Lab 1 Report

Course: ENSF 619 - Fall 2020
Lab #: Lab 1
Student Name: Davis Allan, 10016543
Submission Date: 09-22-2020

Exercise B:

Source Code:

```c
/*
* File Name: lab1exe_B.c
* Lab # and Assignment #: Lab 1 Exercise B
* Lab section: B01
* Completed by: Davis Allan, 10016543
* Submission Date: 09-22-2020
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.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 degrees[19];
    double radians[19];
    double deg = 0.0;

    for (int i = 0; i < 19; i++) {
        degrees[i] = deg;
        radians[i] = degree_to_radian(deg);
        deg += 5;
    }

    printf("%s \t\t  %s\t\t   %s\n", "Angle", "t", "d");
    printf("%s \t\t %s\t\t  %s\n", "(deg)", "(sec)", "(m)");

    for (int i = 0; i < 19; i++) {
        double time = Projectile_travel_time(radians[i], v);
        double dist = fabs(Projectile_travel_distance(radians[i], v));
        printf("%6.3lf \t\t%6.3lf \t\t%6.3lf\n", degrees[i], time, dist);
    }

}

double Projectile_travel_time(double a, double v) {
    return (2 * v * sin(a)) / G;
}

double Projectile_travel_distance(double a, double v) {
    return (pow(v, 2) / G) * sin(2 * a);
}

double degree_to_radian(double d) {
```

```
    return d * (PI / 180.0);
}
```

Program output

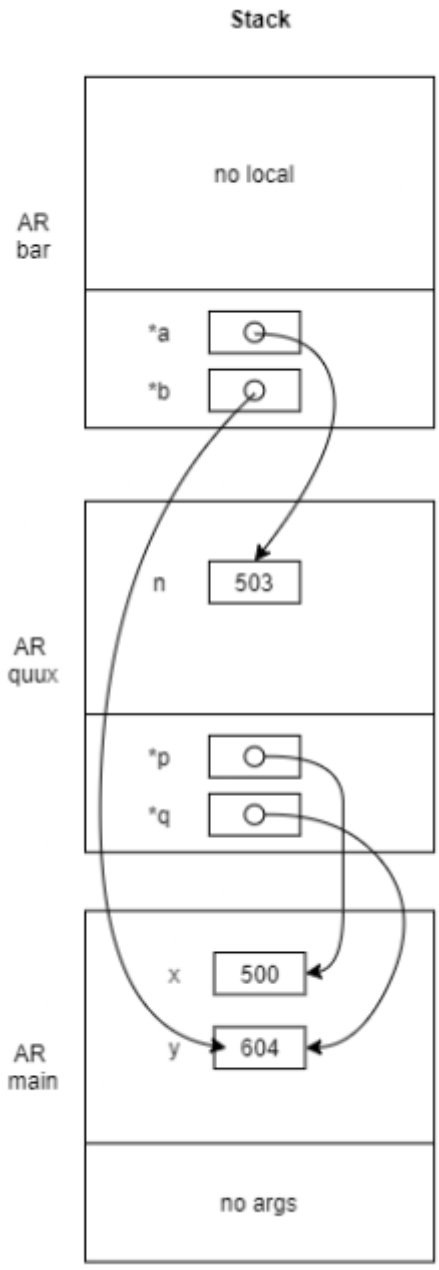
Testing with an input of 100m/s:

```
PS C:\Users\davis\Desktop\ENSF 619\Labs\Lab1> cd "c:\Users\davis\Desktop\ENSF 619\Labs\Lab1\"
Please enter the velocity at which the projectile is launched (m/sec): 100
Angle             t                d
(deg)           (sec)            (m)
 0.000          0.000           0.000
 5.000          1.779         177.192
10.000          3.544         349.000
15.000          5.282         510.204
20.000          6.980         655.906
25.000          8.625         781.678
30.000         10.204         883.699
35.000         11.706         958.870
40.000         13.118        1004.906
45.000         14.431        1020.408
50.000         15.634        1004.906
55.000         16.717         958.870
60.000         17.674         883.699
65.000         18.496         781.678
70.000         19.177         655.906
75.000         19.713         510.204
80.000         20.098         349.000
85.000         20.331         177.192
90.000         20.408           0.000
PS C:\Users\davis\Desktop\ENSF 619\Labs\Lab1>
```

Exercise D Part 2:

AR Diagram

Drawn using draw.io



Exercise E:

Source Code:

```c
/*
* File Name: lab1exe_E.c
* Lab # and Assignment #: Lab 1 Exercise E
* Lab section: B01
* Completed by: Davis Allan, 10016543
* Submission Date: 09-22-2020
*/

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

void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr);

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);

  time_convert(millisec, &minutes, &seconds);

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

  return 0;
}
```

```c
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.
 */
  double total_seconds = ms_time / 1000.0;
  *minutes_ptr = total_seconds / 60;
  *seconds_ptr = fmod(total_seconds, 60);
}
```

Program Output:

Testing 123400ms input