Course: ENSF 614 - Fall 2023

Lab #: Lab 1

Instructor: Mahmood Moussavi

Student Name: Christian Valdez

Submission Date: September 20, 2023

Exercise B:

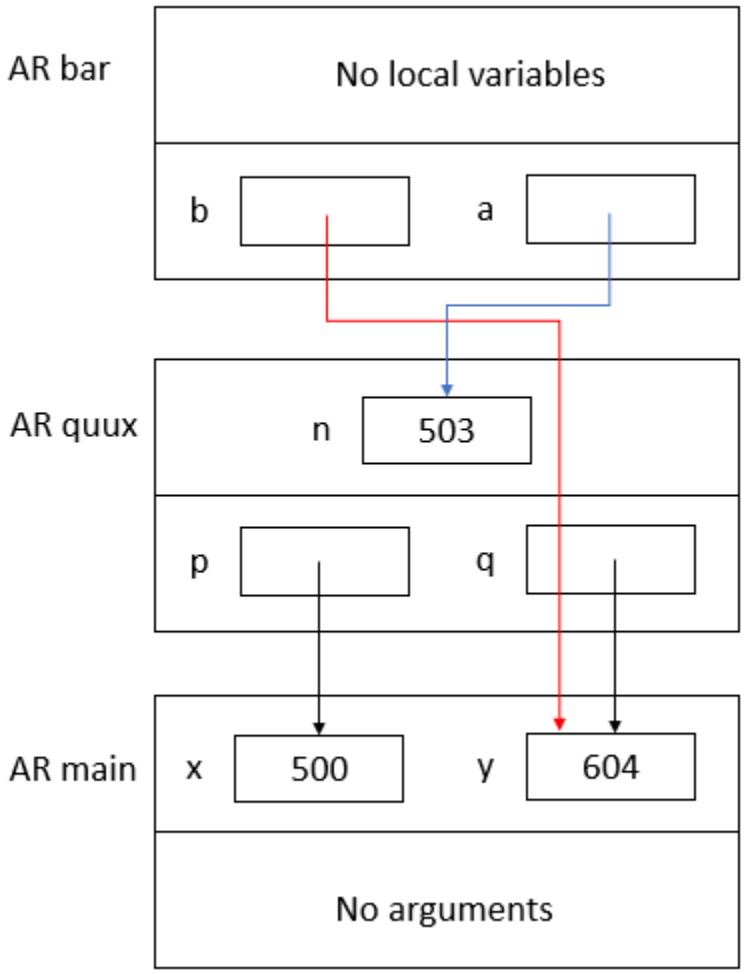
```
/*
 *  Filename: lab1exe_B.cpp
 *  Assignment: Lab 1 Exercise B
 *  Section: B01
 *  Completed by: Christian Valdez
 *  Submission date: Sep 20, 2023
 */

#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;

const double G = 9.8;   /* Gravitational acceleration 9.8 m/s^2 */
const double PI = 3.141592654;
const int MAX_ANGLE_DEG = 95; // Maximum angle in degrees to be used in the table

/**
 * Create a table displaying the time and distance traveled by a projectile
 * at varying angles for a given velocity.
 *
 * @param v - Initial velocity of the projectile.
 */
void create_table(double v);

/**
 * Calculate the time of flight for a projectile.
 *
 * @param a - Gravitational acceleration.
 * @param v - Initial velocity of the projectile.
 * @return Time in seconds the projectile remains in the air.
 */
double Projectile_travel_time(double a, double v);

/**
 * Calculate the horizontal distance traveled by a projectile.
 *
 * @param a - Gravitational acceleration.
 * @param v - Initial velocity of the projectile.
 * @return Distance in meters the projectile covers horizontally.
 */
double Projectile_travel_distance(double a, double v);

/**
 * Convert a given angle from degrees to radians.
 *
 * @param d - Angle in degrees.
 * @return Equivalent angle in radians.
 */
double degree_to_radian(double d);

double angleDeg = 0;  // Angle in degrees for the projectile
double angleRad;      // Angle in radians for the projectile

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

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

    // Check for valid input
    if (!cin) {
        cout << "Invalid input. Bye...\n";
        exit(1);
    }

    // Ensure the velocity is non-negative
    while (velocity < 0) {
        cout << "\nplease enter a positive number for velocity: ";
        cin >> velocity;
        if (!cin) {
            cout << "Invalid input. Bye...";
            exit(1);
        }
    }

    // Generate the table for projectile data
    create_table(velocity);
    return 0;
}

void create_table(double v) {
    double t, d; // time and distance

    // Table header
    cout << "Angle" << "\t\t t" << "\t\t d" << endl;
    cout << " (deg)" << "\t\t (sec)" << "\t\t (m)" << endl;
    cout << fixed << setprecision(6);

    // Calculate and display data for each angle until MAX_ANGLE_DEG
    while (angleDeg < MAX_ANGLE_DEG) {
        angleRad = degree_to_radian(angleDeg);
        t = Projectile_travel_time(G, v);
        d = Projectile_travel_distance(G, v);

        cout << angleDeg << "\t " << t << "\t " << d << endl;
        angleDeg += 5;  // Increase angle by 5 degrees
    }
}

double degree_to_radian(double d) {
    return angleDeg * (PI / 180);
}

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

double Projectile_travel_distance(double a, double v) {
    return (v * v * sin(2 * angleRad)) / a;
}
```

