**Course: ENSF 694** – Summer 2024

**Lab #:** Lab 1

**Instructor:** Mahmood Moussavi

**Student Name:** Ryan Baker

**Submission Date:** June 27, 2024

# Exercise A

## lab1exe\_A.cpp

/\*

 \*  lab1exe\_A.cpp

 \*  ENSF 694 Lab 1, exercise A

 \*  Created by Mahmood Moussavi

 \*  Completed by: Ryan Baker

 \*  Development Date: June 26, 2024

 \*/

#include <iostream>

#include <cmath>

#include <iomanip> // included for table formatting

using namespace std;

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

const double PI = 3.141592654;

void create\_table(double v);

/\* REQUIRES

 \*   v (double) the velocity to base the table on

 \* PROMISES

 \*   prints a table of angle (0-90 degrees) with corresponding times and distances

 \*/

double Projectile\_travel\_time(double a, double v);

/\* REQUIRES

 \*   0 < a < 90.

 \* PROMISES

 \*  return value is projectile travel time when maximum horizontal distance is reached (in seconds).

 \*/

double Projectile\_travel\_distance(double a, double v);

/\* REQUIRES

 \*   0 < a < 90.

 \* PROMISES

 \*  return value is maximum projectile distance (in metres).

 \*/

double degree\_to\_radian(double d);

/\* REQUIRES

 \*  Angle in degrees between 0 and 360.

 \* PROMISES

 \*  returns value is angle in radians.

 \*/

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\t\t\tt\t\t\td\n(deg)\t\t\t(sec)\t\t\t(m)\n";

    cout << std::fixed << std::setprecision(6); // format digits of table

    for(double i = 0; i <= 90; i +=5){

        cout << i << "\t\t" << Projectile\_travel\_time(i, v) << "\t\t" << Projectile\_travel\_distance(i, v) << "\n";

    }

}

double degree\_to\_radian(double d){

    return (d / 180 \* PI);

}

double Projectile\_travel\_time(double a, double v){

    return abs(2 \* v \* sin(degree\_to\_radian(a)) / G);

}

double Projectile\_travel\_distance(double a, double v){

    return abs(v \* v \* sin(degree\_to\_radian(2\*a)) / G);

}

## Code output:

QWE+RyanB@ryanb-pc /cygdrive/c/Users/RyanB/OneDrive - Quick Way Electrical (1999) Ltd/ENSF 694/Labs/ensf-694-a01

$ ./lab1exe\_A

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

## Part I

* No submission required

## Part II

A diagram of a diagram

Description automatically generated

# Exercise C

## lab1exe\_C.cpp

 /\*

 \*  lab1exe\_C.cpp

 \*  ENSF 694 Lab 1 Exercise C

 \*  Completed by: Ryan Baker

 \*  Development Date: June 29, 2024

 \*/

#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){

  \*minutes\_ptr = ms\_time/6000;

  \*seconds\_ptr = (double)(ms\_time % 6000) / 1000.0;

}

## lab1exe\_C.cpp output

QWE+RyanB@ryanb-pc /cygdrive/c/Users/RyanB/OneDrive - Quick Way Electrical (1999) Ltd/ENSF 694/Labs/ensf-694-a01

$ ./lab1exe\_C

Enter a time interval as an integer number of milliseconds: 7450

Doing conversion for input of 7450 milliseconds ...

That is equivalent to 1 minute(s) and 1.45 second(s).