Solving Simple Problems in C

Lab 3

Section M

Submitted by:

Haadi-Mohammad Majeed

Submission Date

September 21, 2018

Date

September 16, 2018

Lab Problem

The primary objective of lab 3 is to learn how to funnel the outputs of the controller and use them in code, along with creating basic functions utilizing inputs and returning modified values.

Analysis

The lab is parsed into 4, with 75% of them working with the creation and usage of methods, while all are utilizing the outputs of ds4rd.exe. The parts created have some synergy between each other.

Design

Part one of the lab consists of formatting the output of ds4rd executable, with the accelerations of each dimension shown with the time in seconds, with seconds having 8 characters allocated with 3 decimal places, and the accelerations having a 7 char allocation with 4 decimal places.

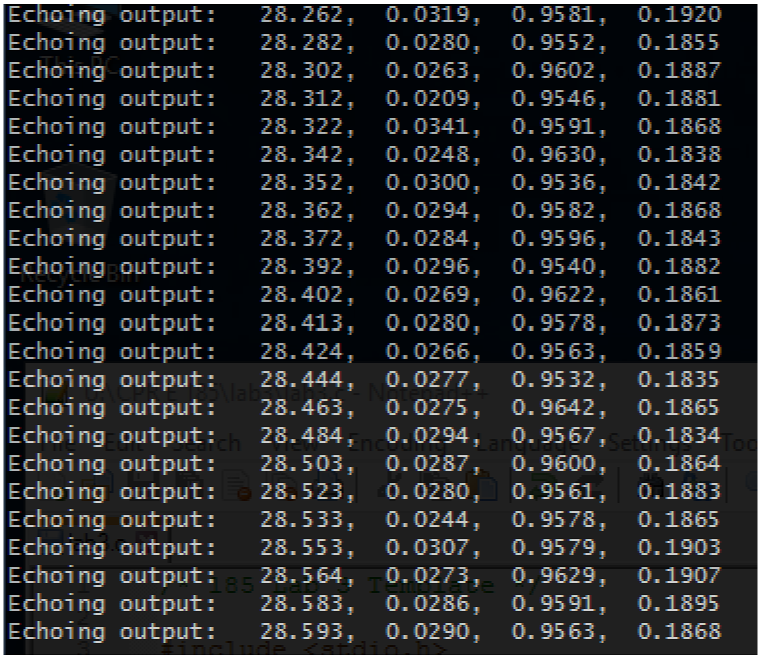
Part two requests that a function be made defined as mag. Mag is to intake the 3 accelerations and return the overall magnitude of them. It also requires the importation of the math library as to use exponents more efficiently.

Part three, upon completion, results in an additional 3 functions each which calculate the time based on the milliseconds, in minutes, seconds and milliseconds. It requires the usage for modulus and standard division to successfully achieve the correct results.

Part four constitutes the creation of a program that takes the output of ds4re.exe and throws it into a newly formed method to return how many of the 4 primary buttons are being pressed at that instant and return that into console.

