

**CPRE 1850 – INTRODUCTION TO FUNCTIONS**

**LAB 3**  
**SECTION CN**

**SUBMITTED BY:**

**MAXWELL MILLER**

**9/26/2025**

## **Problem**

This lab requires the use of organization, print formatting, and abstraction through functions, to build the most effective and readable programs. For this lab we use the Dual Shock controller in as an input device and print a variety of formatted outputs in each section. Each of these parts are designed to test our knowledge on metamathematical operators in C, along with our implementation of a variety of functions.

## **Analysis**

This lab requires use of several problem-solving techniques, particularly in integer division and functional abstraction. The lab was clearly designed to practice in these two key areas.

## **Design**

My implementation of the criteria set forth by this lab was relatively straight forward. I did decide to have functional prototypes, even though it was technically not necessary, it makes more cleaner, readable, code.

## **Testing**

Each file was run multiple times with a variety of inputs (if applicable). These test admirably arbitrary tests indicated that the programs ran as they should, but without a proper rigorous unit test it is unknown what limits these files may have. Despite this, all files ran correctly given the sample inputs from the lab 3 outline.

## **Comments**

This lab was not particularly challenging given my vast prior programming experience, but it did provide me an opportunity to remind myself of the unique attributes of the C programming language. It also allowed me to refresh my knowledge on integer division and functional prototype syntax.

## Part 1 – Code Section 0

```
/* 185 Lab 3 */

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

const int MILLIS_IN_MIN = 60000;
const int MILLIS_IN_SEC = 1000;

int main(void) {
    int t;
    double ax, ay, az;

    while (1) {
        scanf("%d,%lf,%lf,%lf", &t, &ax, &ay, &az);

        printf("Echoing output: %d, %lf, %lf, %lf\n", t, ax, ay, az);
        printf("%8.3lf, %7.4lf, %7.4lf, %7.4lf\n", t / MILLIS_IN_SEC, ax,
ay, az);
    }

    return 0;
}
```

## Part 1 – Output

```
 15.841, 0.0342, 0.9860, 0.2014
 15.857, 0.0339, 0.9873, 0.1999
 15.874, 0.0305, 0.9857, 0.2033
 15.883, 0.0275, 0.9858, 0.2019
 15.904, 0.0277, 0.9952, 0.2016
 15.920, 0.0300, 0.9850, 0.1995
 15.935, 0.0324, 0.9833, 0.2010
 15.952, 0.0329, 0.9895, 0.2025
 15.969, 0.0331, 0.9878, 0.2010
 15.984, 0.0322, 0.9889, 0.2014
 16.002, 0.0320, 0.9849, 0.2011
 16.018, 0.0279, 0.9829, 0.2030
 16.035, 0.0290, 0.9907, 0.1965
 16.054, 0.0335, 0.9875, 0.1975
 16.072, 0.0339, 0.9803, 0.2004
 16.085, 0.0324, 0.9919, 0.2048
 16.099, 0.0301, 0.9864, 0.1951
 16.115, 0.0333, 0.9965, 0.1979
 16.134, 0.0296, 0.9849, 0.2025
 16.151, 0.0342, 0.9852, 0.2020
 16.169, 0.0308, 0.9854, 0.2030
 16.187, 0.0353, 0.9836, 0.1994
 16.200, 0.0281, 0.9845, 0.1956
 16.218, 0.0277, 0.9885, 0.1968
 16.232, 0.0284, 0.9854, 0.2019
 16.248, 0.0305, 0.9814, 0.1994
 16.264, 0.0287, 0.9810, 0.1999
 16.279, 0.0314, 0.9891, 0.2010
 16.293, 0.0341, 0.9960, 0.2014
 16.307, 0.0329, 0.9894, 0.2014
 16.320, 0.0325, 0.9825, 0.2014
 16.331, 0.0325, 0.9857, 0.1951
 16.352, 0.0300, 0.9907, 0.1937
 16.368, 0.0311, 0.9868, 0.1993
 16.386, 0.0331, 0.9843, 0.2025
 16.402, 0.0344, 0.9851, 0.2017
 16.418, 0.0335, 0.9902, 0.1951
 16.434, 0.0300, 0.9864, 0.2025
 16.449, 0.0340, 0.9861, 0.2006
 16.465, 0.0317, 0.9857, 0.1987
 16.482, 0.0338, 0.9884, 0.2038
 16.491, 0.0320, 0.9899, 0.2037
 16.503, 0.0287, 0.9828, 0.2008
 16.523, 0.0301, 0.9878, 0.1964
 16.536, 0.0297, 0.9817, 0.2012
 16.556, 0.0325, 0.9863, 0.2003
 16.572, 0.0322, 0.9900, 0.1977
 16.591, 0.0319, 0.9860, 0.1958
 16.603, 0.0297, 0.9891, 0.2004
 16.620, 0.0328, 0.9858, 0.1997
 16.637, 0.0339, 0.9857, 0.2021
 16.652, 0.0316, 0.9915, 0.1992
 16.668, 0.0295, 0.9864, 0.2030
 16.684, 0.0320, 0.9817, 0.1975
 16.700, 0.0324, 0.9897, 0.2003
 16.716, 0.0339, 0.9884, 0.2011
 16.732, 0.0290, 0.9860, 0.1983
 16.748, 0.0318, 0.9812, 0.1955

mill06@CO2048-02 /cygdrive/u/CprE185/lab3
$ |
```

