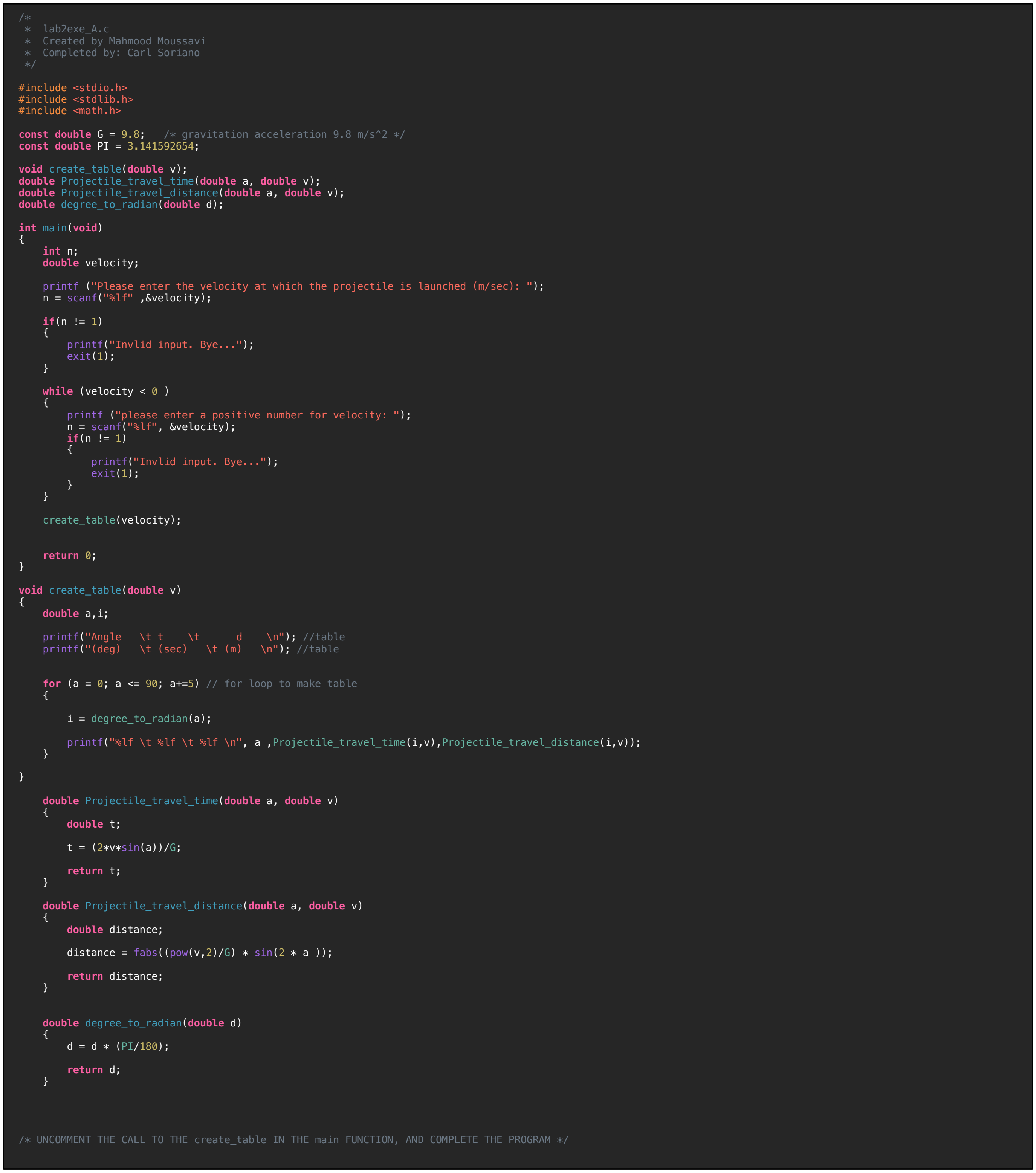
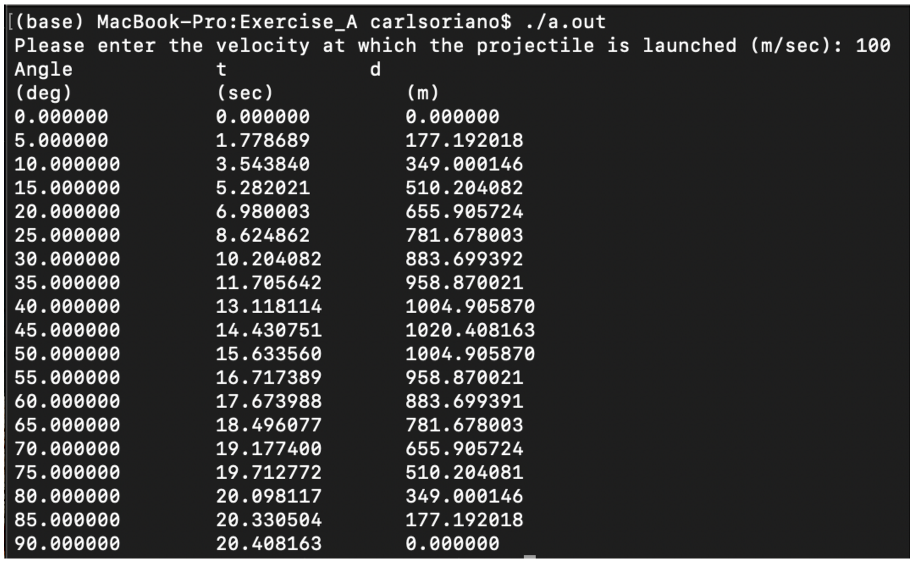
Course: Programming Fundamental – ENSF 337
Lab #: Lab 1
Instructor: M. Moussavi
Student Names: Carl Soriano, Zuriel Abasola
Lab Section: B01
Date submitted: Sept 28, 2022