Sample output:

```
Microsoft Visual Studio Debug Console
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 D2


AR bar
No local variables
b
a
AR quux
n
503
p
q
AR main
x
500
y
604
No arguments

Exercise E

```
/*
 *  Filename: lab1exe_E.cpp
 *  Assignment: Lab 1 rxercise E
 *  Section: B01
 *  Completed by: Christian Valdez
 *  Submission date: Sep 20, 2023
 */

#include <iostream>
using namespace std;

/**
 * Converts milliseconds to minutes and seconds.
 * @param ms_time - Time in milliseconds to convert.
 * @param minutes_ptr - Pointer to store the resulting minutes.
 * @param seconds_ptr - Pointer to store the resulting seconds.
 */
void time_convert(int ms_time, int* minutes_ptr, double* seconds_ptr);


 // Constants for time conversion
const int MS_PER_SECOND = 1000;
const int SECONDS_PER_MINUTE = 60;

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

    // Prompt the user to input time in milliseconds
    cout << "Enter a time interval as an integer number of milliseconds: ";
    cin >> millisec;

    // Check if input conversion to integer was successful
    if (!cin) {
        cout << "Unable to convert your input to an int.\n";
        exit(1);
    }

    // Check for non-negative time input
    if (millisec < 0) {
        cout << "Please enter a non-negative value.\n";
        exit(1);
    }

    // Convert the input milliseconds to minutes and seconds
    cout << "Doing conversion for input of " << millisec << " milliseconds ... \n",
millisec;
    time_convert(millisec, &minutes, &seconds);
    cout << "That is equivalent to " << minutes << " minute(s) and " << seconds << "
second(s).\n";
    return 0;
}
```

```
void convert_time(int ms_time, int* minutes_ptr, double* seconds_ptr) {
    // Calculate the number of whole minutes
    *minutes_ptr = ms_time / (MS_PER_SECOND * SECONDS_PER_MINUTE);
    // Calculate the remaining seconds
    *seconds_ptr = (double)(ms_time % (MS_PER_SECOND * SECONDS_PER_MINUTE)) /
MS_PER_SECOND;
}
```

Sample output:

```
Microsoft Visual Studio Debug Console
Enter a time interval as an integer number of milliseconds: 123400
Doing conversion for input of 123400 milliseconds ...
That is equivalent to 2 minute(s) and 3.4 second(s).
```