Course: ENSF 614 – Fall 2023
Lab 1:
Instructor: M. Moussavi
Student Name: Emmanuel Alafonye
Submission Date: September 20, 2023

Lab 1 B

```
/*
 *  File Name: lab1exe_B.cpp
 *  Assignment: ENSF 614 Lab 1, exercise B
 *  Created by Mahmood Moussavi
 *  Completed by: Emmanuel Alafonye
 *  Submission Date: Sept 20, 2023.
 */

#include <iostream>
#include <cmath>
#include <iomanip> // Include the <iomanip> header for setprecision and setw
using namespace std;

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

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){

   double velocity;
   cout << "Please enter the velocity at which the projectile is launched (m/sec): ";
   cin >> velocity;

   if(!cin){  // means if cin failed to read
      cout << "Invlid input. Bye...\n";
      exit(1);
   }

   while (velocity < 0 ){
      cout << "\nplease enter a positive number for velocity: ";
      cin >> velocity;
      if(!cin){
         cout << "Invlid input. Bye...";
         exit(1);
      }
   }

   create_table(velocity);

   return 0;
}

void create_table(double v){
   cout << "Angle (deg) Time (s) Distance (m) " << endl;
```

```
   for (int angle = 0; angle <=90; angle+= 5) {

       double radian = degree_to_radian(angle);
       double time = Projectile_travel_time(radian, v);
       double distance = Projectile_travel_distance(radian, v);

       cout << fixed << setprecision(9) << setw(10) << angle << " " << setw(8) << time << " " << setw(12) <<
distance << endl;

   }
}

double Projectile_travel_time(double a, double v){ // Use to calculate the fligt time
   return (2.0 * v * sin(a)) / G;
}

double Projectile_travel_distance(double a, double v){ // Used to calculate the horizontal distance
   return (v * v * sin(2.0 * a)) / G;
}

double degree_to_radian(double d){ // Units conversion from degree to radians
   return (d * PI) / 180.0;
}
```

Sample Run:

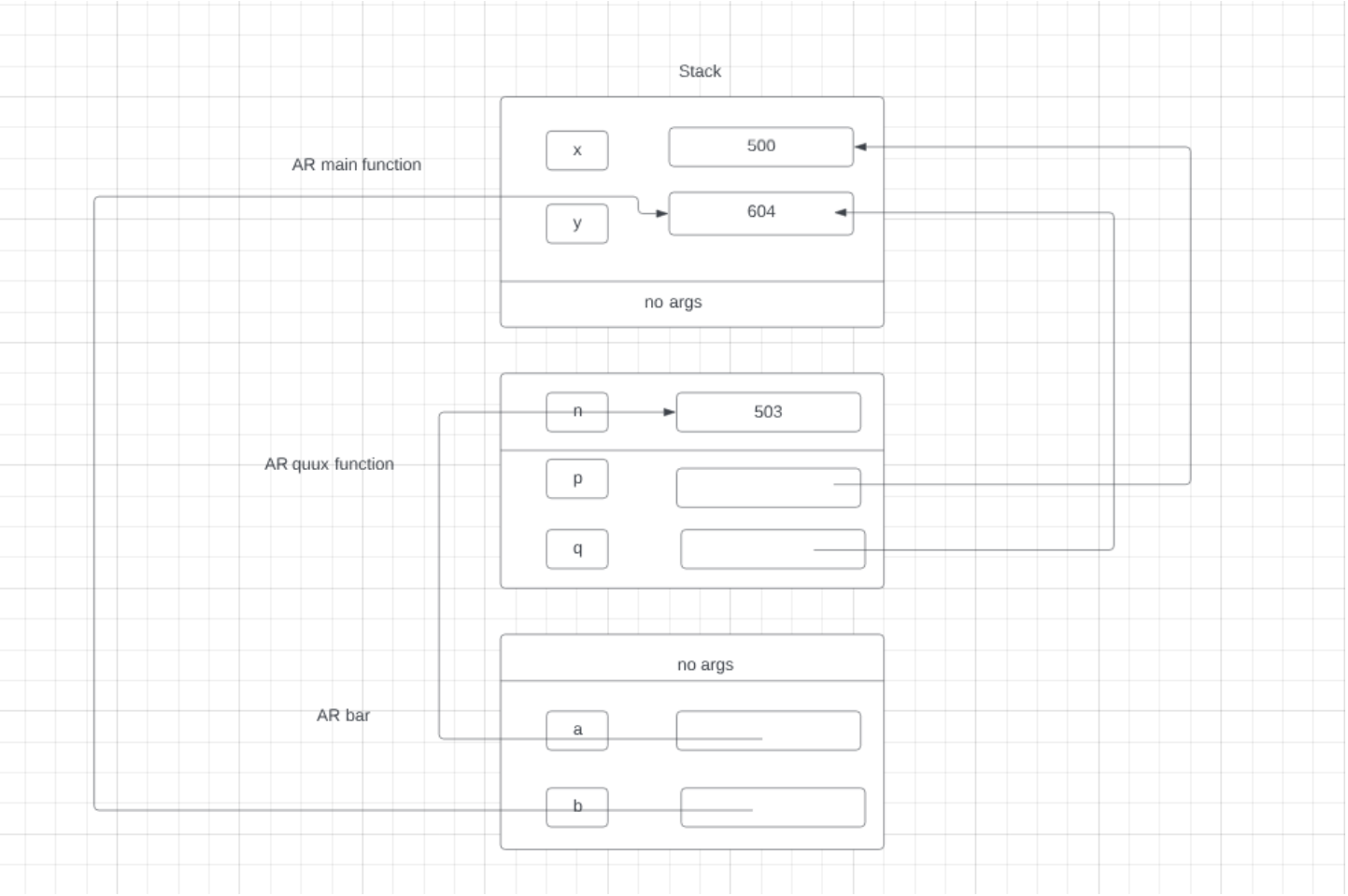
```
Please enter the velocity at which the projectile is launched (m/sec): 10
Angle (deg) Time (s) Distance (m)
         0 0.000000000  0.000000000
         5 0.177868863  1.771920181
        10 0.354384036  3.490001463
        15 0.528202133  5.102040817
        20 0.698000293  6.559057242
        25 0.862486249  7.816780033
        30 1.020408163  8.836993917
        35 1.170564156  9.588700213
        40 1.311811448 10.049058705
        45 1.443075064 10.204081633
        50 1.563356007 10.049058704
        55 1.671738866  9.588700211
        60 1.767398783  8.836993915
        65 1.849607729  7.816780030
        70 1.917740043  6.559057239
        75 1.971277197  5.102040813
        80 2.009811741  3.490001459
        85 2.033050404  1.771920176
        90 2.040816327 -0.000000004
Program ended with exit code: 0
```