Exercise A



```c
/*
 *  lab2exe_A.c
 *  Created by Mahmood Moussavi
 *  Completed by: Carl Soriano
 */

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

const double G = 9.8;   /* gravitation acceleration 9.8 m/s^2 */
const double PI = 3.141592654;

void create_table(double v);
double Projectile_travel_time(double a, double v);
double Projectile_travel_distance(double a, double v);
double degree_to_radian(double d);

int main(void)
{
    int n;
    double velocity;

    printf ("Please enter the velocity at which the projectile is launched (m/sec): ");
    n = scanf("%lf" ,&velocity);

    if(n != 1)
    {
        printf("Invlid input. Bye...");
        exit(1);
    }

    while (velocity < 0 )
    {
        printf ("please enter a positive number for velocity: ");
        n = scanf("%lf", &velocity);
        if(n != 1)
        {
            printf("Invlid input. Bye...");
            exit(1);
        }
    }

    create_table(velocity);


    return 0;
}

void create_table(double v)
{
    double a,i;

    printf("Angle   \t t    \t      d    \n"); //table
    printf("(deg)   \t (sec)   \t (m)   \n"); //table


    for (a = 0; a <= 90; a+=5) // for loop to make table
    {

        i = degree_to_radian(a);

        printf("%lf \t %lf \t %lf \n", a ,Projectile_travel_time(i,v),Projectile_travel_distance(i,v));
    }

}

    double Projectile_travel_time(double a, double v)
    {
        double t;

        t = (2*v*sin(a))/G;

        return t;
    }

    double Projectile_travel_distance(double a, double v)
    {
        double distance;

        distance = fabs((pow(v,2)/G) * sin(2 * a ));

        return distance;
    }


    double degree_to_radian(double d)
    {
        d = d * (PI/180);

        return d;
    }



/* UNCOMMENT THE CALL TO THE create_table IN THE main FUNCTION, AND COMPLETE THE PROGRAM */
```

Exercise A output:



```
(base) MacBook-Pro:Exercise_A carlsoriano$ ./a.out
Please enter the velocity at which the projectile is launched (m/sec): 100
Angle           t           d
(deg)           (sec)          (m)
0.000000        0.000000       0.000000
5.000000        1.778689       177.192018
10.000000       3.543840       349.000146
15.000000       5.282021       510.204082
20.000000       6.980003       655.905724
25.000000       8.624862       781.678003
30.000000       10.204082      883.699392
35.000000       11.705642      958.870021
40.000000       13.118114      1004.905870
45.000000       14.430751      1020.408163
50.000000       15.633560      1004.905870
55.000000       16.717389      958.870021
60.000000       17.673988      883.699391
65.000000       18.496077      781.678003
70.000000       19.177400      655.905724
75.000000       19.712772      510.204081
80.000000       20.098117      349.000146
85.000000       20.330504      177.192018
90.000000       20.408163      0.000000
```

Exercise B

Diagram for point 1:

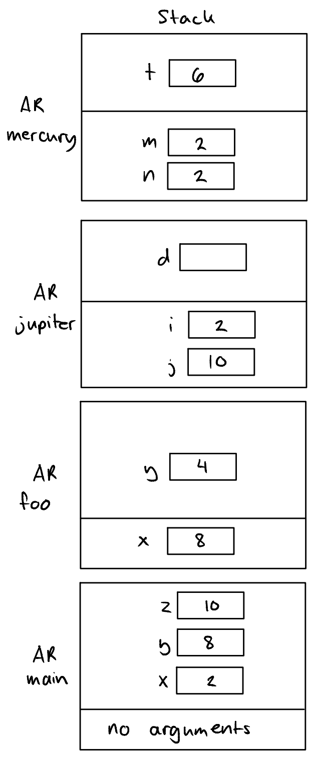


Diagram for point 2:

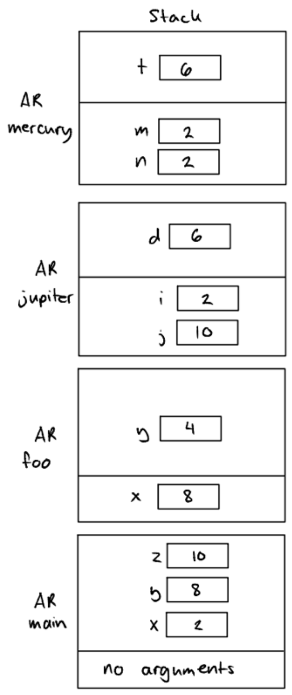
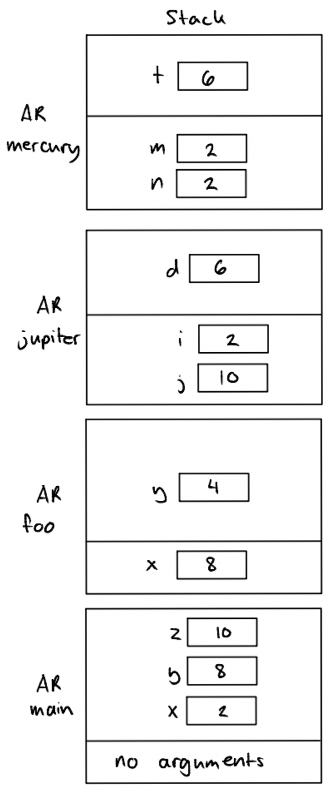
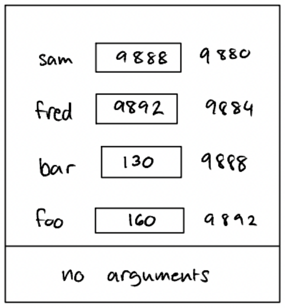


Diagram for point 3:

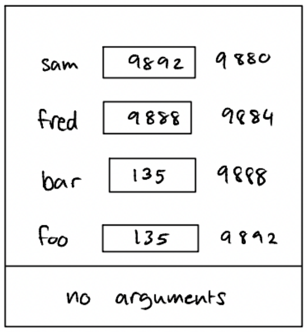


Exercise C

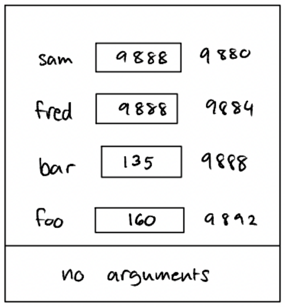
Point two:



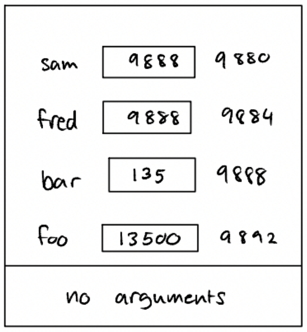
Point 4:



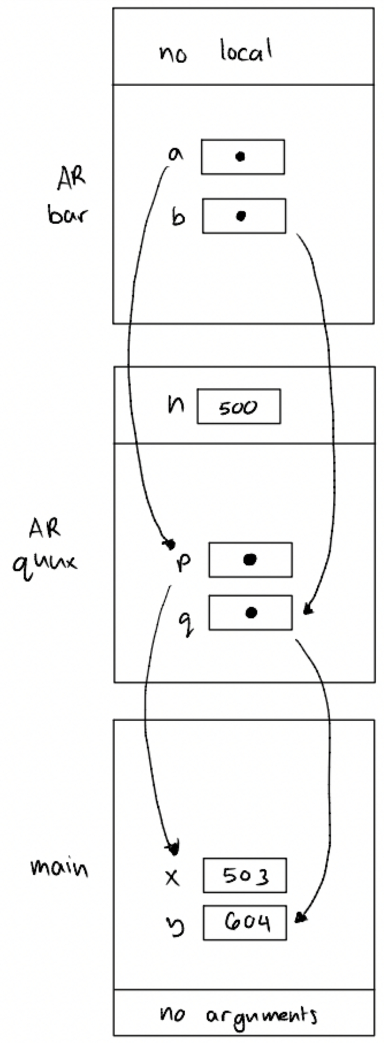
Point three:



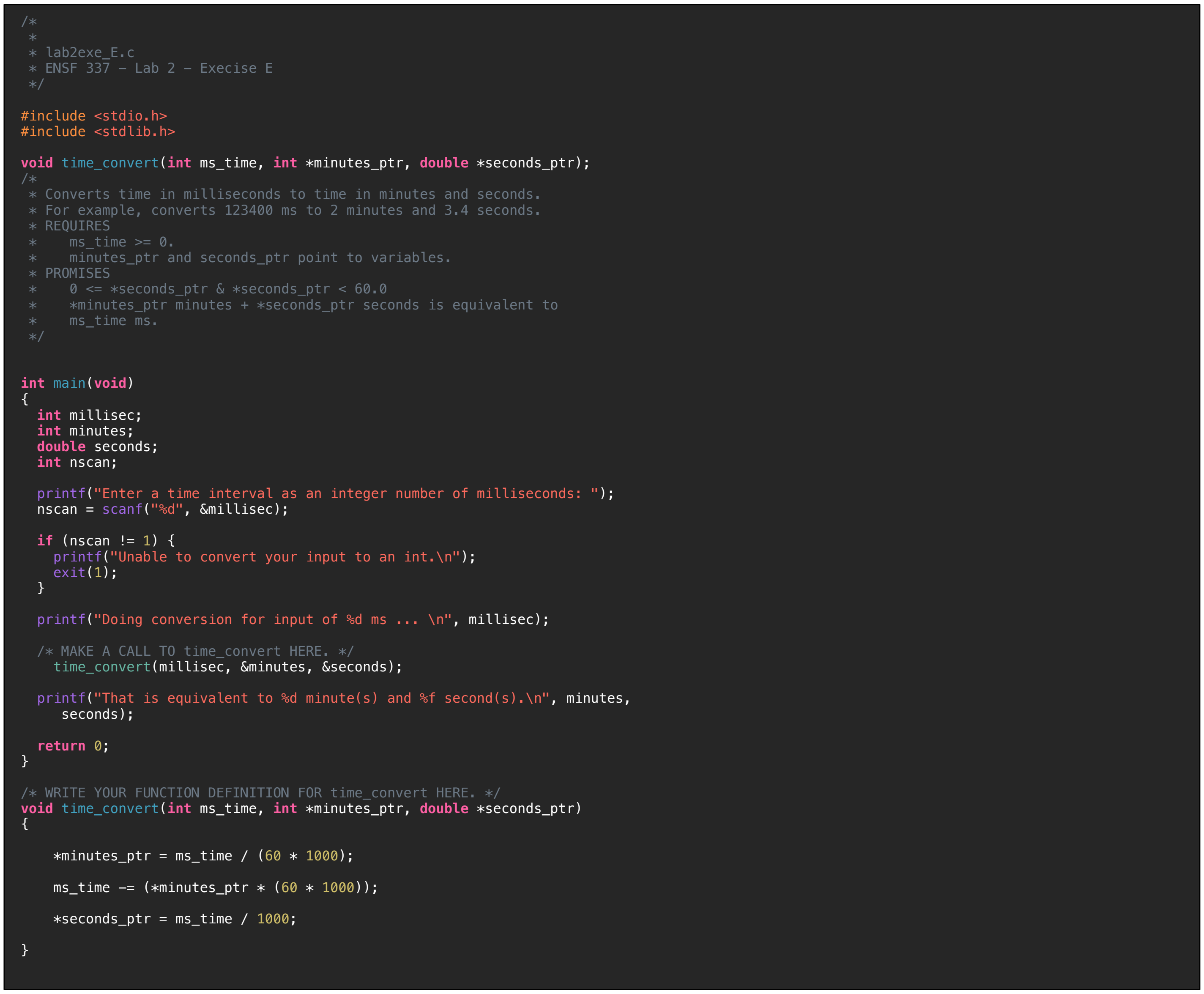
Point 5:



Exercise D

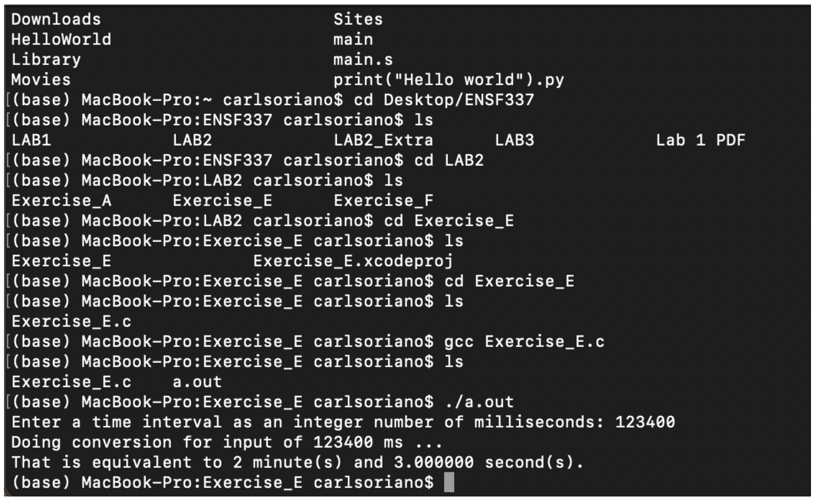


Exercise E



```c
/*
 *
 * lab2exe_E.c
 * ENSF 337 - Lab 2 - Execise E
 */

#include <stdio.h>
#include <stdlib.h>

void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr);
/*
 * Converts time in milliseconds to time in minutes and seconds.
 * For example, converts 123400 ms to 2 minutes and 3.4 seconds.
 * REQUIRES
 *    ms_time >= 0.
 *    minutes_ptr and seconds_ptr point to variables.
 * PROMISES
 *    0 <= *seconds_ptr & *seconds_ptr < 60.0
 *    *minutes_ptr minutes + *seconds_ptr seconds is equivalent to
 *    ms_time ms.
 */


int main(void)
{
  int millisec;
  int minutes;
  double seconds;
  int nscan;

  printf("Enter a time interval as an integer number of milliseconds: ");
  nscan = scanf("%d", &millisec);

  if (nscan != 1) {
    printf("Unable to convert your input to an int.\n");
    exit(1);
  }

  printf("Doing conversion for input of %d ms ... \n", millisec);

  /* MAKE A CALL TO time_convert HERE. */
    time_convert(millisec, &minutes, &seconds);

  printf("That is equivalent to %d minute(s) and %f second(s).\n", minutes,
     seconds);

  return 0;
}

/* WRITE YOUR FUNCTION DEFINITION FOR time_convert HERE. */
void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr)
{

    *minutes_ptr = ms_time / (60 * 1000);

    ms_time -= (*minutes_ptr * (60 * 1000));

    *seconds_ptr = ms_time / 1000;

}
```

Exercise E output:



```
Downloads                       Sites
HelloWorld                      main
Library                         main.s
Movies                          print("Hello world").py
[(base) MacBook-Pro:~ carlsoriano$ cd Desktop/ENSF337
[(base) MacBook-Pro:ENSF337 carlsoriano$ ls
LAB1            LAB2            LAB2_Extra      LAB3            Lab 1 PDF
[(base) MacBook-Pro:ENSF337 carlsoriano$ cd LAB2
[(base) MacBook-Pro:LAB2 carlsoriano$ ls
Exercise_A      Exercise_E      Exercise_F
[(base) MacBook-Pro:LAB2 carlsoriano$ cd Exercise_E
[(base) MacBook-Pro:Exercise_E carlsoriano$ ls
Exercise_E              Exercise_E.xcodeproj
[(base) MacBook-Pro:Exercise_E carlsoriano$ cd Exercise_E
[(base) MacBook-Pro:Exercise_E carlsoriano$ ls
Exercise_E.c
[(base) MacBook-Pro:Exercise_E carlsoriano$ gcc Exercise_E.c
[(base) MacBook-Pro:Exercise_E carlsoriano$ ls
Exercise_E.c    a.out
[(base) MacBook-Pro:Exercise_E carlsoriano$ ./a.out
Enter a time interval as an integer number of milliseconds: 123400
Doing conversion for input of 123400 ms ...
That is equivalent to 2 minute(s) and 3.000000 second(s).
(base) MacBook-Pro:Exercise_E carlsoriano$ 
```

Exercise F

No Submission for Part I

Submission for Part II

Run	**Inputs**	**Value of n**	**Value of i**	**Value of d**
1	12 0.56	2	12	0.569
2	5.12 9.56	2	5	.12
3	12 ab	1	12	1234.5
4	ab 12	0	333	1234.5
5	5ab 9.56	1	5	1234.5
6	13 67	2	13	67