LAB 7

SECTION G

SUBMITTED BY:

RYAN SAMUELSON

SUBMISSION DATE:

11/5/15

Problem

Creating a graph, that shows the tilt in either pitch or roll starting from point 40 and going outword towards the edges. The graph was based off of the esploras degrees of tilt, the more it tilted, the more the graph showed on either side. The left side was marked with “l” and the right was marked with “r”. the amount of l’s and r’s was also determend by the tilt of the esplora. the graph also needed to be able to switch beween pitch and roll with just one button.

Analysis

The tilt from the esplora, was expressed in a number from -1 to 1. We then needed to change that into radians and then change radians into a number from -39 to 39. Taking the number from radians to and integer from -39 to 39 was the easy protion of the lab. Having to get the radians from -1 to 1 was difficult.

The graph its self was going to be hard, the fact that you can’t write to a specific place in the command line, proved this part very difficult. We needed to write a lot of “for” loops for each of the separate variables (l, r, 0).

Design

Writing the numbers conversions were somewhat simple once I found the correct equations, tangent and 180/PI.

In the graph, having to add the spaces before the r’s in concept, was easy. In reality it proved very difficult, after overthinking the problem for a while, it came to me that all I needed was a “for” loop to add the spaces that would have been taken by the l’s. For the l’s, all I needed to do was subtract the difference of l’s from the midpoint to get the correct number of spaces.

Testing

I did most of my testing in parts on a separate file. The fact that the function was spit in to multiple functions that could easily be tested made it easier to compile and debug. But near the end of the week I needed to compile the original file, I found some compatibly errors from the data of each of the functions not working together like they should be. That was the major portion that hindered my progress on the file. All of the errors dealt with pointers, and sense I am still new to the concept, I had a hard time finding the solution.

