EGR: 226 Microcontroller Programming and Applications

Winter 2021

Instructor Prof. Trevor Ekin

**Lab 2: Programming Refresher Part 2**

Gabriel Gasbarre

February 3rd, 2021

[1. Objectives 2](#_Toc62592272)

[2. Equipment 3](#_Toc62592273)

[3. Introduction 3](#_Toc62592274)

[3.1 Part 1: Resistor Analysis Tool 3](#_Toc62592275)

[3.2 Part 2: Statistics Library 4](#_Toc62592276)

[4. Procedure 4](#_Toc62592277)

[4.1 Part 1: Resistor analysis tool 4](#_Toc62592278)

[4.2 Part 2: Statistics Library 5](#_Toc62592279)

[5. Results 6](#_Toc62592280)

[5.1: Part 1 7](#_Toc62592281)

[5.2: Part 2 7](#_Toc62592282)

[6 Conclusions and Future Work 8](#_Toc62592283)

# Objectives

Lab 2 is split up into two parts. Part 1 builds upon the resistor analysis tool created in lab one with the added objective of practicing console C programming by creating a function that accepts color bands and outputs the respective resistance. Part 2 is designed with the objective to develop a searchable database of books using C-structures

# Equipment

|  |  |  |
| --- | --- | --- |
| Part | Description | Model |
| Code::Blocks | cross-platform IDE that supports multiple compilers | 20.03 |
| EGR:226 Structured Laboratory Activity | C programming refresher guide Part2 | N/A |
|  |  |  |

# Introduction

## 3.1 Part 1: Resistor Analysis Tool Part 2

For Lab 1, students were asked to create a C program with function prototypes to prompt the user to input a desired Ohm reading for a resistor. Lab 2 Part 2 asks students to update this analysis tool to allow the user the choice between entering resistance or a color code, outputting the conversion in either direction.

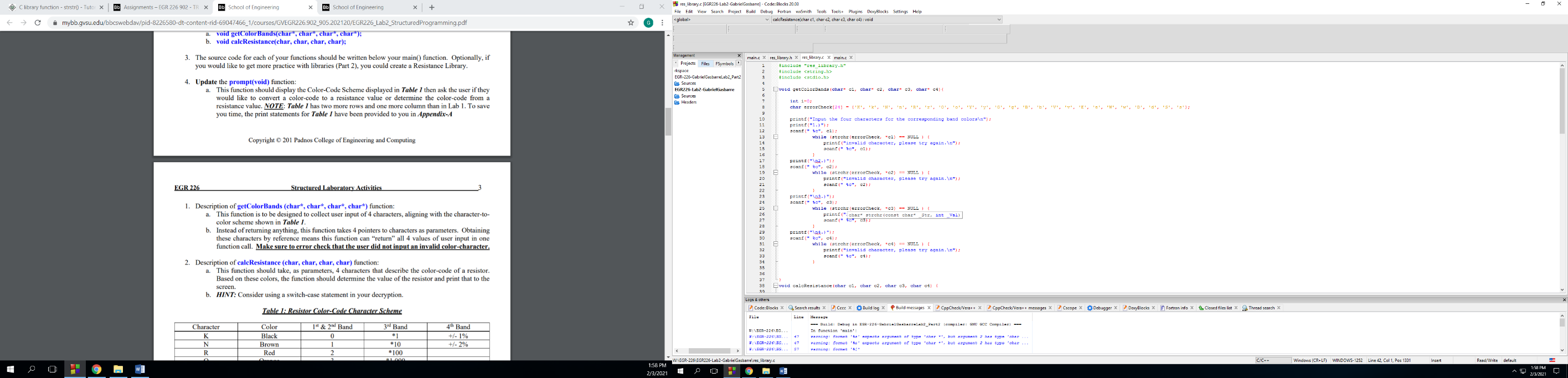
## 3.2 Part 2: Book Database

For part 2, students were asked to create a C program that uses the concepts of structures to create a database of books from an external input file. This text file will be read into the program, parsed into an array of book structures, and then made available for the user to navigate using user-input of integers.

