The DS4 Equalizer

LAB 07

SECTION 5

ARYAN RAO

DATE- 11/2/2021

Problem

Use the DualShock 4 and determine using its roll and pitch whether it’s being tipped left or right.

Analysis

The program should start by displaying graph roll. To display roll, user should press triangle button, while the X button is to be pushed for pitch.

Design

- Scan inputs from DS4
- Collect gyroscope values
- Print the character respectively
- Collect joystick data
- Associate that with circle button

Testing

Calculate the roll and pitch values manually and compare them with the values we get on the program.

Comments

Observing patterns and readings enables you to obtain good range of value for each movement. This is crucial for an accurate and a good working program.

Questions

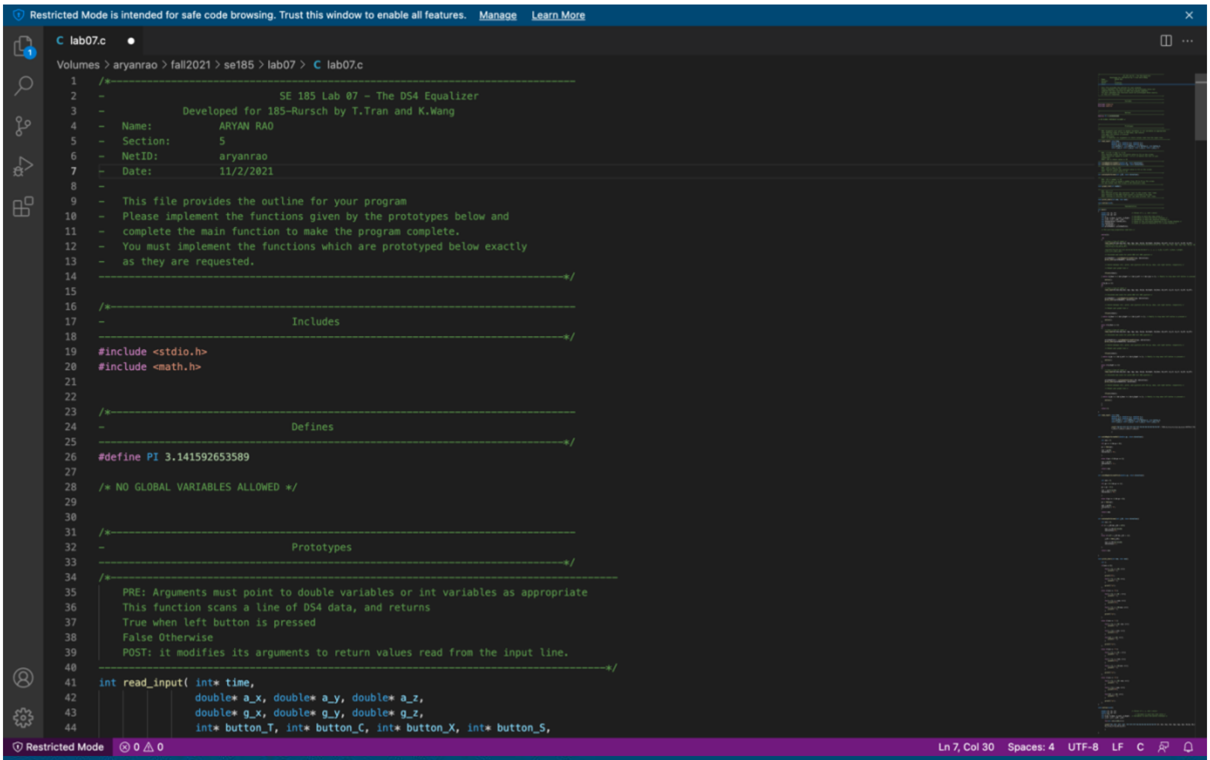
1. How did you scale your values? Write an equation to justify it.

   ```
   if (gz > 0.2 && gz <= 1)
   gz=gz*0.2

   num = (gz/0.8)*40;

   *direction = ‘B’;
   ```

2. As your experiment with the roll and pitch, what do you notice about the graph's behavior near the limits of its values?

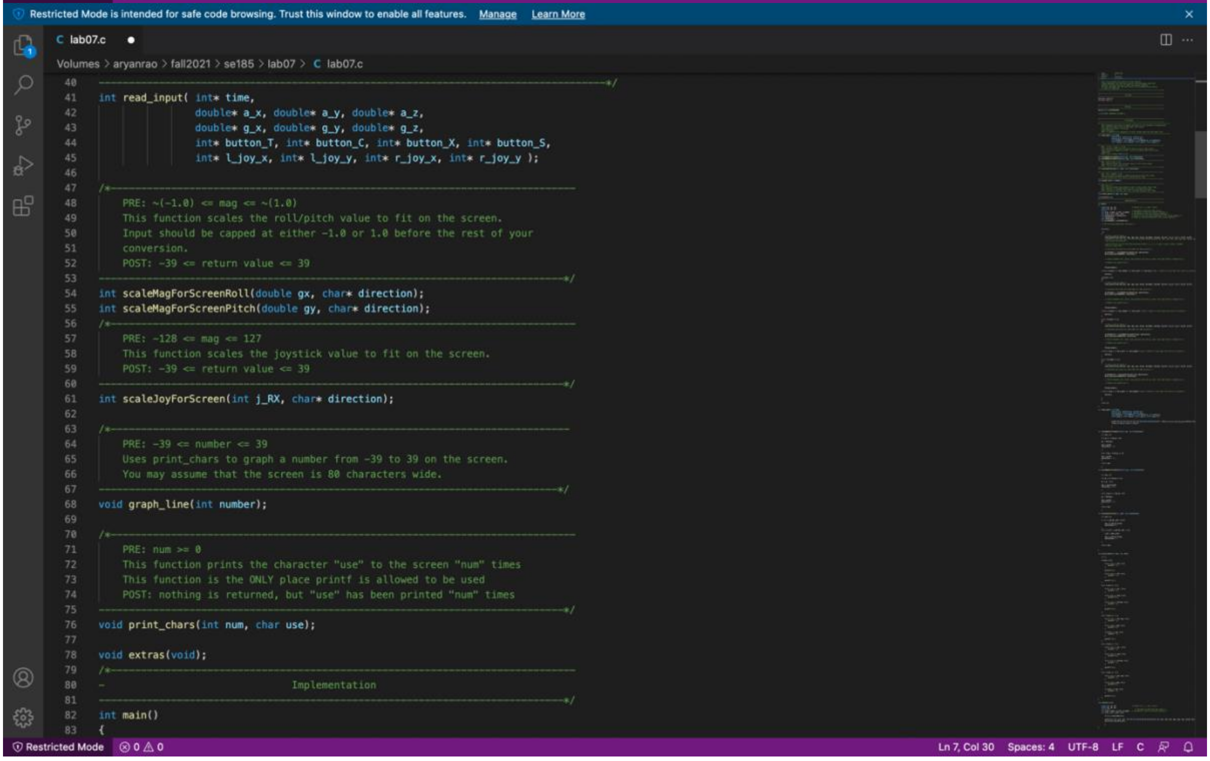
The graph becomes more stable and constant near the limits of its value.

