A picture containing logo

Description automatically generated



# ENSF 614

# Advanced System Analysis and Software Design

## LAB 1

Author:

Steven Duong

(30022492)

Affiliation

Department of Electrical and Computer Engineering

University of Calgary

Calgary, Alberta

Lab Block: B01

Date of Report: Jan 20, 2023

# Exercise B

## C++ code to be executed:

/\*

\* File Name: lab1exe\_B.cpp

\* Assignment: ENSF 614 Lab 1 Exercise B

\* Lab Section: Lab B01

\* Created by: Mahmood Moussavi

\* Completed by: Steven Duong (30022492)

\* Submission Date: Jan 20, 2023

\*/

#include <iostream>

#include <cmath>

#include <iomanip>

**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);

**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 << "Invalid input. Bye...\n";

exit(1);

}

**while** (velocity < 0) {

cout << "\nPlease enter a positive number for velocity: ";

cin >> velocity;

**if** (!cin) {

cout << "Invalid input. Bye...";

exit(1);

}

}

create\_table(velocity);

**return** 0;

}

// Creates a table with 3 columns which represents the angle in degrees,

// time in seconds and distance in meters.

**void** create\_table(**double** v) {

**int** size = 90 / 5;

**int** angles\_d[size];

cout << "Angle\t\t" << "t\t\t\t" << "d" << endl;

cout << "(deg)\t\t" << "(sec)\t\t" << "(m)" << endl;

**for** (**int** i = 0; i <= size; i++) {

angles\_d[i] = 5 \* (i);

cout << fixed;

cout << setprecision(5);

cout << angles\_d[i] << "\t\t\t" << Projectile\_travel\_time(angles\_d[i], v) << "\t\t" <<

Projectile\_travel\_distance(angles\_d[i], v) << endl;

}

}

// Calculates the travel time for the projectile.

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

**double** rad = degree\_to\_radian(a);

**return** (2 \* v \* sin(rad)) / G;

}

// Calculates the travel distance for the projectile.

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

**double** rad = degree\_to\_radian(a);

**return** abs((pow(v, 2) / G) \* sin(2 \* rad));

}

// Converts degrees to radians.

**double** degree\_to\_radian(**double** d) {

**return** d \* (PI / 180);

}

## Program Output for Exercise B:

Text

Description automatically generated

# Exercise D

## AR Diagram for lab1\_exeD2 at point one:

Diagram

Description automatically generated

# Exercise E

## C++ Code to be executed:

/\*

\* File Name: lab1exe\_E.cpp

\* Assignment: ENSF 614 Lab 1 Exercise E2

\* Lab Section: Lab B01

\* Created by: Mahmood Moussavi

\* Completed by: Steven Duong (30022492)

\* Submission Date: Jan 20, 2023

\*/

#include <iostream>

#include <cmath>

**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;

// This allows the user to continuously input a number until the number is

// positive.

**while** (millisec < 0) {

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", millisec;

/\* 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) {

// Local variable mins.

**int** mins = floor(ms\_time / 60000);

// Modulus ensures that:

// 0 <= \*seconds\_ptr & \*seconds\_ptr < 60.0

// Local variable secs.

**double** secs = (ms\_time % 60000) / 1000.0;

// Ensures that the minutes and seconds pointers have

// values that add up to the total original ms\_time

// before re-assigning.

**if** ((mins \* 60000 + secs \* 1000) == ms\_time) {

\* minutes\_ptr = mins; // points to local variable mins

\* seconds\_ptr = secs; // points to local variable secs

}

}

## Program Output for Exercise E:

Text

Description automatically generated