Testing

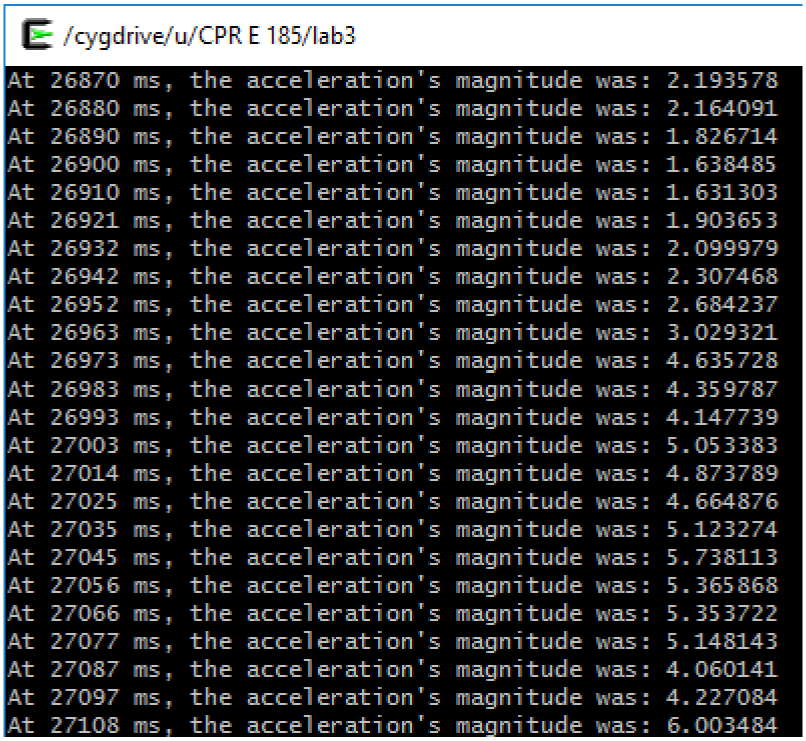
Part 1



```
Echoing output:   28.262,  0.0319,  0.9581,  0.1920
Echoing output:   28.282,  0.0280,  0.9552,  0.1855
Echoing output:   28.302,  0.0263,  0.9602,  0.1887
Echoing output:   28.312,  0.0209,  0.9546,  0.1881
Echoing output:   28.322,  0.0341,  0.9591,  0.1868
Echoing output:   28.342,  0.0248,  0.9630,  0.1838
Echoing output:   28.352,  0.0300,  0.9536,  0.1842
Echoing output:   28.362,  0.0294,  0.9582,  0.1868
Echoing output:   28.372,  0.0284,  0.9596,  0.1843
Echoing output:   28.392,  0.0296,  0.9540,  0.1882
Echoing output:   28.402,  0.0269,  0.9622,  0.1861
Echoing output:   28.413,  0.0280,  0.9578,  0.1873
Echoing output:   28.424,  0.0266,  0.9563,  0.1859
Echoing output:   28.444,  0.0277,  0.9532,  0.1835
Echoing output:   28.463,  0.0275,  0.9642,  0.1865
Echoing output:   28.484,  0.0294,  0.9567,  0.1834
Echoing output:   28.503,  0.0287,  0.9600,  0.1864
Echoing output:   28.523,  0.0280,  0.9561,  0.1883
Echoing output:   28.533,  0.0244,  0.9578,  0.1865
Echoing output:   28.553,  0.0307,  0.9579,  0.1903
Echoing output:   28.564,  0.0273,  0.9629,  0.1907
Echoing output:   28.583,  0.0286,  0.9591,  0.1895
Echoing output:   28.593,  0.0290,  0.9563,  0.1868
```

This code consisted of a printf statement that outputs seconds, and the 3 accelerations all to be formatting in such a particular fashion