Source Code



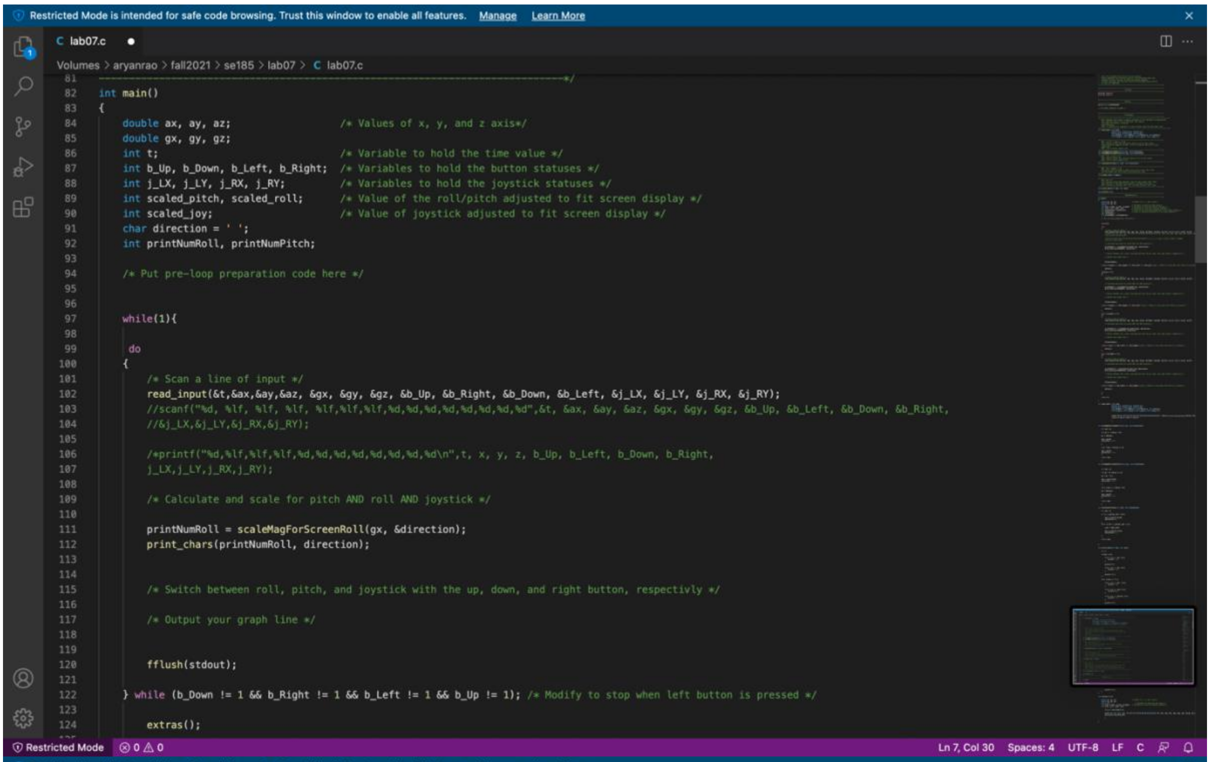
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

```c
/*----------------------------------------------------------------------------
-                       SE 185 Lab 07 - The DS4 Equalizer
-              Developed for 185-Rursch by T.Tran and K.Wang
-   Name:           ARYAN RAO
-   Section:        5
-   NetID:          aryanrao
-   Date:           11/2/2021
-
-   This file provides 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.
----------------------------------------------------------------------------*/

/*----------------------------------------------------------------------------
-                               Includes
----------------------------------------------------------------------------*/
#include <stdio.h>
#include <math.h>


/*----------------------------------------------------------------------------
-                               Defines
----------------------------------------------------------------------------*/
#define PI 3.141592653589

/* NO GLOBAL VARIABLES ALLOWED */


/*----------------------------------------------------------------------------
-                               Prototypes
----------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------------
    PRE: Arguments must point to double variables or int variables as appropriate
    This function scans a line of DS4 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_input( int* time,
                double* a_x, double* a_y, double* a_z,
                double* g_x, double* g_y, double* g_z,
                int* button_T, int* button_C, int* button_X, int* button_S,
```