Comments

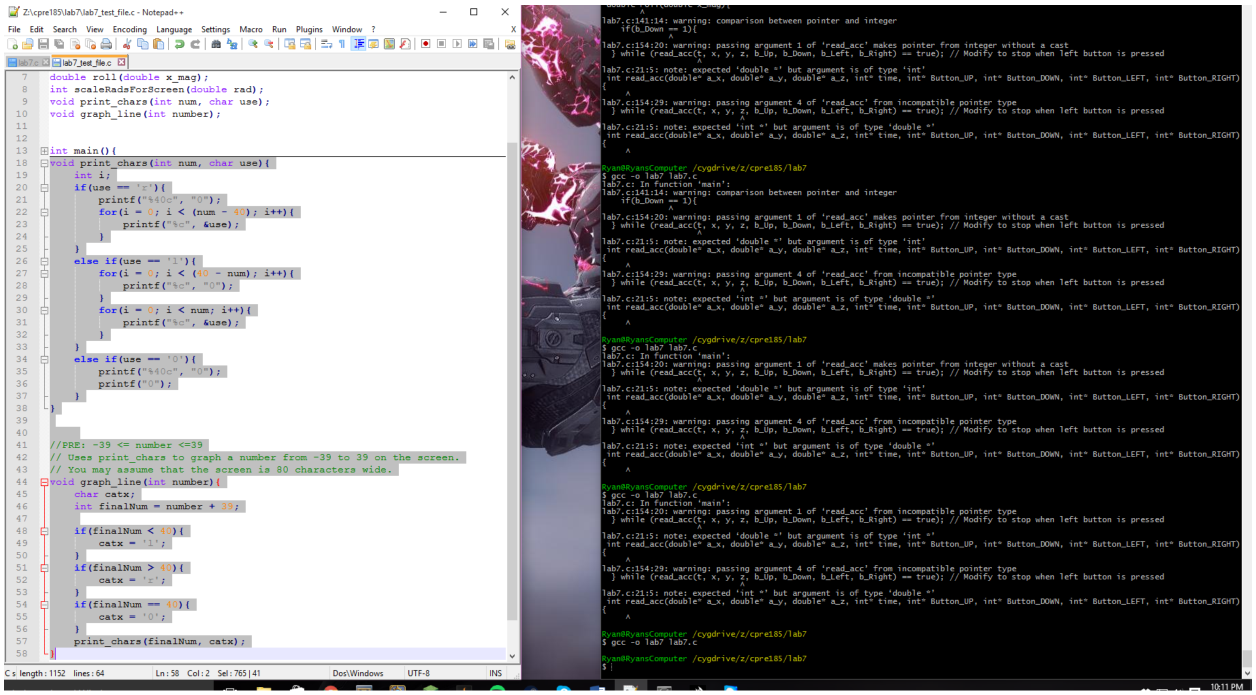
Lab7

```
// lab7.c
//
// This is the outline for your program
// Please implement the functions given by the prototypes below and
// complete the main function to make the program complete.
// You must implement the functions which are prototyped below exactly
//  as they are requested.

#include <stdio.h>
#include <math.h>
#define PI 3.141592653589

//NO GLOBAL VARIABLES ALLOWED


//PRE: Arguments must point to double variables or int variables as appropriate
//This function scans a line of explore data, and returns
//  True when left button is pressed
//  False Otherwise
//POST: it modifies its arguments to return values read from the input line.
int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
Button_LEFT, int* Button_RIGHT){
	int True;
	int False;
	if(*Button_LEFT ==  1){
		return True;
	}
	else{
		return False;
	}
}
// PRE: -1.0 <= x_mag <= 1.0
// This function computes the roll of the esplora in radians
// if x_mag outside of -1 to 1, treat it as if it were 1 or -1
// POST: -PI/2 <= return value <= PI/2
double roll(double x_mag){
	if(x_mag > 1){
		x_mag = 1;
	}
	else if (x_mag < -1){
		x_mag = -1;
	}
	double x_rad_value = tan(x_mag);
	return x_rad_value;
}
```

```
// PRE: -1.0 <= y_mag <= 1.0
// This function computes the pitch of the esplora in radians
// if y_mag outside of -1 to 1, treat it as if it were 1 or -1
// POST: -PI/2 <= return value <= PI/2
double pitch(double y_mag){
        if(y_mag > 1){
                        y_mag = 1;
                }
                else if (y_mag < -1){
                        y_mag = -1;
                }
                double y_rad_value = tan(y_mag);
                return y_rad_value;
}

// PRE: -PI/2 <= rad <= PI/2
// This function scales the roll value to fit on the screen
// POST: -39 <= return value <= 39
int scaleRadsForScreen(double rad){
        int value = rad / (PI / 2);
        int ReturedValue = value * 39;
        return ReturedValue;
}


// PRE: num >= 0
// This function prints the character use to the screen num times
// This function is the ONLY place printf is allowed to be used
// POST: nothing is returned, but use has been printed num times
void print_chars(int num, char use){
        int i;
        if(use == 'r'){
                printf("%40c", "0");
                for(i = 0; i < (num - 40); i++){
                        printf("%c", &use);
                }
        }
        else if(use == 'l'){
                for(i = 0; i < (40 - num); i++){
                        printf("%c", "0");
                }
                for(i = 0; i < num; i++){
                        printf("%c", &use);
                }
        }
        else if(use == '0'){
                printf("%40c", "0");
```

```
            printf("0");
        }
}


//PRE: -39 <= number <=39
// Uses print_chars to graph a number from -39 to 39 on the screen.
// You may assume that the screen is 80 characters wide.
void graph_line(int number){
        char catx;
        int finalNum = number + 39;

        if(finalNum < 40){
                catx = 'l';
        }
        if(finalNum > 40){
                catx = 'r';
        }
        if(finalNum == 40){
                catx = '0';
        }
        print_chars(finalNum, catx);
}

int main(){
        double *x, *y, *z;                              // magnitude values of x, y, and z accelerations
        int *b_Up, *b_Down, *b_Left, *b_Right;// variables to hold the button statuses
        double roll_rad, pitch_rad;                     // value of the roll measured in radians
        int scaled_value;               // value of the roll adjusted to fit screen display

        //insert any beginning needed code here
        int true;
        int *t;


        do
        {
                // Get line of input
                scanf("%d, %lf, %lf, %lf, %d, %d, %d, %d", t, x, y, z, b_Up, b_Down, b_Left, b_Right);

                        // calculate roll and pitch
                        pitch_rad = pitch(*y);
                        roll_rad = roll(*x);

                        // switch between roll and pitch(up vs. down button)
                        double *final = &roll_rad;
                        double *swap1 = &pitch_rad;
                        if(*b_Down == 1){
```