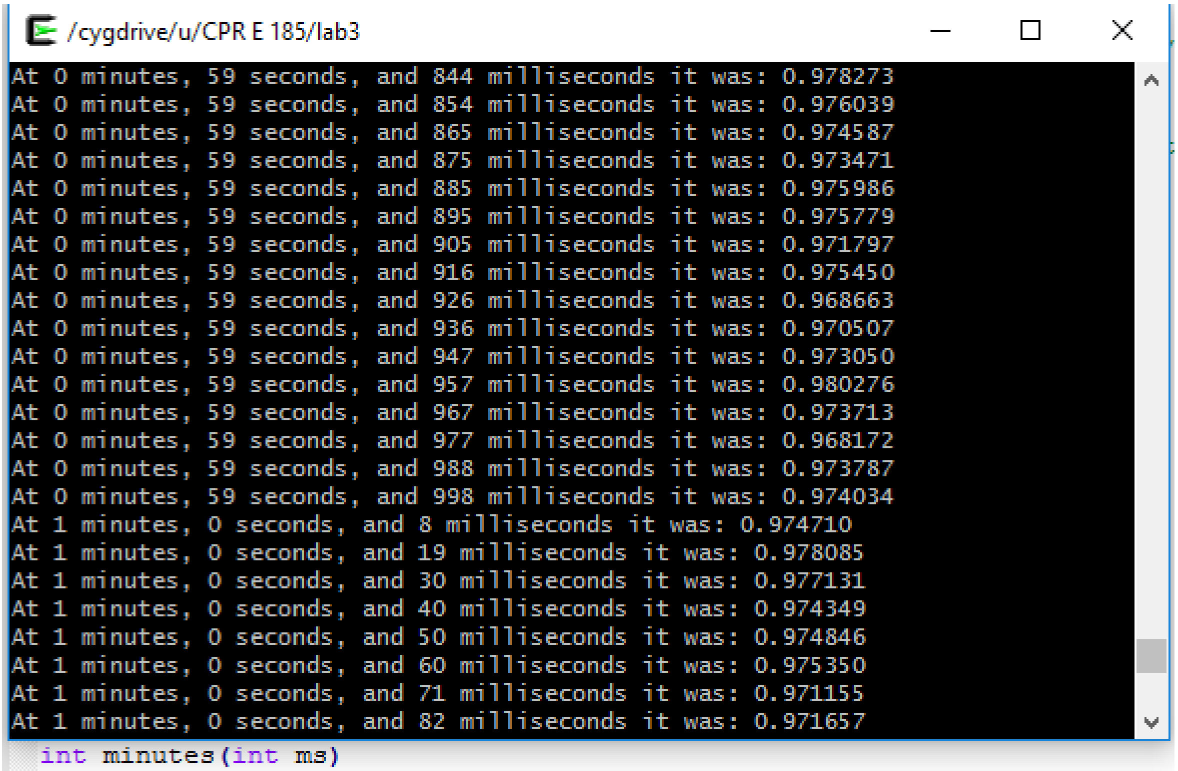
Part 2



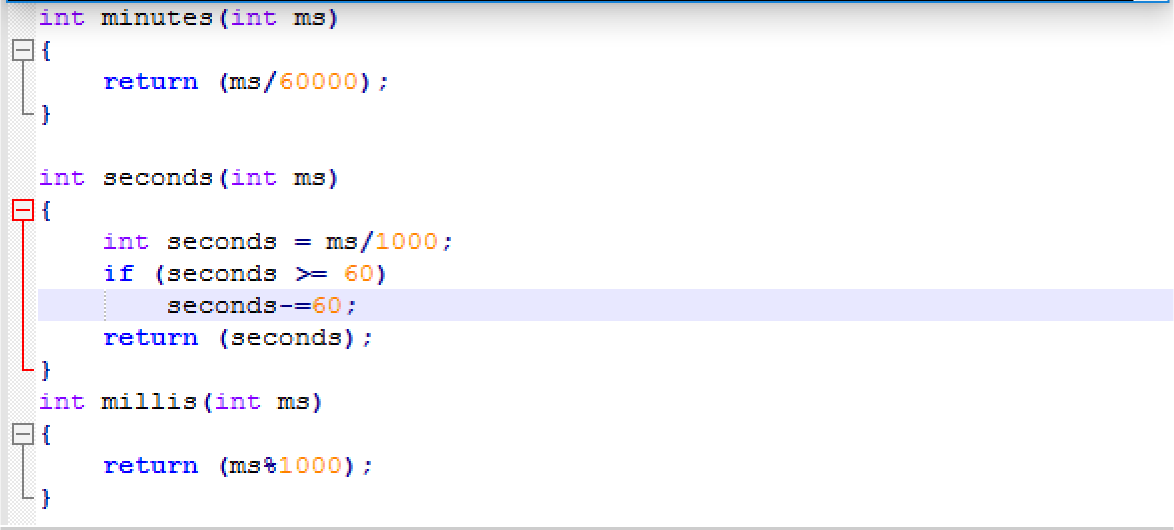
```
At 26870 ms, the acceleration's magnitude was: 2.193578
At 26880 ms, the acceleration's magnitude was: 2.164091
At 26890 ms, the acceleration's magnitude was: 1.826714
At 26900 ms, the acceleration's magnitude was: 1.638485
At 26910 ms, the acceleration's magnitude was: 1.631303
At 26921 ms, the acceleration's magnitude was: 1.903653
At 26932 ms, the acceleration's magnitude was: 2.099979
At 26942 ms, the acceleration's magnitude was: 2.307468
At 26952 ms, the acceleration's magnitude was: 2.684237
At 26963 ms, the acceleration's magnitude was: 3.029321
At 26973 ms, the acceleration's magnitude was: 4.635728
At 26983 ms, the acceleration's magnitude was: 4.359787
At 26993 ms, the acceleration's magnitude was: 4.147739
At 27003 ms, the acceleration's magnitude was: 5.053383
At 27014 ms, the acceleration's magnitude was: 4.873789
At 27025 ms, the acceleration's magnitude was: 4.664876
At 27035 ms, the acceleration's magnitude was: 5.123274
At 27045 ms, the acceleration's magnitude was: 5.738113
At 27056 ms, the acceleration's magnitude was: 5.365868
At 27066 ms, the acceleration's magnitude was: 5.353722
At 27077 ms, the acceleration's magnitude was: 5.148143
At 27087 ms, the acceleration's magnitude was: 4.060141
At 27097 ms, the acceleration's magnitude was: 4.227084
At 27108 ms, the acceleration's magnitude was: 6.003484
```