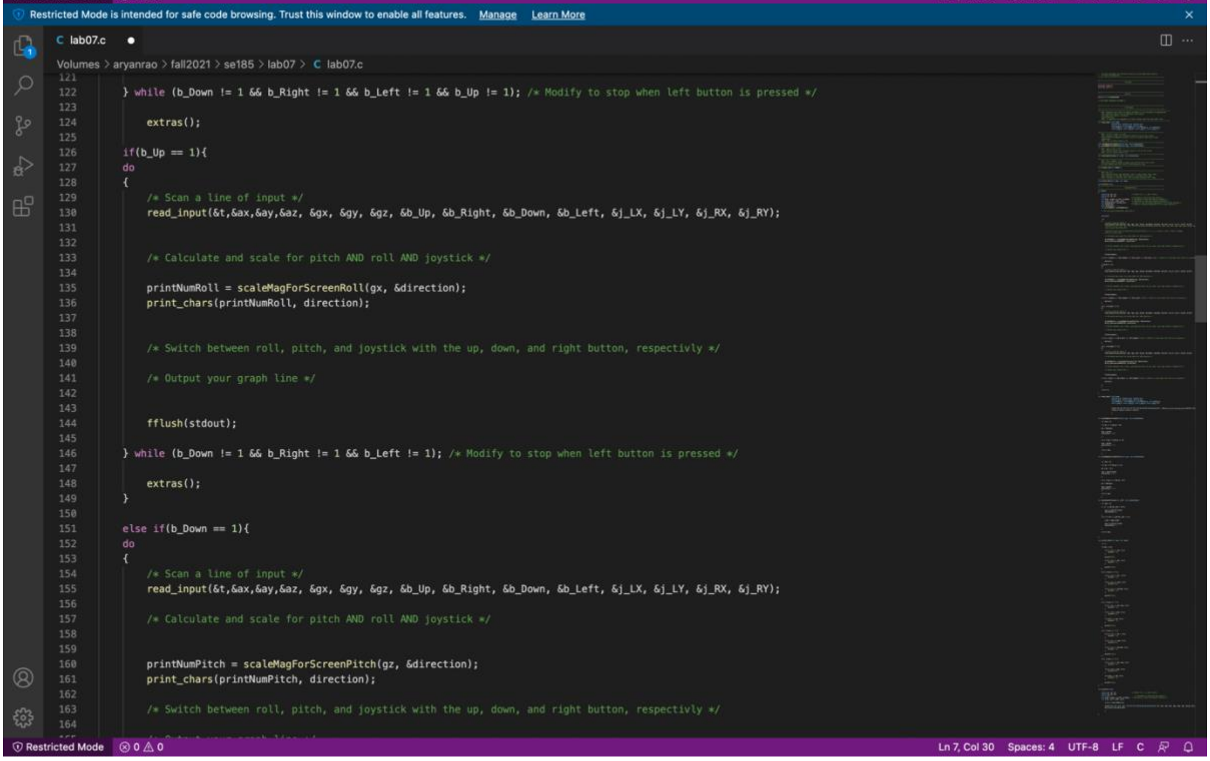


Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

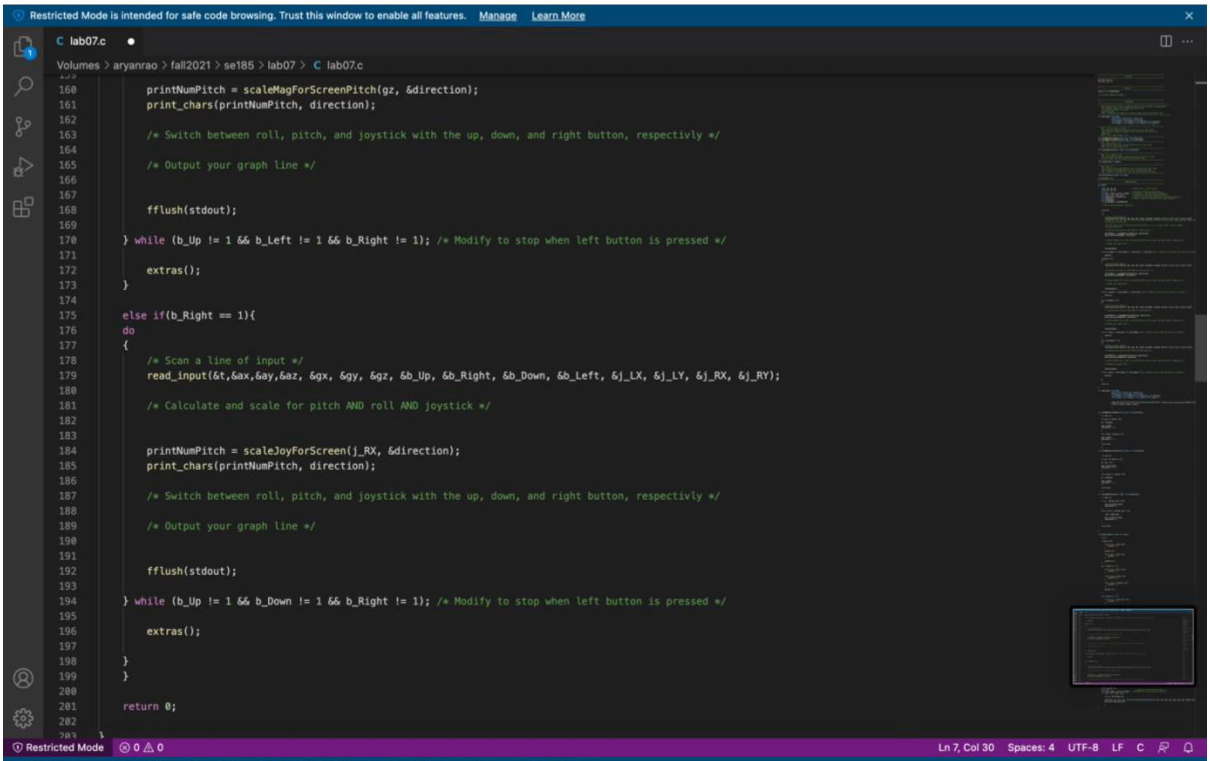
```c
-------------------------------------------------------------------------------*/
int read_input( int* time,
                double* a_x, double* a_y, double* a_z,
                double* g_x, double* g_y, double* g_z,
                int* button_T, int* button_C, int* button_X, int* button_S,
                int* l_joy_x, int* l_joy_y, int* r_joy_x, int* r_joy_y );

/*----------------------------------------------------------------------------
    PRE: ~(-1.0) <= mag <= ~(1.0)
    This function scales the roll/pitch value to fit on the screen.
    Input should be capped at either -1.0 or 1.0 before the rest of your
    conversion.
    POST: -39 <= return value <= 39
----------------------------------------------------------------------------*/
int scaleMagForScreenRoll(double gx, char* direction);
int scaleMagForScreenPitch(double gy, char* direction);
/*----------------------------------------------------------------------------
    PRE: -128 <= mag <= 127
    This function scales the joystick value to fit on the screen.
    POST: -39 <= return value <= 39
----------------------------------------------------------------------------*/
int scaleJoyForScreen(int j_RX, char* direction);

/*---------------------------------------------------------------------------
    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);

/*----------------------------------------------------------------------------
    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);

void extras(void);
/*----------------------------------------------------------------------------
-                               Implementation
----------------------------------------------------------------------------*/
int main()
{
```


Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > lab07.c
Restricted Mode
0 0
Ln 7, Col 30 Spaces: 4 UTF-8 LF C

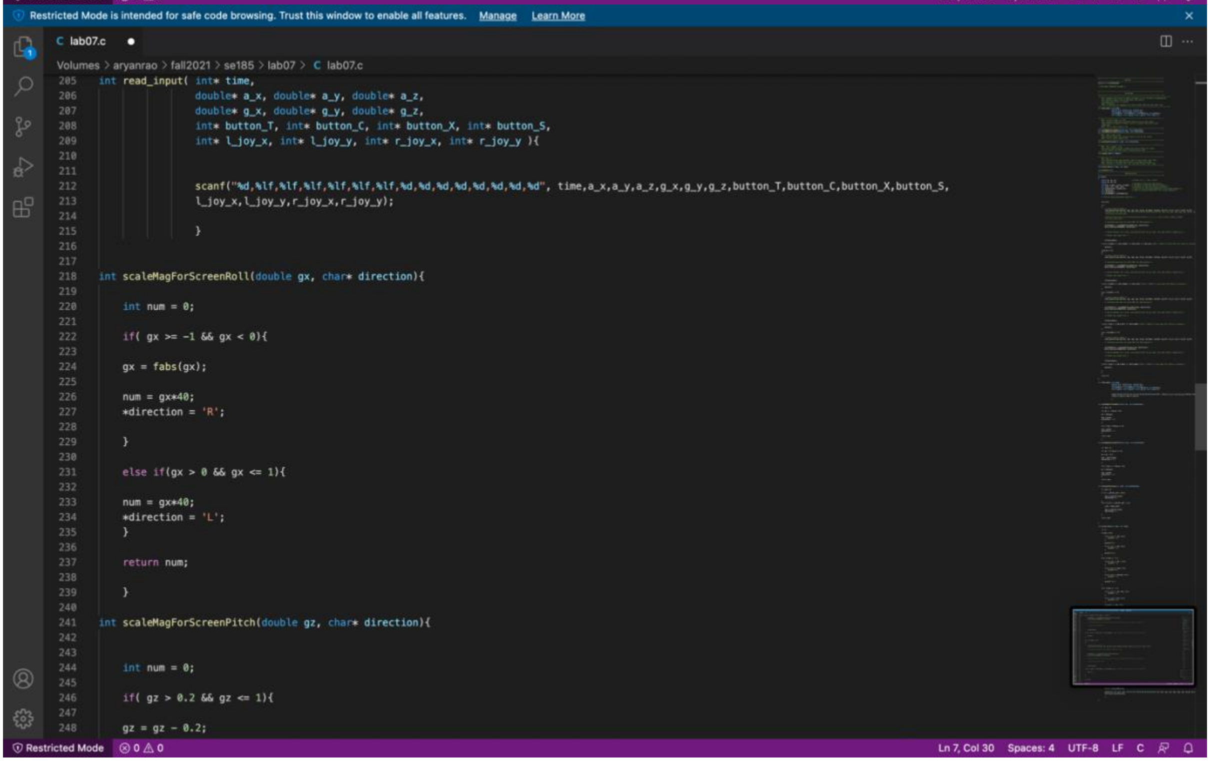
```c
-----------------------------------------------------------------------------*/
int main()
{
    double ax, ay, az;                 /* Values of x, y, and z axis*/
    double gx, gy, gz;
    int t;                             /* Variable to hold the time value */
    int b_Up, b_Down, b_Left, b_Right; /* Variables to hold the button statuses */
    int j_LX, j_LY, j_RX, j_RY;        /* Variables to hold the joystick statuses */
    int scaled_pitch, scaled_roll;     /* Value of the roll/pitch adjusted to fit screen display */
    int scaled_joy;                    /* Value of joystick adjusted to fit screen display */
    char direction = ' ';
    int printNumRoll, printNumPitch;

    /* Put pre-loop preparation code here */


    while(1){

     do
    {
        /* Scan a line of input */
        read_input(&t,&ax,&ay,&az, &gx, &gy, &gz, &b_Up, &b_Right, &b_Down, &b_Left, &j_LX, &j_LY, &j_RX, &j_RY);
        //scanf("%d, %lf, %lf, %lf, %lf,%lf,%lf,%d,%d,%d,%d,%d,%d,%d,%d",&t, &ax, &ay, &az, &gx, &gy, &gz, &b_Up, &b_Left, &b_Down, &b_Right,
        //&j_LX,&j_LY,&j_RX,&j_RY);

        /*printf("%d,%lf,%lf,%lf,%d,%d,%d,%d,%d,%d,%d,%d\n",t, x, y, z, b_Up, b_Left, b_Down, b_Right,
        j_LX,j_LY,j_RX,j_RY);

        /* Calculate and scale for pitch AND roll AND joystick */

        printNumRoll = scaleMagForScreenRoll(gx, &direction);
        print_chars(printNumRoll, direction);


        /* Switch between roll, pitch, and joystick with the up, down, and right button, respectivly */

        /* Output your graph line */


        fflush(stdout);

    } while (b_Down != 1 && b_Right != 1 && b_Left != 1 && b_Up != 1); /* Modify to stop when left button is pressed */

        extras();
```


Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > lab07.c
Restricted Mode
0 0
Ln 7, Col 30 Spaces: 4 UTF-8 LF C

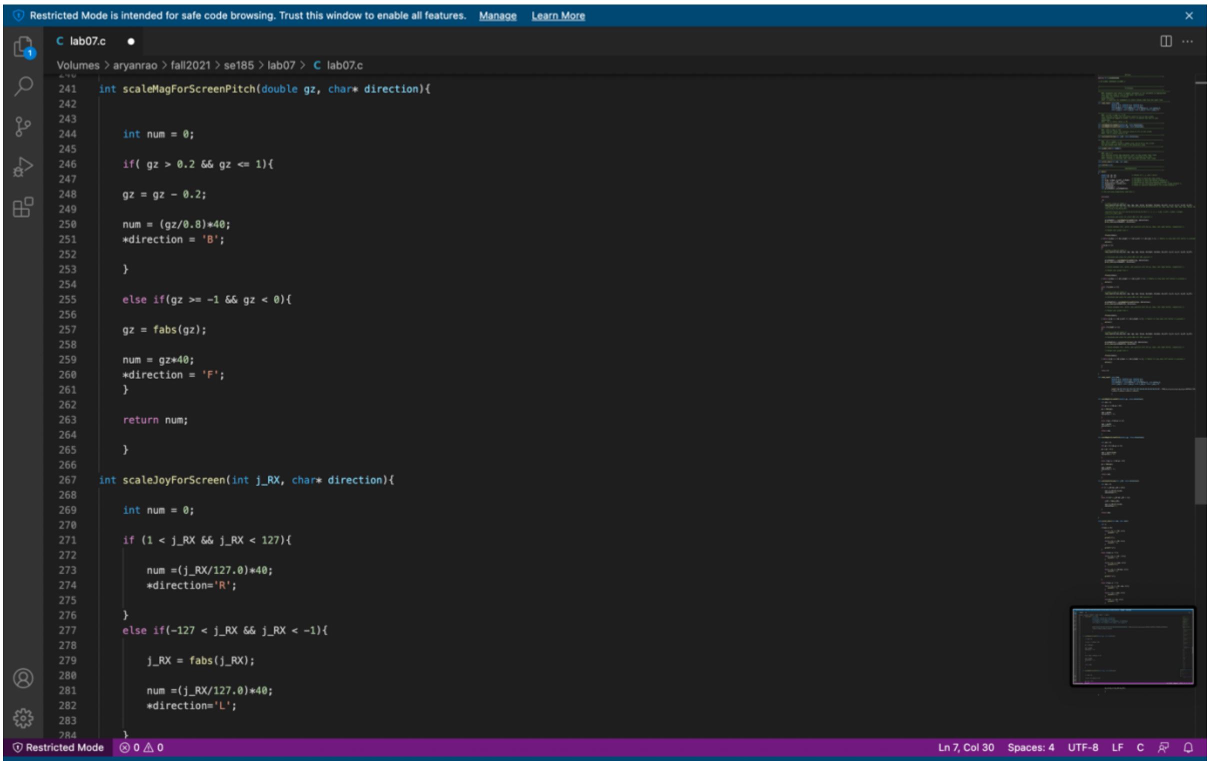
```c
    } while (b_Down != 1 && b_Right != 1 && b_Left != 1 && b_Up != 1); /* Modify to stop when left button is pressed */

        extras();

    if(b_Up == 1){
    do
    {
        /* Scan a line of input */
        read_input(&t,&ax,&ay,&az, &gx, &gy, &gz, &b_Up, &b_Right, &b_Down, &b_Left, &j_LX, &j_LY, &j_RX, &j_RY);


        /* Calculate and scale for pitch AND roll AND joystick */

        printNumRoll = scaleMagForScreenRoll(gx, &direction);
        print_chars(printNumRoll, direction);


        /* Switch between roll, pitch, and joystick with the up, down, and right button, respectivly */

        /* Output your graph line */


        fflush(stdout);

    } while (b_Down != 1 && b_Right != 1 && b_Left != 1); /* Modify to stop when left button is pressed */

        extras();
    }

    else if(b_Down == 1){
    do
    {
        /* Scan a line of input */
        read_input(&t,&ax,&ay,&az, &gx, &gy, &gz, &b_Up, &b_Right, &b_Down, &b_Left, &j_LX, &j_LY, &j_RX, &j_RY);

        /* Calculate and scale for pitch AND roll AND joystick */


        printNumPitch = scaleMagForScreenPitch(gz, &direction);
        print_chars(printNumPitch, direction);

        /* Switch between roll, pitch, and joystick with the up, down, and right button, respectivly */
```