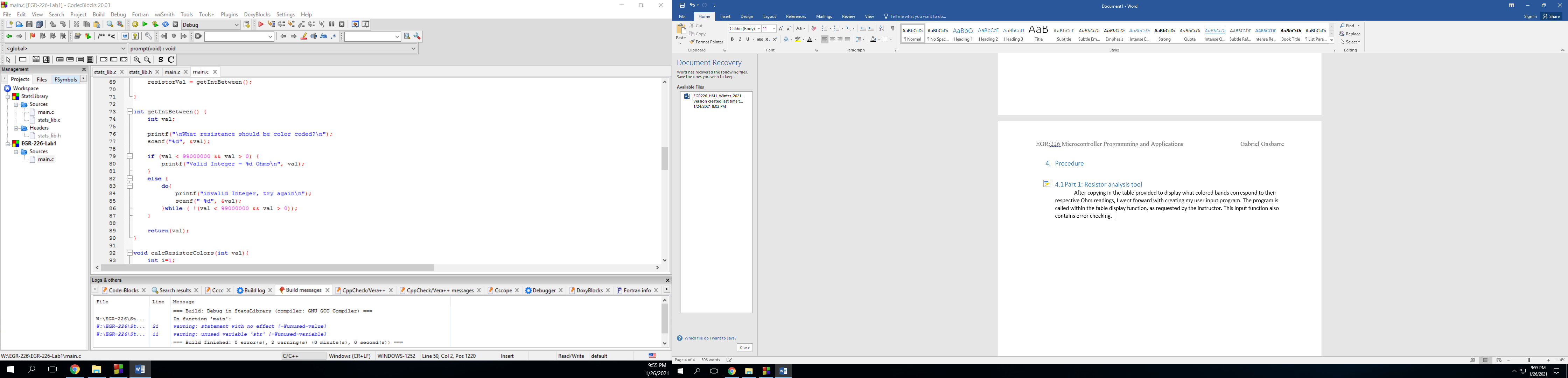
# 4. Procedure

## Part 1: Resistor analysis tool

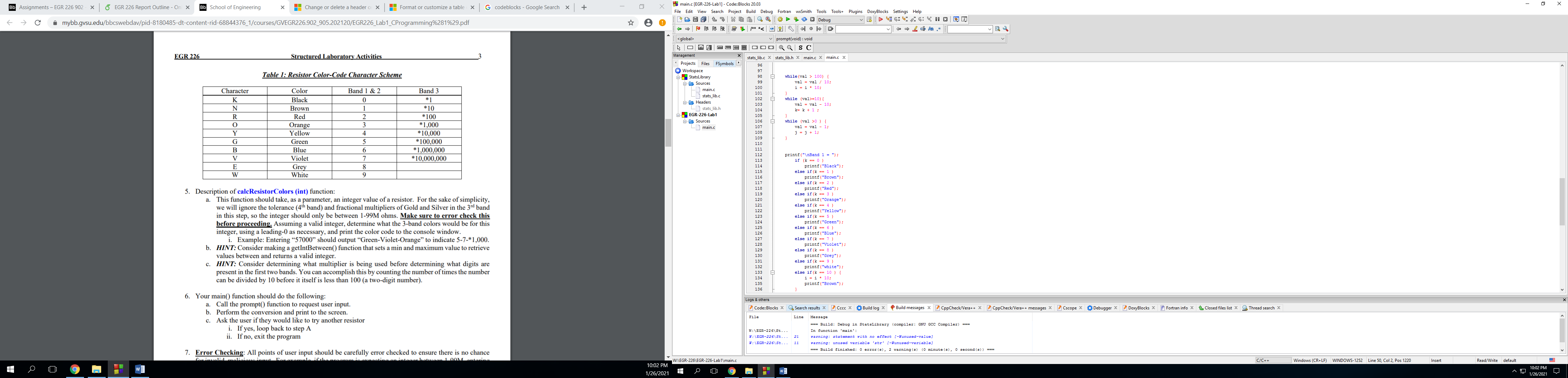
After copying in the information from part 1 and updating the prompt(void) display table, two functions were added to the program.