The instructions were to create a method to calculate the magnitude of the overall acceleration, thus by creating the prototype and function, it was infinitely looped.

Part 3



```
At 0 minutes, 59 seconds, and 844 milliseconds it was: 0.978273
At 0 minutes, 59 seconds, and 854 milliseconds it was: 0.976039
At 0 minutes, 59 seconds, and 865 milliseconds it was: 0.974587
At 0 minutes, 59 seconds, and 875 milliseconds it was: 0.973471
At 0 minutes, 59 seconds, and 885 milliseconds it was: 0.975986
At 0 minutes, 59 seconds, and 895 milliseconds it was: 0.975779
At 0 minutes, 59 seconds, and 905 milliseconds it was: 0.971797
At 0 minutes, 59 seconds, and 916 milliseconds it was: 0.975450
At 0 minutes, 59 seconds, and 926 milliseconds it was: 0.968663
At 0 minutes, 59 seconds, and 936 milliseconds it was: 0.970507
At 0 minutes, 59 seconds, and 947 milliseconds it was: 0.973050
At 0 minutes, 59 seconds, and 957 milliseconds it was: 0.980276
At 0 minutes, 59 seconds, and 967 milliseconds it was: 0.973713
At 0 minutes, 59 seconds, and 977 milliseconds it was: 0.968172
At 0 minutes, 59 seconds, and 988 milliseconds it was: 0.973787
At 0 minutes, 59 seconds, and 998 milliseconds it was: 0.974034
At 1 minutes, 0 seconds, and 8 milliseconds it was: 0.974710
At 1 minutes, 0 seconds, and 19 milliseconds it was: 0.978085
At 1 minutes, 0 seconds, and 30 milliseconds it was: 0.977131
At 1 minutes, 0 seconds, and 40 milliseconds it was: 0.974349
At 1 minutes, 0 seconds, and 50 milliseconds it was: 0.974846
At 1 minutes, 0 seconds, and 60 milliseconds it was: 0.975350
At 1 minutes, 0 seconds, and 71 milliseconds it was: 0.971155
At 1 minutes, 0 seconds, and 82 milliseconds it was: 0.971657
```



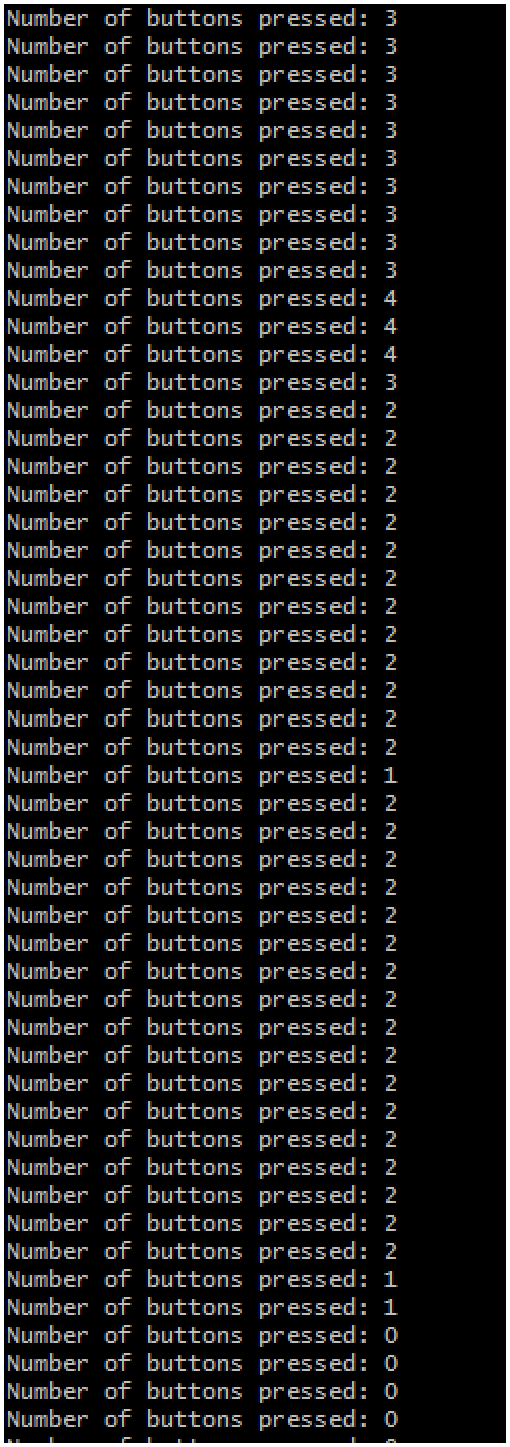
```
int minutes(int ms)
{
    return (ms/60000);
}

int seconds(int ms)
{
    int seconds = ms/1000;
    if (seconds >= 60)
        seconds-=60;
    return (seconds);
}
int millis(int ms)
{
    return (ms%1000);
}
```

