**Introduction to Functions**

**LAB 4 SECTION E**

**SUBMITTED BY: Joshua Bump**

**SUBMISSION DATE: 2/17/17**

**2.) REPORT PAGE**

**Problem**

In this lab there were three different problems that had to be solved. The first one that needed to be solved was coming up with a way to make the acceleration of the device more readable when we run the code. The second problem was to come up with a way to be able to display the default milliseconds as minutes, seconds and milliseconds therefore making it more familiar to the user. The last part was to figure out a way to display the number of buttons being pressed at a given time.

**Analysis**

To complete the many tasks that we were given, they had to be done using the existing code and we also needed to use functions for most of the parts of the lab. For each section we were given various information such as what to run in Cygwin once we compiled the code and also different equations that we should use. One of the equations we were given was sqrt(x^2+y^2+z^2) which helped find the magnitude of the acceleration at various times in one of the sections. We were also given fflush(stdout) when dealing with the button problem to make sure instead of adding up how many buttons we pressed, it would reset each time we press them. One of the assumptions we had to make for the button section of the lab was what to call each of the buttons being pressed as we were not given names. We also had to test the buttons to see what happens when you press them, and that helped to find out that we had to use 1 and add 1 for each button being pressed.

**Design**

On the first section, using what was given, i had to make a lot of assumptions. For example, for the mag function. We were not given much info so a lot had to thought of on our own. In order to figure out how to make the time more readable, I referred back to the textbook a lot as it had a lot about different types of functions and how to use them. In the end for this part, i used % in order to find remainders to use it for seconds and milliseconds. The minutes were the easiest, considering all I needed to do was divide by 60,000.

In the second part of the lab, with the buttons, I had to create a lot of the code on my own. For each of the buttons, I called them b1, b2, b3, b4, b5, b6 in order to make it short and simple to understand. Using the fact that when you press a button, in its column in Cygwin it says 1, I knew to use 1 in the code in each of my if statements inside the loop. From there, I just had to add scan and print functions along with fflush(stdout) which was given to us in the lab instructions.

**Testing**

When testing, I came across a lot of issues, which is why testing a lot of times is good. When i first started working on the timing part of the lab, I noticed that when I got to one minute, seconds kept going up and didn’t go to 0. Since testing showed me this, It was easy to just go back and fix the issue. When I was testing the button part of the lab, I noticed that the output would not show that I was pressing buttons. Testing it helped me figure out that I had written my if statements wrong and only put one equals sign not two, therefore making the code not work.

**Comments**

I learned a lot of things from this lab. One of the main things which I described in the testing section, was little mistakes that completely threw off the code. A lot of these mistakes I made in this lab, I will not do again because, I realized how much trouble these issues can cause. I also learned more about how the esplora works and was able to kind of form inferences on how we might use all that we are learning to create a game at the end. Since I have completed this lab, I definitely understand the esplora a lot more.

3.) IMPLEMENTATION

**Part 1 Source Code**

/\* Lab 4 Wrapper Program \*/

#include <stdio.h>

#include <math.h>

/\* Put your function prototypes here \*/

int main(void) {

int t;

double ax, ay, az;

while (1) {

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

/\* CODE SECTION 0 \*/

printf("Echoing output: %8.3d, %7.4f, %7.4f, %7.4f\n", t, 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)); \*/

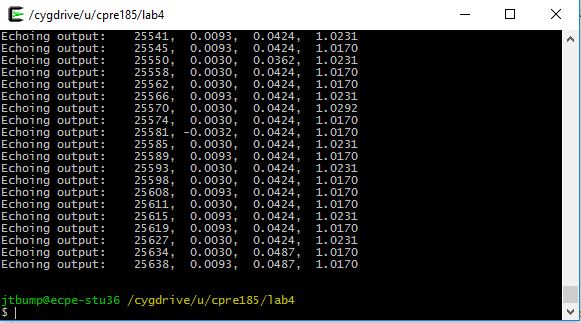
}

return 0;

}

/\* Put your functions here \*/

**Part 1 Output**



**Part 2 Source Code**

/\* Lab 4 Wrapper Program \*/

#include <stdio.h>

#include <math.h>

int 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 0 \*/

printf("Echoing output: %8.3d, %7.4f, %7.4f, %7.4f\n", t, 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)); \*/

}

return 0;

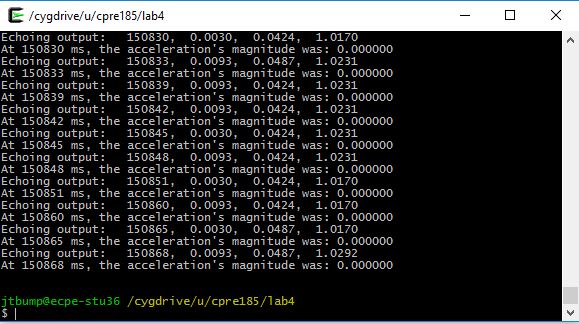
}

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

return sqrt(ax\*ax\*ay\*ay\*az\*az);

}

**Part 2 Output**



**Part 3 Source Code**

/\* Lab 4 Wrapper Program \*/

#include <stdio.h>

#include <math.h>

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

int main(void) {

int t;

double ax, ay, az;

int minutes,seconds,millis;

while (1) {

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

/\* CODE SECTION 0 \*/

printf("Echoing output: %8.3d, %7.4f, %7.4f, %7.4f\n", t, 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",(t/60000), ((t%60000)/(1000)), t % 1000, mag(ax,ay,az));

}

return 0;

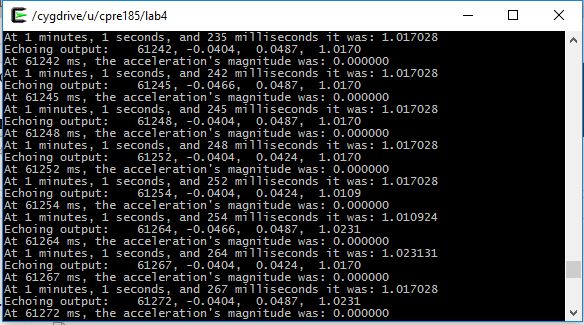
}

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

return sqrt(ax\*ax\*ay\*ay\*az\*az);

}

**Part 3 Output**



**Part 3 functions**

int minutes(int t) {

return t / 60000;

}

int seconds(int t) {

return ((t % 60000) / (1000));

}

int millis(int t) {

return t % 1000;

**Part 4 Source code**

/\* Lab 4 Wrapper Program \*/

#include <stdio.h>

#include <math.h>

int main(void) {

int b1,b2,b3,b4,b5,b6,buttons;

while (1) {

scanf("%d,%d,%d,%d,%d,%d", &b1, &b2, &b3, &b4, &b5, &b6);

buttons = 0;

if(b1 == 1){

buttons = buttons + 1;

}

if(b2 == 1){

buttons = buttons + 1;

}

if(b3 == 1){

buttons = buttons + 1;

}

if(b4 == 1){

buttons = buttons + 1;

}

if(b5 == 1){

buttons = buttons + 1;

}

printf("Number of buttons is %d\n", buttons);

fflush(stdout);

}

return 0;

}

**Part 4 Output**