Restricted Mode is intended for safe code browsing. Trust this window to enable all features.
Manage
Learn More
lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > lab07.c
Restricted Mode
0 0
Ln 7, Col 30
Spaces: 4
UTF-8
LF
C

```c
        printNumPitch = scaleMagForScreenPitch(gz, &direction);
        print_chars(printNumPitch, direction);

        /* Switch between roll, pitch, and joystick with the up, down, and right button, respectivly */

        /* Output your graph line */


        fflush(stdout);

    } while (b_Up != 1 && b_Left != 1 && b_Right != 1); /* Modify to stop when left button is pressed */

        extras();
    }

    else if(b_Right == 1){
    do
    {
        /* Scan a line of input */
        read_input(&t,&ax,&ay,&az, &gx, &gy, &gz, &b_Up, &b_Right, &b_Down, &b_Left, &j_LX, &j_LY, &j_RX, &j_RY);

        /* Calculate and scale for pitch AND roll AND joystick */


        printNumPitch = scaleJoyForScreen(j_RX, &direction);
        print_chars(printNumPitch, direction);

        /* Switch between roll, pitch, and joystick with the up, down, and right button, respectivly */

        /* Output your graph line */


        fflush(stdout);

    } while (b_Up != 1 && b_Down != 1 && b_Right != 1); /* Modify to stop when left button is pressed */

        extras();

    }
    }

    return 0;

}
```

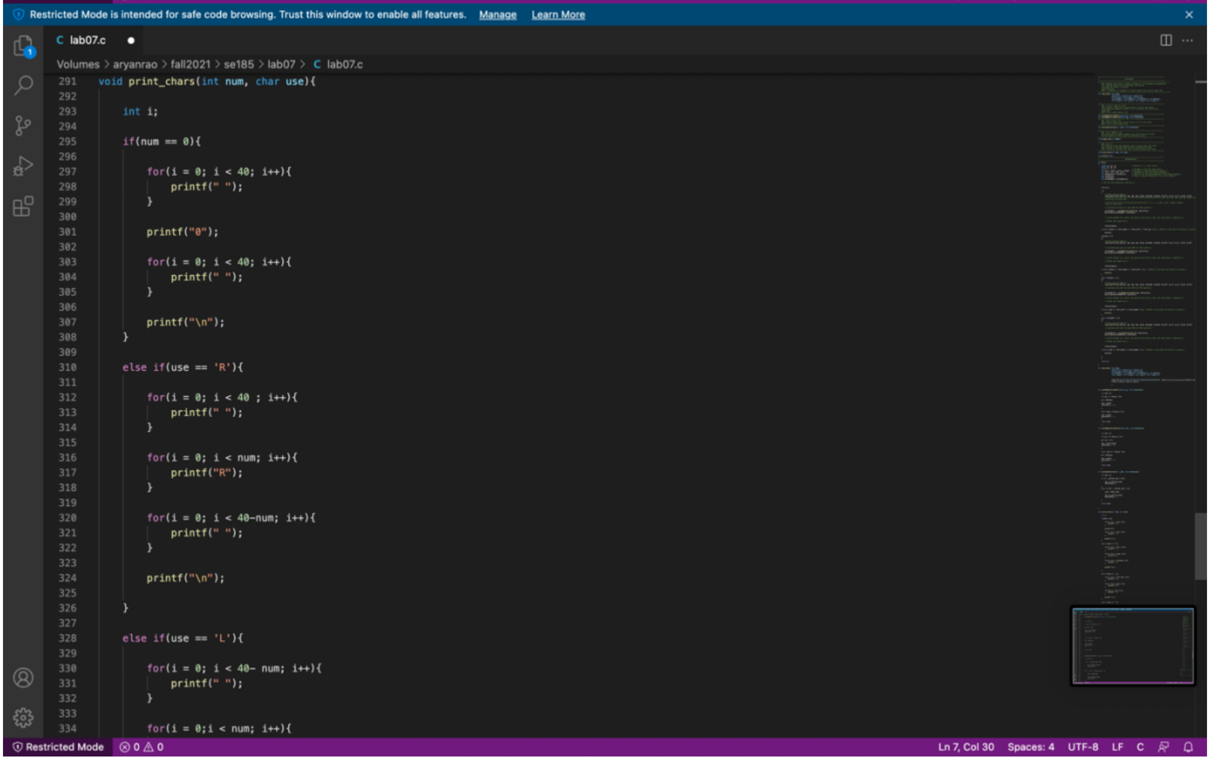

Restricted Mode is intended for safe code browsing. Trust this window to enable all features.
Manage
Learn More
lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > lab07.c
Restricted Mode
0 0
Ln 7, Col 30
Spaces: 4
UTF-8
LF
C

```c
int read_input( int* time,
                double* a_x, double* a_y, double* a_z,
                double* g_x, double* g_y, double* g_z,
                int* button_T, int* button_C, int* button_X, int* button_S,
                int* l_joy_x, int* l_joy_y, int* r_joy_x, int* r_joy_y ){


                scanf("%d,%lf,%lf,%lf,%lf,%lf,%lf,%d,%d,%d,%d,%d,%d,%d,%d", time,a_x,a_y,a_z,g_x,g_y,g_z,button_T,button_C,button_X,button_S,
                l_joy_x,l_joy_y,r_joy_x,r_joy_y);

                }


int scaleMagForScreenRoll(double gx, char* direction){

    int num = 0;

    if( gx >= -1 && gx < 0){

    gx = fabs(gx);

    num = gx*40;
    *direction = 'R';

    }

    else if(gx > 0 && gx <= 1){

    num = gx*40;
    *direction = 'L';
    }

    return num;

    }

int scaleMagForScreenPitch(double gz, char* direction){


    int num = 0;

    if( gz > 0.2 && gz <= 1){

    gz = gz - 0.2;
```