Requested the creation of the methods:

- Minutes(int ms)
- Seconds(int ms)
- Millis(int ms)

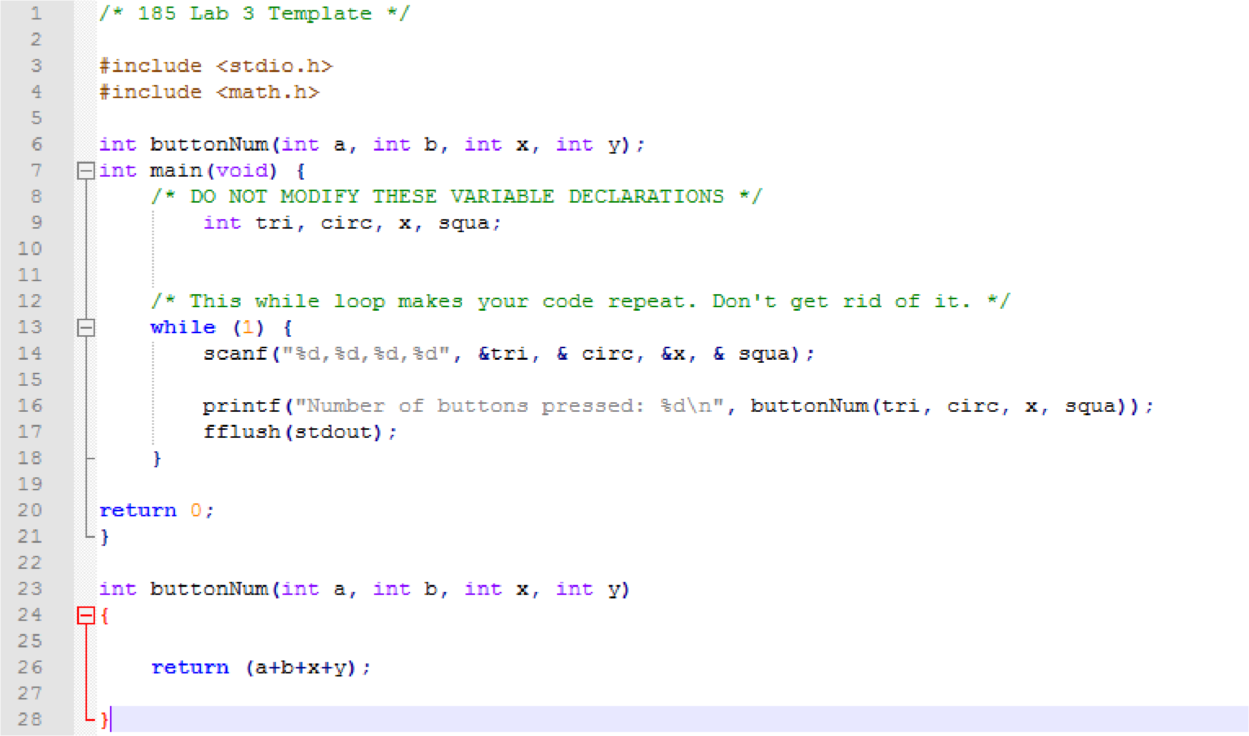
All which would return their respective amount based on the millisecond input