First, getColorBands() was added to collect the user input of 4 characters, all of which are error checked to ensure they align with the color scheme shown in Table 1 of the lab manual.

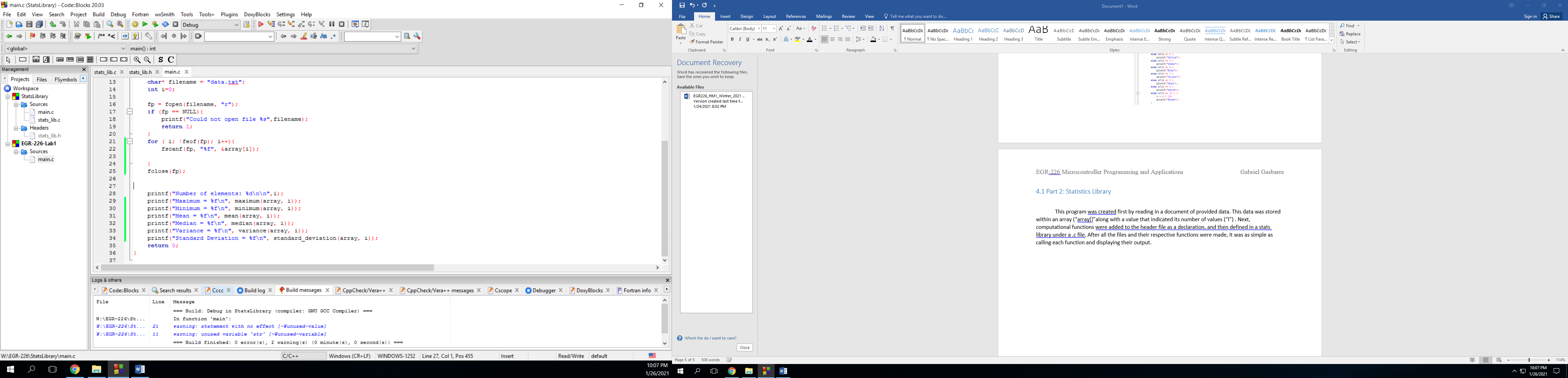
Next, calcResistance() was added to receive the 4 characters from the previous function to determine and describe the color code of a resistor that corresponds to the information entered.



Next, that input is read and saved as “resistorVal” which is then passed to the next function, which calculates the corresponding band colors. I figured out the third band multiplier first by dividing by 10 until the value was less than or equal to 100. After that, the first band was calculated by subtracting 10 until it was less than or equal to 10. Finally, the second band was calculated by subtracting 1 until the program reached 0. Each division or subtraction was counted, and saved as a number that would then be used to find the corresponding color code using a simple if-else tree.