Lab 1 D2

```
/*
 *  File Name: lab1exe_D2.cpp
 *  Assignment: ENSF 614 Lab 1, exercise D
 *  Created by Mahmood Moussavi
 *  Completed by: Emmanuel Alafonye
 *  Submission Date: Sept 20, 2023.
 */


#include <iostream>
using namespace std;

void bar(int *a, int *b);
void quux(int *p, int *q);

int main(void){
  int x = 500, y = 600;
  quux(&x, &y);
  cout << "x is " << x << ", y is " << y << "." << endl;
  return 0;
}

void bar(int *a, int *b){
  *a += 3;
  *b += 4;

  /* point one */
  cout << "*a is " << *a << ", *b is " << *b << ".\n";
}

void quux(int *p, int *q){
  int n;
  n = *p;
  bar(&n, q);
  cout << "*p is "<< *p << ", *q is " << *q << ".\n";
}
```

Output:

```
*a is 503, *b is 604.
*p is 500, *q is 604.
x is 500, y is 604.
Program ended with exit code: 0
```



Lab 1 E

```
/*
 *  File Name: lab1exe_E.cpp
 *  Assignment: ENSF 614 Lab 1, exercise E
 *  Created by Mahmood Moussavi
 *  Completed by: Emmanuel Alafonye
 *  Submission Date: Sept 20, 2023.
 */
#include <iostream>
using namespace std;

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;

  cout << "Enter a time interval as an integer number of milliseconds: ";

 // printf("Enter a time interval as an integer number of milliseconds: ");
  cin >> millisec;

  if (!cin) {
    cout << "Unable to convert your input to an int.\n";
    exit(1);
  }

  cout << "Doing conversion for input of " <<  millisec <<" milliseconds ... \n";

  /* MAKE A CALL TO time_convert HERE. */
  time_convert(millisec, &minutes, &seconds);
  cout << "That is equivalent to " << minutes << " minute(s) and " << seconds << " second(s).\n";

  return 0;
}

/* PUT YOUR FUNCTION DEFINITION FOR time_convert HERE. */
void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr){
    if(ms_time < 0){
        cerr << "Error: ms_time should be non-negative\n"; // Checks for error for negative values
        exit(1);
    }
    *minutes_ptr = ms_time / 60000;  // 1 minute is equivalent to 60,000 ms
    *seconds_ptr = (ms_time % 6000) / 1000.0; // Millisecond converted to seconds.
}
```

Output:

```
Enter a time interval as an integer number of milliseconds: 100
Doing conversion for input of 100 milliseconds ...
That is equivalent to 0 minute(s) and 0.1 second(s).
Program ended with exit code: 0
```