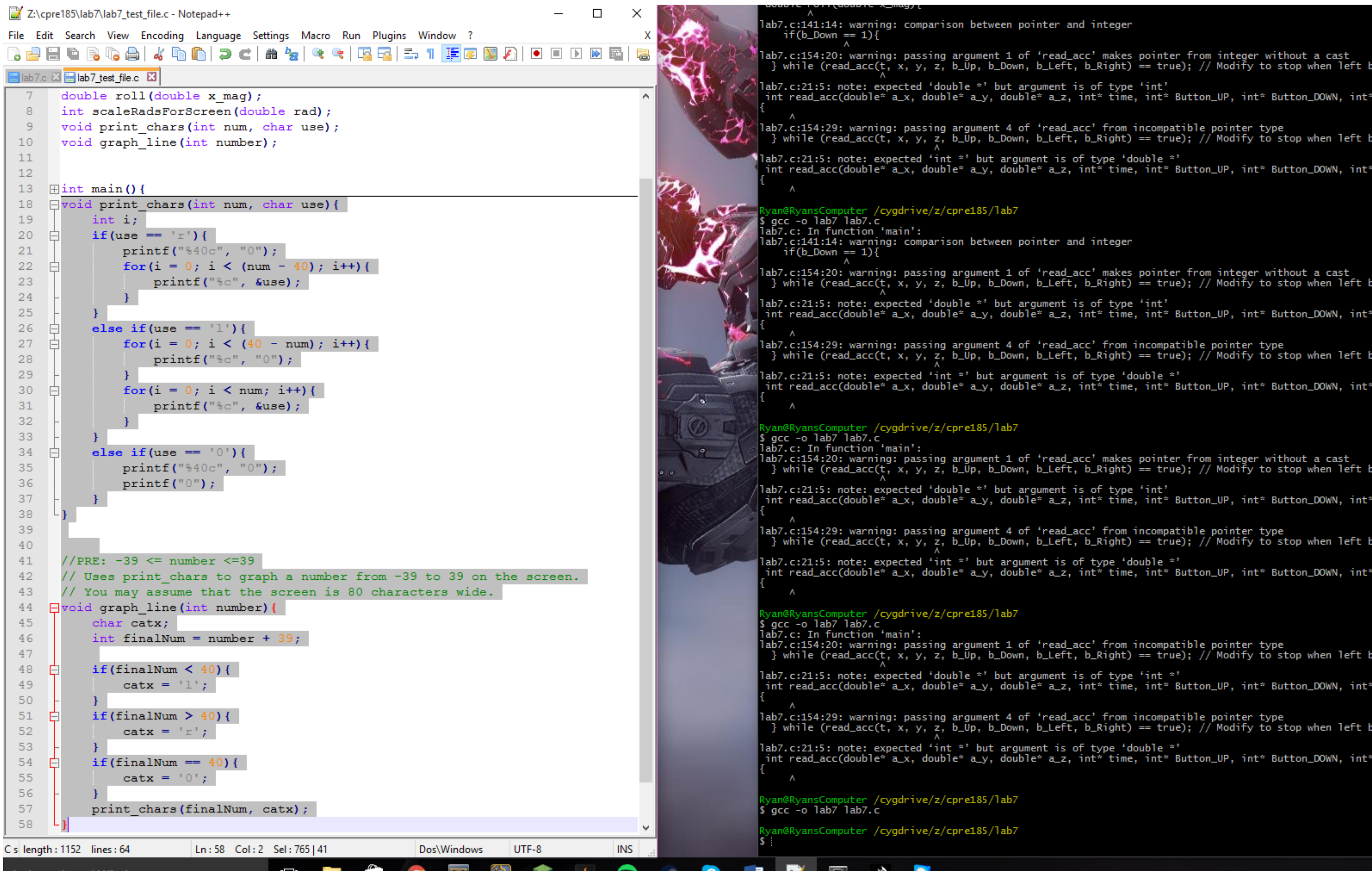
```
                        double temp = *final;
                        *final = *swap1;
                        *swap1 = temp;
                    }

                    // Scale your output value
                    int scaled_value = scaleRadsForScreen(*final);

                    // Output your graph line
                    graph_line(scaled_value);

                fflush(stdout);
    } while (read_acc(x, y, z, t, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left
button is pressed
    return 0;
}
```


