Course: ENSF 614 – Fall 2023  
Lab #: Lab 1  
Instructor: Mahmood Moussavi  
Student Name: Satchytan Karalasingham  
Submission Date: September 20, 2023

Lab 1 Exercise B:

CODE:

/\*

\* Filename: lab1exe\_B.cpp

\* Assignment: ENSF 614 Lab 1, exercise B

\* Created by Mahmood Moussavi

\* Completed by: Satchytan Karalasingham

\* Submission Date: Sept 20, 2023

\*/

#include <iostream>

#include <cmath>

**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 << "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 (sec) Distance (m)" << endl;

cout.precision(6); // Set the precision for the output

**for** (**double** angle = 0; angle <= 90; angle += 5)

{

**double** radians = degree\_to\_radian(angle);

**double** time = Projectile\_travel\_time(radians, v);

**double** distance = Projectile\_travel\_distance(radians, v);

cout << angle << " " << time << " " << distance << endl;

}

}

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

{

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

}

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

{

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

}

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

{

**return** d \* (PI / 180.0);

}

OUTPUT:

**Please enter the velocity at which the projectile is launched (m/sec):** 10

**Angle (deg) Time (sec) Distance (m)**

**0 0 0**

**5 0.177869 1.77192**

**10 0.354384 3.49**

**15 0.528202 5.10204**

**20 0.698 6.55906**

**25 0.862486 7.81678**

**30 1.02041 8.83699**

**35 1.17056 9.5887**

**40 1.31181 10.0491**

**45 1.44308 10.2041**

**50 1.56336 10.0491**

**55 1.67174 9.5887**

**60 1.7674 8.83699**

**65 1.84961 7.81678**

**70 1.91774 6.55906**

**75 1.97128 5.10204**

**80 2.00981 3.49**

**85 2.03305 1.77192**

**90 2.04082 -4.18578e-09**

**Program ended with exit code: 0**

SCREENSHOT:

A screenshot of a computer

Description automatically generated

Lab 1 Exercise D Part 2:

CODE:

/\*

\* Filename: lab1exe\_D2.cpp

\* Assignment: ENSF 614 - Lab 1 - Execise D Part Two

\* Created by Mahmood Moussavi

\* Completed by: Satchytan Karalasingham

\* 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";

AR DIAGRAM:

A graph on a piece of paper

Description automatically generated

Lab 1 Exercise E:

CODE:

/\*

\* Filename: lab1exe\_E.cpp

\* Assignment: ENSF 614 Lab 1 Exercise E

\* Created by Mahmood Moussavi

\* Completed by: Satchytan Karalasingham

\* 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.

\*/

\*minutes\_ptr = ms\_time / (1000 \* 60);

ms\_time %= (1000 \* 60);

\*seconds\_ptr = **static\_cast**<**double**>(ms\_time) / 1000.0;

}

**int** main(**void**) {

**int** millisec;

**int** minutes;

**double** seconds;

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

cin >> millisec;

**if** (!cin) {

cout << "Unable to convert your input to an int.\n";

**return** 1;

}

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

// Call the time\_convert function

time\_convert(millisec, &minutes, &seconds);

cout << "That is equivalent to " << minutes << " minute(s) and " << seconds << " second(s).\n";

**return** 0;

}

OUTPUT:

**Enter a time interval as an integer number of milliseconds:** 850000

**Doing conversion for input of 850000 milliseconds ...**

**That is equivalent to 14 minute(s) and 10 second(s).**

**Program ended with exit code: 0**

SCREENSHOT:

A screenshot of a computer program

Description automatically generated