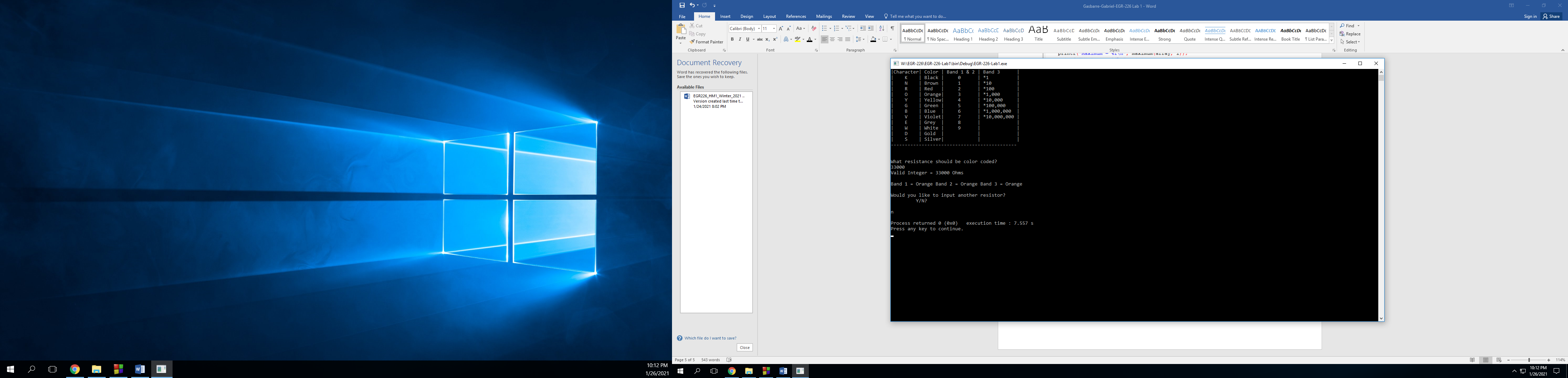


## 4.2 Part 2: Statistics Library

 This program was created first by reading in a document of provided data. This data was stored within an array (“array[]”along with a value that indicated its number of values (“I”) . Next, computational functions were added to the header file as a declaration, and then defined in a stats library under a .c file. After all the files and their respective functions were made, it was as simple as calling each function and displaying their output.

# Results

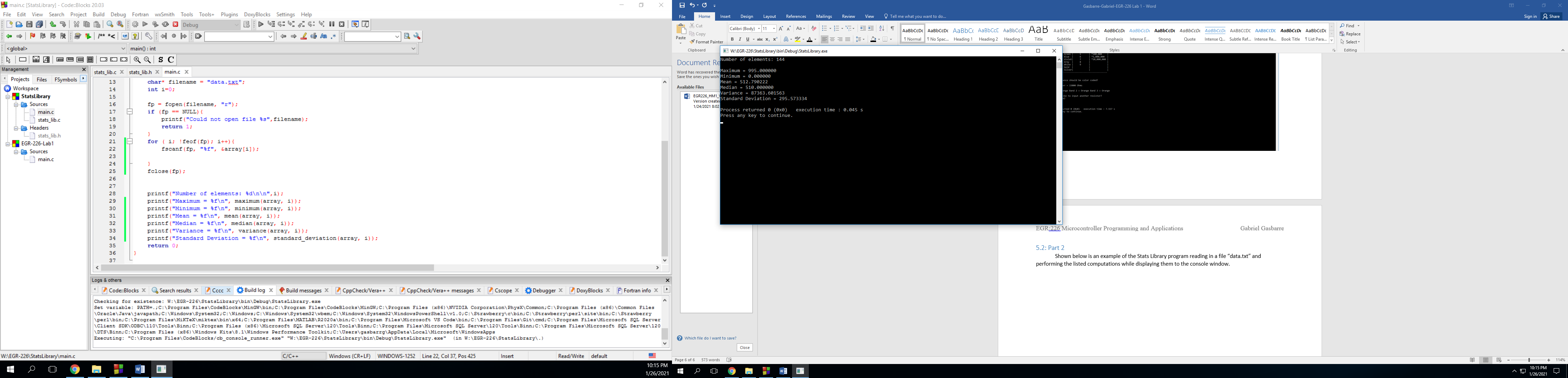
## 5.1: Part 1

Running the resistor analysis tool worked successfully and provided the desired output. Listed below is an example of how the program responds to the input of “33,000” Ohms, without another desired input.

## 

## 5.2: Part 2

Shown below is an example of the Stats Library program reading in a file “data.txt” and performing the listed computations while displaying them to the console window.



# 6 Conclusions and Future Work

I am quite satisfied with the functionality of my programs. Both seem to work flawlessly, however I would like to include more comments in the future for someone