Z:\cpre185\lab7\lab7_test_file.c - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
lab7.c
lab7_test_file.c
C s length : 1152 lines : 64 Ln : 58 Col : 2 Sel : 765 | 41 Dos\Windows UTF-8 INS
10:11 PM

```c
double roll(double x_mag);
int scaleRadsForScreen(double rad);
void print_chars(int num, char use);
void graph_line(int number);


int main(){
void print_chars(int num, char use){
    int i;
    if(use == 'r'){
        printf("%40c", "0");
        for(i = 0; i < (num - 40); i++){
            printf("%c", &use);
        }
    }
    else if(use == 'l'){
        for(i = 0; i < (40 - num); i++){
            printf("%c", "0");
        }
        for(i = 0; i < num; i++){
            printf("%c", &use);
        }
    }
    else if(use == '0'){
        printf("%40c", "0");
        printf("0");
    }
}


//PRE: -39 <= number <=39
// Uses print_chars to graph a number from -39 to 39 on the screen.
// You may assume that the screen is 80 characters wide.
void graph_line(int number){
    char catx;
    int finalNum = number + 39;

    if(finalNum < 40){
        catx = 'l';
    }
    if(finalNum > 40){
        catx = 'r';
    }
    if(finalNum == 40){
        catx = '0';
    }
    print_chars(finalNum, catx);
}
```

```
 double roll(double x_mag){
        ^
lab7.c:141:14: warning: comparison between pointer and integer
     if(b_Down == 1){
              ^
lab7.c:154:20: warning: passing argument 1 of ‘read_acc’ makes pointer from integer without a cast
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                    ^
lab7.c:21:5: note: expected ‘double *’ but argument is of type ‘int’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^
lab7.c:154:29: warning: passing argument 4 of ‘read_acc’ from incompatible pointer type
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                             ^
lab7.c:21:5: note: expected ‘int *’ but argument is of type ‘double *’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function ‘main’:
lab7.c:141:14: warning: comparison between pointer and integer
     if(b_Down == 1){
              ^
lab7.c:154:20: warning: passing argument 1 of ‘read_acc’ makes pointer from integer without a cast
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                    ^
lab7.c:21:5: note: expected ‘double *’ but argument is of type ‘int’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^
lab7.c:154:29: warning: passing argument 4 of ‘read_acc’ from incompatible pointer type
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                             ^
lab7.c:21:5: note: expected ‘int *’ but argument is of type ‘double *’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function ‘main’:
lab7.c:154:20: warning: passing argument 1 of ‘read_acc’ makes pointer from integer without a cast
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                    ^
lab7.c:21:5: note: expected ‘double *’ but argument is of type ‘int’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^
lab7.c:154:29: warning: passing argument 4 of ‘read_acc’ from incompatible pointer type
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                             ^
lab7.c:21:5: note: expected ‘int *’ but argument is of type ‘double *’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function ‘main’:
lab7.c:154:20: warning: passing argument 1 of ‘read_acc’ from incompatible pointer type
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                    ^
lab7.c:21:5: note: expected ‘double *’ but argument is of type ‘int *’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^
lab7.c:154:29: warning: passing argument 4 of ‘read_acc’ from incompatible pointer type
   } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left button is pressed
                             ^
lab7.c:21:5: note: expected ‘int *’ but argument is of type ‘double *’
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int* Button_LEFT, int* Button_RIGHT)
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$
```

If you need a bigger picture, there is a bigger one below.