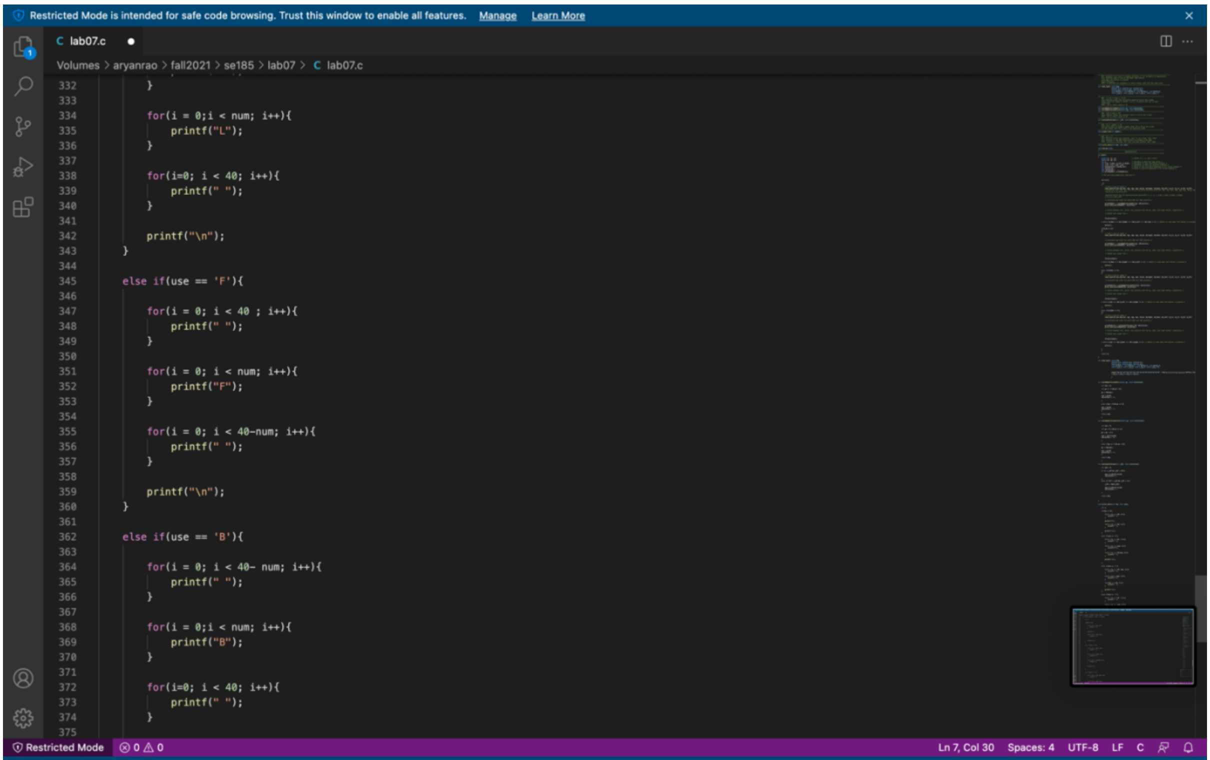
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

```c
int scaleMagForScreenPitch(double gz, char* direction){


    int num = 0;

    if( gz > 0.2 && gz <= 1){

    gz = gz - 0.2;

    num = (gz/0.8)*40;
    *direction = 'B';

    }

    else if(gz >= -1 && gz < 0){

    gz = fabs(gz);

    num = gz*40;
    *direction = 'F';
    }

    return num;

    }

int scaleJoyForScreen(int j_RX, char* direction){

    int num = 0;

    if (1 < j_RX && j_RX < 127){

        num =(j_RX/127.0)*40;
        *direction='R';

    }
    else if(-127 < j_RX && j_RX < -1){

        j_RX = fabs(j_RX);

        num =(j_RX/127.0)*40;
        *direction='L';

    }
```



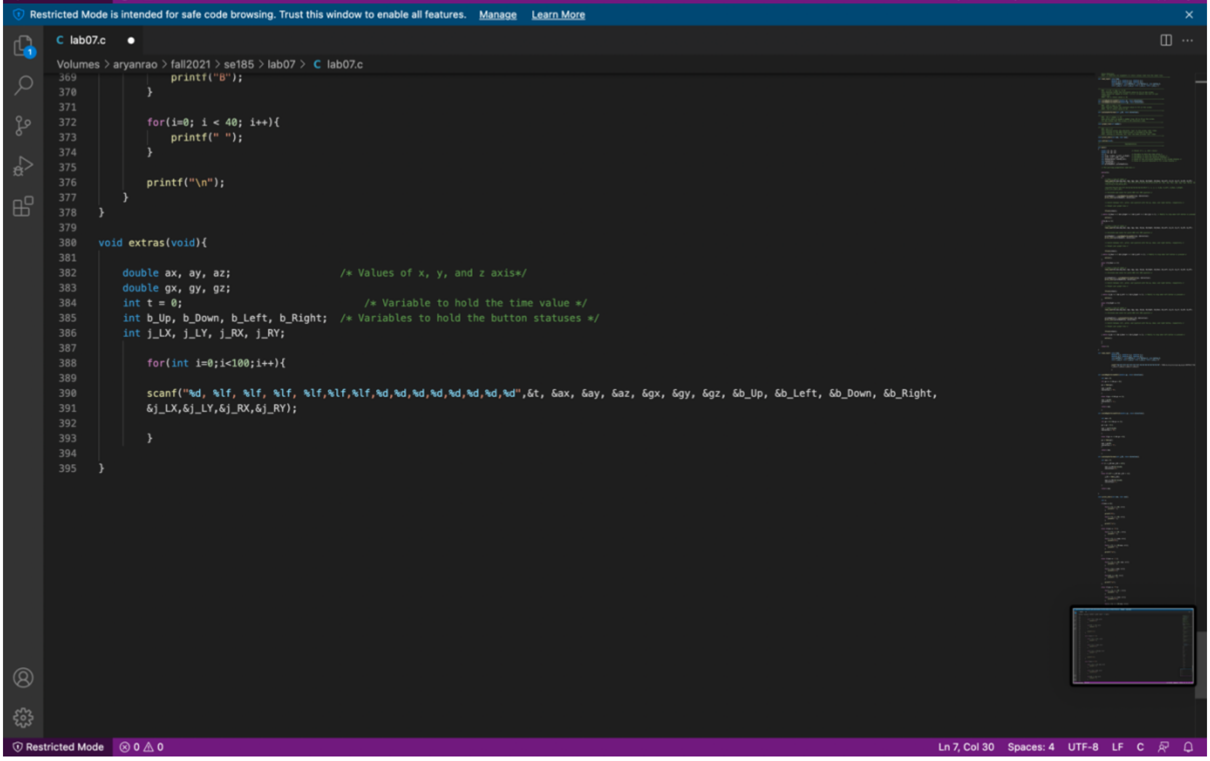
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.

```c
void print_chars(int num, char use){

    int i;

    if(num == 0){

        for(i = 0; i < 40; i++){
            printf(" ");
        }

        printf("0");

        for(i = 0; i < 40; i++){
            printf(" ");
        }

        printf("\n");
    }

    else if(use == 'R'){

        for(i = 0; i < 40 ; i++){
            printf(" ");
        }

        for(i = 0; i < num; i++){
            printf("R");
        }

        for(i = 0; i < 40-num; i++){
            printf(" ");
        }

        printf("\n");

    }

    else if(use == 'L'){

        for(i = 0; i < 40- num; i++){
            printf(" ");
        }

        for(i = 0;i < num; i++){
```


Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > C lab07.c
Restricted Mode ⊗ 0 ⚠ 0
Ln 7, Col 30 Spaces: 4 UTF-8 LF C

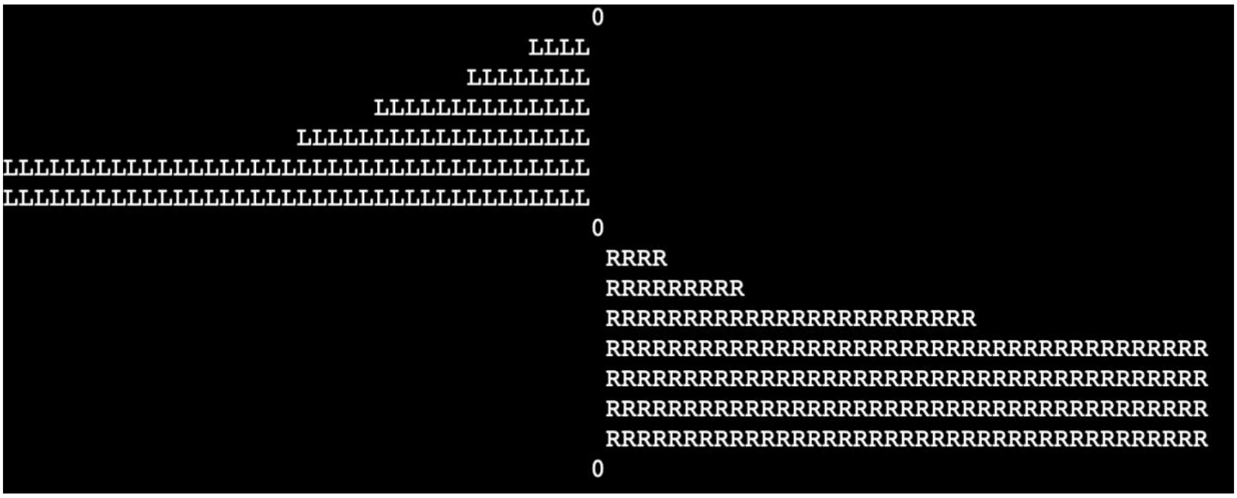
```
        }

        for(i = 0;i < num; i++){
            printf("L");
        }

        for(i=0; i < 40; i++){
            printf(" ");
        }

        printf("\n");
    }

    else if(use == 'F'){

        for(i = 0; i < 40 ; i++){
            printf(" ");
        }

        for(i = 0; i < num; i++){
            printf("F");
        }

        for(i = 0; i < 40-num; i++){
            printf(" ");
        }

        printf("\n");
    }

    else if(use == 'B'){

        for(i = 0; i < 40- num; i++){
            printf(" ");
        }

        for(i = 0;i < num; i++){
            printf("B");
        }

        for(i=0; i < 40; i++){
            printf(" ");
        }

```


Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab07.c
Volumes > aryanrao > fall2021 > se185 > lab07 > C lab07.c
Restricted Mode ⊗ 0 ⚠ 0
Ln 7, Col 30 Spaces: 4 UTF-8 LF C

```
            printf("B");
        }

        for(i=0; i < 40; i++){
            printf(" ");
        }

        printf("\n");
    }
}

void extras(void){

    double ax, ay, az;                  /* Values of x, y, and z axis*/
    double gx, gy, gz;
    int t = 0;                              /* Variable to hold the time value */
    int b_Up, b_Down, b_Left, b_Right;  /* Variables to hold the button statuses */
    int j_LX, j_LY, j_RX, j_RY;

        for(int i=0;i<100;i++){

        scanf("%d, %lf, %lf, %lf, %lf,%lf,%lf,%d,%d,%d,%d,%d,%d,%d,%d",&t, &ax, &ay, &az, &gx, &gy, &gz, &b_Up, &b_Left, &b_Down, &b_Right,
        &j_LX,&j_LY,&j_RX,&j_RY);

        }

}
```

OUTPUT



```
                                      0
                                  LLLL
                              LLLLLLLL
                        LLLLLLLLLLLLLL
                   LLLLLLLLLLLLLLLLLLL
LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
                                      0
                                       RRRR
                                       RRRRRRRRR
                                       RRRRRRRRRRRRRRRRRRRRRRRR
                                       RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
                                       RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
                                       RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
                                       RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
                                      0
```