Part 4



```
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 4
Number of buttons pressed: 4
Number of buttons pressed: 4
Number of buttons pressed: 3
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 1
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 2
Number of buttons pressed: 1
Number of buttons pressed: 1
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
```

This part of the code used the –b part of the controller input commands to get the button binary value of on or off. Adding up the values resulted in the number of buttons currently toggled



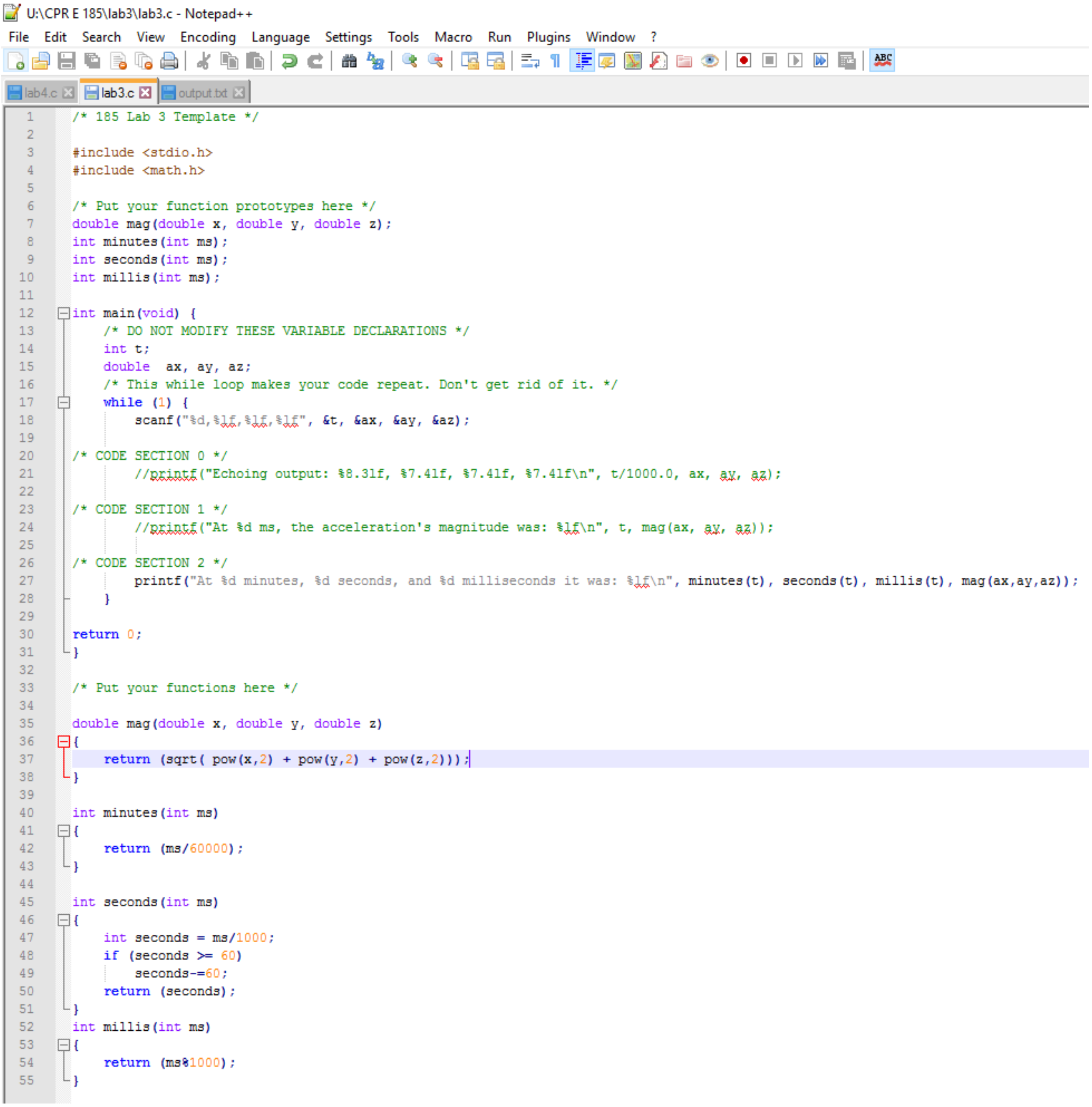
```c
/* 185 Lab 3 Template */

#include <stdio.h>
#include <math.h>

int buttonNum(int a, int b, int x, int y);
int main(void) {
    /* DO NOT MODIFY THESE VARIABLE DECLARATIONS */
        int tri, circ, x, squa;


    /* This while loop makes your code repeat. Don't get rid of it. */
    while (1) {
        scanf("%d,%d,%d,%d", &tri, & circ, &x, & squa);

        printf("Number of buttons pressed: %d\n", buttonNum(tri, circ, x, squa));
        fflush(stdout);
    }

return 0;
}

int buttonNum(int a, int b, int x, int y)
{

    return (a+b+x+y);

}
```

