename << "' opened successfully" <<
    83      std::endl;
    84
    85      Statistics stats = processFile(filename);
    86
    87      std::cout << "\ndata process done" << std::endl;
    88      printResults(stats);
    89
    90
    91      return 0;
    92  }
e_prevost@ares:~/Portfolio_2/Lab 2-new$ CPP roundemup
roundemup.cpp***


e_prevost@ares:~/Portfolio_2/Lab 2-new$ ./roundemup.out

                Welcome to your worst nightmare

Please enter the name of your data file: data.txt

file 'data.txt' opened successfully

data process done

For your data, the statistics are as follows:

     Count:  10
   Minimum:  1.00000
   Average:  5.50000
   Maximum:  10.00000
    StdDev:  2.87228
e_prevost@ares:~/Portfolio_2/Lab 2-new$ cat data.txt
1 2 3 4

5 6 7 8



9 10e_prevost@ares:~/Portfolio_2/Lab 2-new$ cat roundemup.tpq
1. Does spacing between the numbers matter?



no spacing between the numbers does not matter because I read the lines

using the >> operator which skips whitespace when reading items.


2. Do they have to be in order (ascending/descending)?


no they dont have to be in order since we arent doing like median

calculations the actual order of the numbers does not matter.


3. Do they have to be integers? Floating point?


They can be both because I use double which will work for both integers and

floats.


4. Is it possible for your program to run out of space while reading the

file? (It shouldn't be...) (Hint: Do you have to have all of the data

values in memory to complete the calculations requested?)


no because I dont store any of the individual variables from the file I

only store the answers which doesn't take as much memory. It would be a

different story if i stored every variable because then we could have

1000's of variables which would lead to a memory storage issue.


5. When finding the largest/smallest item in a list, what value should you

start with as your assumedsmallest/largest value?


smallest should be the max int and largest should be the max lowest number

(or negative max number). This can be achieved using std::numeric_limits.e_prevost(
exit

Script done on 2024-11-07 13:53:01-06:00 [COMMAND_EXIT_CODE="0"]
```