## Part 2 – Code Section 1

```
/* 185 Lab 3 */

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

const int MILLIS_IN_MIN = 60000;
const int MILLIS_IN_SEC = 1000;

double mag(double ax, double ay, double az);

int main(void) {
    int t;
    double ax, ay, az;

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

    return 0;
}

double mag(double ax, double ay, double az) {

    // returns the vector acceleration magnitude
    // ie. sqrt(a^2 + b^2 + c^2)
    return sqrt( ( ax * ax ) + ( ay * ay ) + ( az * az ) );
}
```

## Part 2 - Output

```
 /cygdrive/u/CprE185/lab3
At 23300 ms, the acceleration's magnitude was: 1.005000
At 23316 ms, the acceleration's magnitude was: 1.013740
At 23335 ms, the acceleration's magnitude was: 0.997123
At 23352 ms, the acceleration's magnitude was: 1.006447
At 23367 ms, the acceleration's magnitude was: 1.016544
At 23385 ms, the acceleration's magnitude was: 1.025822
At 23400 ms, the acceleration's magnitude was: 1.015448
At 23418 ms, the acceleration's magnitude was: 1.007476
At 23433 ms, the acceleration's magnitude was: 1.005732
At 23450 ms, the acceleration's magnitude was: 1.006198
At 23468 ms, the acceleration's magnitude was: 1.010334
At 23484 ms, the acceleration's magnitude was: 1.009791
At 23500 ms, the acceleration's magnitude was: 1.004626
At 23518 ms, the acceleration's magnitude was: 1.007014
At 23533 ms, the acceleration's magnitude was: 1.004829
At 23550 ms, the acceleration's magnitude was: 1.006597
At 23567 ms, the acceleration's magnitude was: 1.007370
At 23584 ms, the acceleration's magnitude was: 1.011697
At 23601 ms, the acceleration's magnitude was: 1.009695
At 23616 ms, the acceleration's magnitude was: 1.010352
At 23635 ms, the acceleration's magnitude was: 1.004084
At 23651 ms, the acceleration's magnitude was: 1.008144
At 23667 ms, the acceleration's magnitude was: 1.005073
At 23683 ms, the acceleration's magnitude was: 1.010314
At 23700 ms, the acceleration's magnitude was: 1.006773
At 23717 ms, the acceleration's magnitude was: 1.013158
At 23734 ms, the acceleration's magnitude was: 1.011478
At 23751 ms, the acceleration's magnitude was: 1.008657
At 23768 ms, the acceleration's magnitude was: 1.009390
At 23783 ms, the acceleration's magnitude was: 1.012066
At 23800 ms, the acceleration's magnitude was: 1.015277
At 23817 ms, the acceleration's magnitude was: 1.022341
At 23836 ms, the acceleration's magnitude was: 1.004096
At 23853 ms, the acceleration's magnitude was: 1.010951
At 23871 ms, the acceleration's magnitude was: 1.005551
At 23886 ms, the acceleration's magnitude was: 1.008435
At 23901 ms, the acceleration's magnitude was: 1.007244
At 23917 ms, the acceleration's magnitude was: 1.005917
At 23933 ms, the acceleration's magnitude was: 1.006408
At 23951 ms, the acceleration's magnitude was: 1.010828
At 23965 ms, the acceleration's magnitude was: 1.007580
At 23981 ms, the acceleration's magnitude was: 1.016895
At 23999 ms, the acceleration's magnitude was: 1.011699
At 24012 ms, the acceleration's magnitude was: 1.002436
At 24028 ms, the acceleration's magnitude was: 1.010962
At 24044 ms, the acceleration's magnitude was: 1.004036
At 24060 ms, the acceleration's magnitude was: 1.004090
At 24075 ms, the acceleration's magnitude was: 1.006611
At 24092 ms, the acceleration's magnitude was: 1.011124
At 24107 ms, the acceleration's magnitude was: 1.007287
At 24126 ms, the acceleration's magnitude was: 1.008854
At 24139 ms, the acceleration's magnitude was: 1.001413
At 24157 ms, the acceleration's magnitude was: 1.013796
At 24171 ms, the acceleration's magnitude was: 1.005620
At 24191 ms, the acceleration's magnitude was: 1.011359
At 24203 ms, the acceleration's magnitude was: 1.007458
At 24221 ms, the acceleration's magnitude was: 1.014246
At 24235 ms, the acceleration's magnitude was: 1.013615

mml106@C02048-02 /cygdrive/u/CprE185/lab3
$
```

## Part 3 – Code Section 2

```
/* 185 Lab 3 */

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

const int MILLIS_IN_MIN = 60000;
const int MILLIS_IN_SEC = 1000;

int minutes(int t);
int seconds(int t);
int millis(int t);

int main(void) { int t;
    double ax, ay, az;

    while (1) {
        scanf("%d,%lf,%lf,%lf", &t, &ax, &ay, &az);

        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;
}

int minutes(int t) {
    return t / MILLIS_IN_MIN;
}

int seconds(int t) {
    return t % MILLIS_IN_MIN / MILLIS_IN_SEC;
}

int millis(int t) {
    return t % MILLIS_IN_MIN % MILLIS_IN_SEC;
}
```

## Part 4 - lab3\_2.c

```
/* 185 Lab 3-2 */

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

/* Put your function prototypes here */
int numButtons(int a, int b, int c, int d);

int main(void) {
    /* DO NOT MODIFY THESE VARIABLE DECLARATIONS */
    int t;
    int a,b,c,d;

    /* This while loop makes your code repeat. Don't get rid of it. */
    while (1) {
        scanf("%d,%d,%d,%d", &a, &b, &c, &d);
        printf("Buttons pressed: %d\n", numButtons(a,b,c,d));

        fflush(stdout);
    }

    return 0;
}

/* Put your functions here */
int numButtons(int a, int b, int c, int d) {
    return (a + b + c + d);
}
```

## Output: