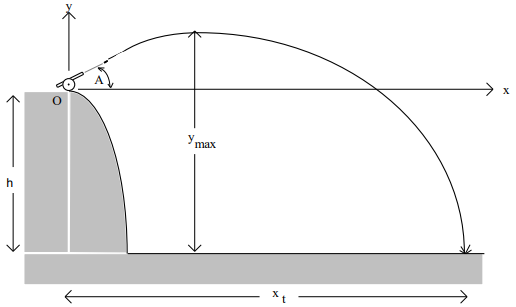
**Unit Project Three:**

Projectile Trajectory



**PROGRAM:**

**MYLIBRARY.H:**

The library header is 22 lines long

using namespace std;

//Function declarations

//Function to change all chars in a string to lowercase

string &**lowercase** (string &s);

//Function that takes each char in a string and puts it in a char

char **string2Char** (string);

//Function to convert degrees to radians if needed

double **deg2rad** (char c, double angle);

//Function for the Quadratic equation

double **quadratic** (double, double);

//Function to clear the screen and go back to the menu

void **goHome** ();

**MAIN:**

The main is 150 lines long

// \_\_\_\_ \_\_ \_\_

// / / \ / \

// / / /

// / \\_\_\_/ \\_\_\_/

//

// Date: 10/26/2021

// Name: David Vermaak

//Project Description: The main function of Unit Project 3: Projectile Trajectory

// Inputs: units, initial and final values, formatting values, etc.

// Outputs: a formatted table or a file

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream> //This header containing cout and cin

#include <conio.h> //This header declares getch which pauses until a key is pushed

#include <math.h> //This header allows for the use of more complex mathematical operators

#include <string> //This header enables string functions

#include <iomanip> //This header contains functions to better format output

#include <ctype.h> // This header contains functions to manipulate chars

#include <vector> //This header allows the use of vectors (better arrays)

#include <algorithm> //This header allows the use of algorithm functions

#include <numeric> //This header allows the use of numeric functions

#include <fstream> //This header allows the use of functions that edit files

#include "MyLibrary.h" //Personal library header

using namespace std; //introduces namespace std

const double g = 9.80665; //setting up a constant value for gravity

const double pi = acos(-1.0); //setting up a constant value for pi

int **main** ( ) //The Main function

{

string units, s2, option; //initializing variables

double height, hmax, v\_initial, v\_final, vx, vy, vy\_final, range, displacement, dx, dy, angle, time, timehmax, tempT;

int n;

char c;

vector <double> N;

do //do while loop to make the program repeatable

{

// display menu

cout << " Hello, this program calculates the parabolic trajectory of a projectile\n"

<< " Please enter your measurement system to continue:\n"

<< " Meters \n" << " Feet \n" << " Quit\n\n";

cin >> units;

**lowercase**(units); //calls the function: lowercase

if( units == "quit") return 0; //The quit condition of the menu and loop

else if (units == "meter" || units == "meters" || units == "feet" || units == "foot" )

{

cout << " Please enter your initial height:\n";

cin >> height;

cout << " Please enter your initial velocity:\n";

cin >> v\_initial;

cout << "Would you like to input your angles in radians or degrees? \n ";

cin >> s2;

cout << " Please enter your initial angle:\n";

cin >> angle;

cout << " Please enter the number of data points you want:\n";

cin >> n;

if (height < 0 || v\_initial <= 0 || n <5 || n > 100) //The failure condition

{

cout <<"Incorrect Input";

**goHome**(); //calls the function goHome

}

else

{

c = **string2Char**(s2); //calls the function string2char and stores to char c

angle = **deg2rad**(c, angle); //calls the function deg2rad and returns angle

time = **quadratic** ((v\_initial\*sin(angle)), height);

//time independent calculations

timehmax = (v\_initial\*sin(angle))/g;

hmax = height + ((pow(v\_initial,2)\*pow(sin(angle),2))/(2\*g));

vx = v\_initial\*cos(angle);

vy = v\_initial\*sin(angle);

vy\_final = v\_initial\*sin(angle)-(g\*time);

v\_final = sqrt(pow(vx,2)+pow(vy\_final,2));

//time-based calculations

dx = vx\*time;

dy = vy\*time - 0.5\*g\*pow(time,2);

displacement = sqrt(pow(dx,2)+pow(dy,2));

range = v\_initial \* cos(angle) \* time;

N.push\_back(0.0); //Time

N.push\_back(0.0); //initial position x

N.push\_back(height); //initial position y

N.push\_back(vy); //initial velocity y

N.push\_back(v\_initial); //initial velocity

for(int i=1; i < n; i++)

{

dx = (i/(n-1.0))\*displacement; //position x

tempT = dx/vx; //Time

N.push\_back(tempT);

N.push\_back(dx);

N.push\_back(height + vy\*tempT+0.5\*-g\*pow(tempT,2)); //position y

N.push\_back(vy-g\*tempT); //velocity y

N.push\_back(sqrt(pow(vx,2)+pow((vy-g\*tempT),2))); //velocity

}

cout << " Time of Flight: " << time << " seconds \n Displacement: " << displacement << " " << units << "\n Range: " << range << " " << units << "\n Maximum Height: " << hmax << " " << units << "\n Final Velocity: " << v\_final << " " << units << "/second\n\n ";

cout <<" Would you like the results Displayed or Stored to a file?\n\n Display or Store:";

cin >> option;

**lowercase**(option);

if (option == "display" || option == "displayed")

{

cout << "\n\nTime\t x("<<units<< ")\t y("<<units<< ")\tVelocity in y\t Velocity(" << units <<"/second)\n";

for (int a=0; a <= (N.size()-5); a+=5)

{

cout << left << fixed << setprecision(2) << setw(10) << N[a] << setw(15) << N[a+1] << setw(15) << N[a+2] << setw(18) << N[a+3] << setw(18) << N[a+4] <<endl;

}

**goHome**();

}

else if (option == "store" || option == "stored")

{

//open file

**ofstream** **outfile**("**File2.csv**");

**outfile** << "Time,x,y,Velocity in y,Velocity\n";

for (int a=0; a <= (N.size()-5); a+=5)

{

**outfile** << N[a] << "," << N[a+1] << "," << N[a+2] << "," << N[a+3] << "," << N[a+4] <<"\n";

}

outfile.close();

**goHome**();

}

}

}

else //The failure condition of the menu

{

cout << " Incorrect Unit\n\n "; //error message for an incorrect input

**goHome**();

}

} while (units != "quit" );

return 0;

}

**MYLIBRARY.CPP:**

The library is 71 lines long

#include <iostream> //This header containing cout and cin

#include <conio.h> //This header declares getch which pauses until a key is pushed

#include <math.h> //This header allows for the use of more complex mathematical operators

#include <string> //This header enables string functions

#include <iomanip> //This header contains functions to better format output

#include <ctype.h> //This header contains functions to manipulate chars

#include "MyLibrary.h" //Personal library header

using namespace std;

const double g = 9.80665; //setting up a constant value for gravity

const double pi = acos(-1.0); //setting up a constant value for pi

//Functions

//Function to change all chars in a string to lowercase

string &**lowercase** (string &S1)

{

for (int i = 0; i<S1.length(); i++)

{ S1[i] = tolower(S1[i]);} //uses tolower in a for loop

return S1;

}

//Function that takes each char in a string and puts it in a char

void **String2Char** (string y, char& C0, char& C1, char& C2)

{

C0 = y[0];

C1 = y[1];

C2 = y[2];

}

//Function to choose and change angles

double deg2rad (char c, double angle)

{

if (c == 'd' || 'D') //if user wants degrees will convert radians to degrees

{

angle = angle \* ( pi/180.0);

if (angle < 90 && angle > 0) return angle;

else cout << "Error\n Angle out of Bounds\n";

goHome();

}

else return angle;

}

//Function for the Quadratic equation

double **quadratic** (double b, double c)

{ double a = 0.5\*(-g), t1, t2;

t1 = (-b + sqrt(b\*b - 4.0\*a\*c))/(2.0\*a);

t2 = (-b - sqrt(b\*b - 4.0\*a\*c))/(2.0\*a);

if (t1>0) return t1;

else return t2;

}

//Function to clear the screen and go back to the menu

void **goHome** ()

{ cout << "\n\n To return to the menu press enter";

getch(); //pauses until a key is pressed

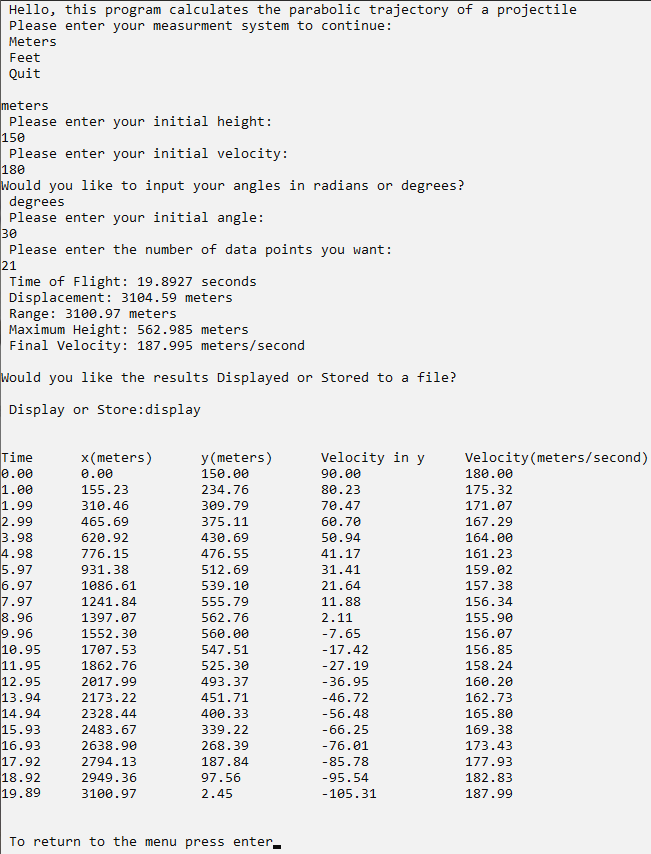
system("cls");

}

* Test Results: Run the program for the following test cases:

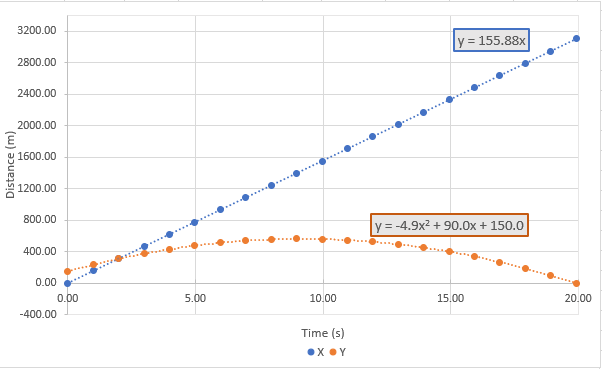
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cases** | **Vo (m/s)** | **A (degrees)** | **h (m)** | **N(points)** |
| 1&2 | 180 | 30 | 150 | 21 |
| 3&4 | 300 | 30 | 160 | 93 |