Z:\cpre185\lab7\lab7_test_file.c - Notepad++

```c
double roll(double x_mag);
int scaleRadsForScreen(double rad);
void print_chars(int num, char use);
void graph_line(int number);


int main(){
void print_chars(int num, char use){
    int i;
    if(use == 'r'){
        printf("%40c", "0");
        for(i = 0; i < (num - 40); i++){
            printf("%c", &use);
        }
    }
    else if(use == 'l'){
        for(i = 0; i < (40 - num); i++){
            printf("%c", "0");
        }
        for(i = 0; i < num; i++){
            printf("%c", &use);
        }
    }
    else if(use == '0'){
        printf("%40c", "0");
        printf("0");
    }
}


//PRE: -39 <= number <=39
// Uses print_chars to graph a number from -39 to 39 on the screen.
// You may assume that the screen is 80 characters wide.
void graph_line(int number){
    char catx;
    int finalNum = number + 39;

    if(finalNum < 40){
        catx = 'l';
    }
    if(finalNum > 40){
        catx = 'r';
    }
    if(finalNum == 40){
        catx = '0';
    }
    print_chars(finalNum, catx);
}
```

C s length : 1152   lines : 64   Ln : 58   Col : 2   Sel : 765 | 41   Dos\Windows   UTF-8   INS

```
double roll(double x_mag){
       ^
lab7.c:141:14: warning: comparison between pointer and integer
     if(b_Down == 1){
              ^
lab7.c:154:20: warning: passing argument 1 of 'read_acc' makes pointer from integer without a cast
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                    ^
lab7.c:21:5: note: expected 'double *' but argument is of type 'int'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^
lab7.c:154:29: warning: passing argument 4 of 'read_acc' from incompatible pointer type
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                             ^
lab7.c:21:5: note: expected 'int *' but argument is of type 'double *'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function 'main':
lab7.c:141:14: warning: comparison between pointer and integer
     if(b_Down == 1){
              ^
lab7.c:154:20: warning: passing argument 1 of 'read_acc' makes pointer from integer without a cast
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                    ^
lab7.c:21:5: note: expected 'double *' but argument is of type 'int'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^
lab7.c:154:29: warning: passing argument 4 of 'read_acc' from incompatible pointer type
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                             ^
lab7.c:21:5: note: expected 'int *' but argument is of type 'double *'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function 'main':
lab7.c:154:20: warning: passing argument 1 of 'read_acc' makes pointer from integer without a cast
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                    ^
lab7.c:21:5: note: expected 'double *' but argument is of type 'int'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^
lab7.c:154:29: warning: passing argument 4 of 'read_acc' from incompatible pointer type
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                             ^
lab7.c:21:5: note: expected 'int *' but argument is of type 'double *'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c
lab7.c: In function 'main':
lab7.c:154:20: warning: passing argument 1 of 'read_acc' from incompatible pointer type
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                    ^
lab7.c:21:5: note: expected 'double *' but argument is of type 'int *'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^
lab7.c:154:29: warning: passing argument 4 of 'read_acc' from incompatible pointer type
  } while (read_acc(t, x, y, z, b_Up, b_Down, b_Left, b_Right) == true); // Modify to stop when left b
                             ^
lab7.c:21:5: note: expected 'int *' but argument is of type 'double *'
 int read_acc(double* a_x, double* a_y, double* a_z, int* time, int* Button_UP, int* Button_DOWN, int*
{
     ^

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$ gcc -o lab7 lab7.c

Ryan@RyansComputer /cygdrive/z/cpre185/lab7
$
```