**EXDROP**

**LAB #6**

**SECTION AA**

**SUBMITTED BY:**

**ANDREW HANCOCK**

**SUBMISSION DATE:**

**10/23/16**

**Lab 6 part 1 code**

#include <stdio.h>

#include <math.h>

#define TRUE 1

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

int close\_to(double TOL, double point, double value);

void printStuff();

int k = 0;

double TOL = 0.2;

int main(void) {

int t, b1, b2, b3, b4, b5, s, i = 0;

double ax, ay, az, secondsFalling, distanceFallen;

printStuff();

while (TRUE) {

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

 k++;

 if((k % 10) == 0){

  printf(".");

  fflush(stdout);

 }

 if(close\_to(TOL, mag(ax, ay, az), 0.1)){

  while(close\_to(TOL, mag(ax, ay, az), 0.1)){

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

   if(i == 0){

    fflush(stdout);

    printf("\nI'm falling");

    fflush(stdout);

    i++;

   }

   i++;

   if((i % 10) == 0){

    fflush(stdout);

    printf("!");

    fflush(stdout);

   }

  }

  secondsFalling = ((double) i / 500.0);

  distanceFallen = 0.0 + (0.0 \* 0.0) + (1.0/2.0)\*(9.8)\*((secondsFalling) \* (secondsFalling));

  printf("\nI fell %lf meters in %lf seconds", distanceFallen, secondsFalling);

  break;

 }

}

return 0;

}

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

double magnitude = sqrt((pow(ax, 2.0) + pow(ay, 2.0) + pow(az, 2.0)));

return magnitude;

}

int close\_to(double TOL, double point, double value){

if((point < (value + TOL)) && (point > (value - TOL))){

 return 1;

} else {

 return 0;

}

}

void printStuff(){

fflush(stdout);

printf("Ok, I'm receiving data.\n");

fflush(stdout);

printf("I'm waiting");

fflush(stdout);

}

**Lab 6 part 2 code**

#include <stdio.h>

#include <math.h>

#define TRUE 1

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

int close\_to(double TOL, double point, double value);

void printStuff();

int k = 0;

double TOL = 0.2;

int main(void) {

int t, b1, b2, b3, b4, b5, s, i = 0;

double ax, ay, az, secondsFalling, distanceFallen;

printStuff();

while (TRUE) {

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

 k++;

 if((k % 10) == 0){

  printf(".");

  fflush(stdout);

 }

 if(close\_to(TOL, mag(ax, ay, az), 0.1)){

  while(close\_to(TOL, mag(ax, ay, az), 0.1)){

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

   if(i == 0){

    fflush(stdout);

    printf("\nI'm falling");

    fflush(stdout);

    i++;

   }

   i++;

   if((i % 10) == 0){

    fflush(stdout);

    printf("!");

    fflush(stdout);

   }

  }

  secondsFalling = ((double) i / 500.0);

  distanceFallen = 0.0 + (0.0 \* 0.0) + (1.0/2.0)\*(9.8)\*((secondsFalling) \* (secondsFalling));

  printf("\nI fell %lf meters in %lf seconds", distanceFallen, secondsFalling);

  break;

 }

}

return 0;

}

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

double magnitude = sqrt((pow(ax, 2.0) + pow(ay, 2.0) + pow(az, 2.0)));

return magnitude;

}

int close\_to(double TOL, double point, double value){

if((point < (value + TOL)) && (point > (value - TOL))){

 return 1;

} else {

 return 0;

}

}

void printStuff(){

fflush(stdout);

printf("Ok, I'm receiving data.\n");

fflush(stdout);

printf("I'm waiting");

fflush(stdout);

}

**Questions and Experiments 6-1**

1. The results are very consistent. There are a lot of factors that could cause variation. Firstly, the cord that is connected to it affects every part of the drop. Secondly, the way it lands affects the overall distance, because if it lands flat than it will be longer than if it lands on the edge.
2. According to my program the total distance of the fall was 0.082641 meters. This is weird because when I ran the program and then dropped it in my lap and the distance was correct, so I’m not sure why it has trouble with the data. I think it might have something to do with the time since my program does the time off of the number of iterations of the while loop.

Question and experiments part 2

1. There are only very small differences in the drops in the lab. When I drop using the sample data for the 9m drop my program returns the same value, which should be expected.
2. There weren’t really any difficulties adding in part 2. I just needed to type in the equations and add some new variables.

**Problem**

In this lab I had to write a program that would calculate the distance and time of a drop of the Arduino Esplora. It also had to output periods and exclamation marks for when it was either resting or falling. The Esplora has an accelerometer that returns values in all three axes. The value of each axis goes toward zero when it falls so that is what I used to tell when the fall started.

**Analysis**

In this program I had to use the mag function and the close to function in order to tell what the magnitude of acceleration was at any time and if a value was close to what it needed to be to meet a certain condition. The program input was the accelerometer values and the time. The output takes those values and uses them to find the velocity, time and position. From there, using some physics equations the program calculates the total position change and then finds the difference between that and the previously calculated value which didn’t factor in air resistance.

**Design**

My program starts by detecting whether or not data input is happening. If so, it prints the “I’m waiting” line. After that a counter starts and every 10 iterations a period is output. Inside the while loop I have an if statement that detects when the magnitude of acceleration is below 0.1. When it is it gets put into another while loop that starts by saying “I’m falling”, and then for every 10 iterations outputs an exclamation mark. Once the while loop condition is broken, which is for the magnitude to be within 0.1, It outputs the total length of the fall and then calculates what it would have been without air resistance. It outputs this difference as a percentage. Finally, the program ends once the data has been output and the drop has happened.

**Testing**

To test this program, I just had to drop the Arduino a bunch and make sure the tolerance was enough to allow some movement but also small enough to detect the drop right away. I also needed to make sure that the equations were calculating the correct distance and that the percentage difference was the correct amount.

**Comments**

Not knowing some very simple things such as the fflush statement and the if statement putting the program directly into another while statement made for some huge headaches in writing this program. It leads to some long hours in the lab but it was very satisfying when the program finally worked. I learned how to continually scan data even inside another while loop which I think will be very useful in the future.