Comments

Prototypes are parts of code that should be in front of main unless you code the entire function above main, which may be better for code efficiency. Initially, forgetting the prototype had caused some error, but upon introducing them into the code, there was no issue.

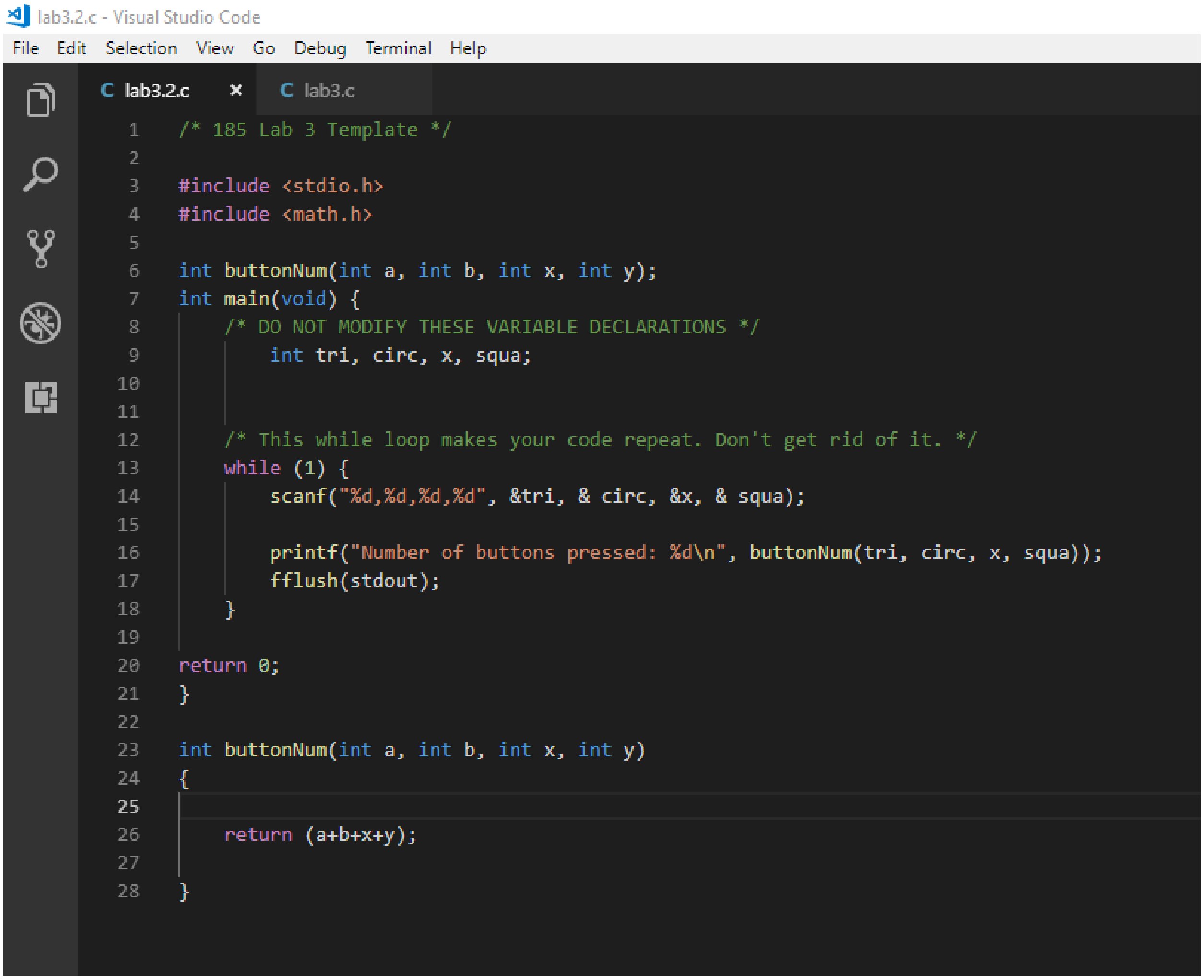
Implementation

Part 1-3

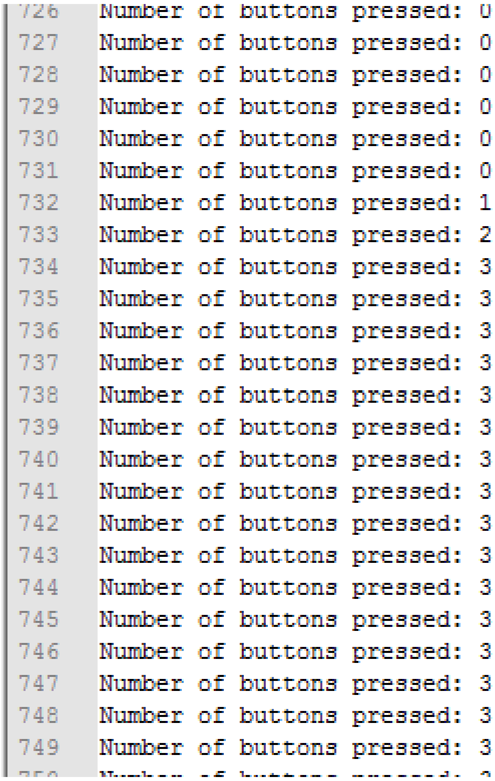


```c
/* 185 Lab 3 Template */

#include <stdio.h>
#include <math.h>

/* Put your function prototypes here */
double mag(double x, double y, double z);
int minutes(int ms);
int seconds(int ms);
int millis(int ms);

int main(void) {
    /* DO NOT MODIFY THESE VARIABLE DECLARATIONS */
    int t;
    double  ax, ay, az;
    /* This while loop makes your code repeat. Don't get rid of it. */
    while (1) {
        scanf("%d,%lf,%lf,%lf", &t, &ax, &ay, &az);

/* CODE SECTION 0 */
        //printf("Echoing output: %8.3lf, %7.4lf, %7.4lf, %7.4lf\n", t/1000.0, ax, ay, az);

/* CODE SECTION 1 */
        //printf("At %d ms, the acceleration's magnitude was: %lf\n", t, mag(ax, ay, az));

/* CODE SECTION 2 */
        printf("At %d minutes, %d seconds, and %d milliseconds it was: %lf\n", minutes(t), seconds(t), millis(t), mag(ax,ay,az));
    }

return 0;
}

/* Put your functions here */

double mag(double x, double y, double z)
{
    return (sqrt( pow(x,2) + pow(y,2) + pow(z,2)));
}

int minutes(int ms)
{
    return (ms/60000);
}

int seconds(int ms)
{
    int seconds = ms/1000;
    if (seconds >= 60)
        seconds-=60;
    return (seconds);
}
int millis(int ms)
{
    return (ms%1000);
}
```

Part 4



```
/* 185 Lab 3 Template */

#include <stdio.h>
#include <math.h>

int buttonNum(int a, int b, int x, int y);
int main(void) {
    /* DO NOT MODIFY THESE VARIABLE DECLARATIONS */
        int tri, circ, x, squa;


    /* This while loop makes your code repeat. Don't get rid of it. */
    while (1) {
        scanf("%d,%d,%d,%d", &tri, & circ, &x, & squa);

        printf("Number of buttons pressed: %d\n", buttonNum(tri, circ, x, squa));
        fflush(stdout);
    }

return 0;
}

int buttonNum(int a, int b, int x, int y)
{

    return (a+b+x+y);

}
```



```
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 0
Number of buttons pressed: 1
Number of buttons pressed: 2
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
Number of buttons pressed: 3
```