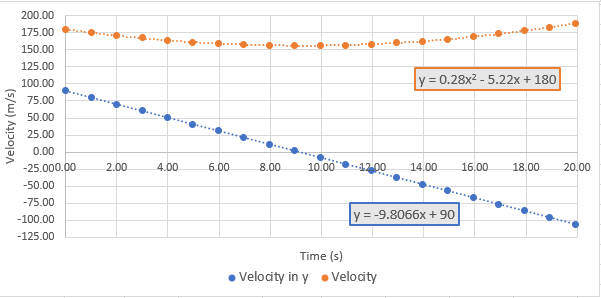
**Case 1:**



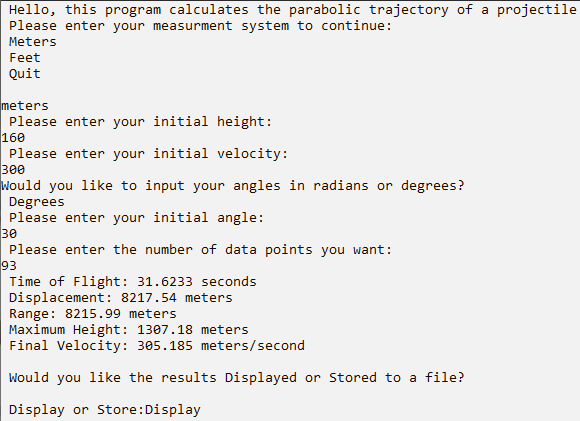
**Case 2:**

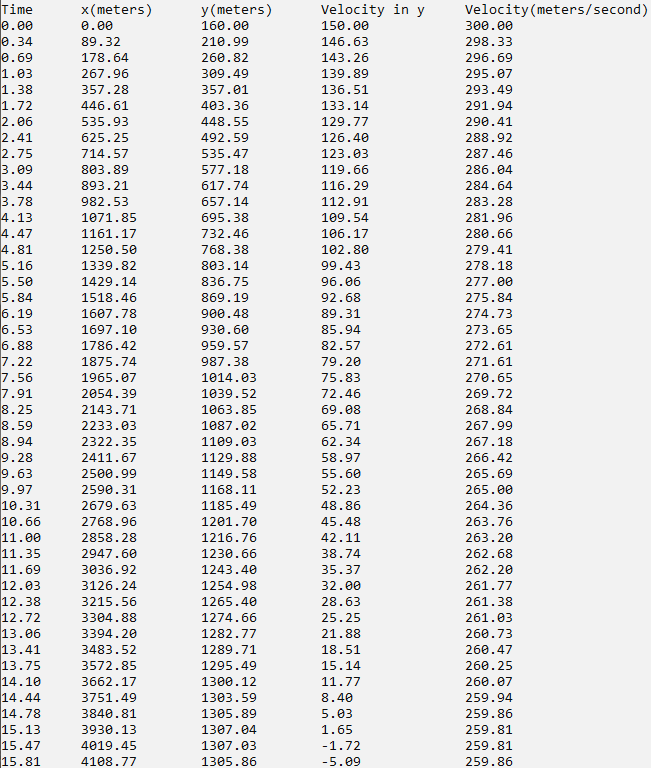
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time | x | y | Velocity in y | Velocity |
| 0.00 | 0.00 | 150.00 | 90.00 | 180.00 |
| 1.00 | 155.23 | 234.76 | 80.23 | 175.32 |
| 1.99 | 310.46 | 309.80 | 70.47 | 171.07 |
| 2.99 | 465.69 | 375.11 | 60.70 | 167.29 |
| 3.98 | 620.92 | 430.69 | 50.94 | 164.00 |
| 4.98 | 776.15 | 476.55 | 41.17 | 161.23 |
| 5.97 | 931.38 | 512.69 | 31.41 | 159.02 |
| 6.97 | 1086.61 | 539.11 | 21.64 | 157.38 |
| 7.97 | 1241.84 | 555.79 | 11.88 | 156.34 |
| 8.96 | 1397.07 | 562.76 | 2.11 | 155.90 |
| 9.96 | 1552.30 | 560.00 | -7.65 | 156.07 |
| 10.95 | 1707.53 | 547.51 | -17.42 | 156.86 |
| 11.95 | 1862.76 | 525.30 | -27.19 | 158.24 |
| 12.95 | 2017.99 | 493.37 | -36.95 | 160.20 |
| 13.94 | 2173.22 | 451.71 | -46.72 | 162.73 |
| 14.94 | 2328.44 | 400.33 | -56.48 | 165.80 |
| 15.93 | 2483.67 | 339.22 | -66.25 | 169.38 |
| 16.93 | 2638.90 | 268.39 | -76.01 | 173.43 |
| 17.92 | 2794.13 | 187.84 | -85.78 | 177.93 |
| 18.92 | 2949.36 | 97.56 | -95.54 | 182.84 |

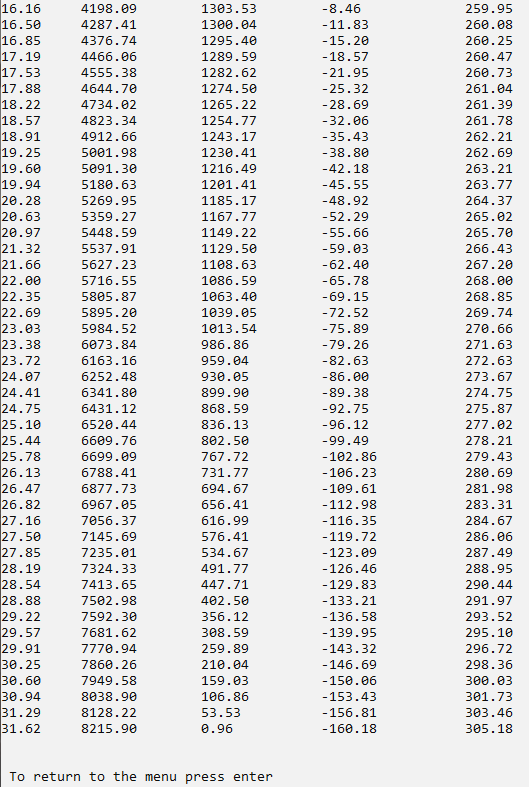
****

****

**Case 3:**

****

****



**Case 4:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time | x | y | Velocity in y | Velocity |
| 0 | 0 | 160 | 150 | 300 |
| 0.34 | 89.32 | 210.99 | 146.63 | 298.33 |
| 0.69 | 178.64 | 260.82 | 143.26 | 296.69 |
| 1.03 | 267.96 | 309.49 | 139.89 | 295.07 |
| 1.38 | 357.29 | 357.01 | 136.51 | 293.49 |
| 1.72 | 446.61 | 403.36 | 133.14 | 291.94 |
| 2.06 | 535.93 | 448.55 | 129.77 | 290.41 |
| 2.41 | 625.25 | 492.59 | 126.40 | 288.92 |
| 2.75 | 714.57 | 535.47 | 123.03 | 287.47 |
| 3.09 | 803.89 | 577.18 | 119.66 | 286.04 |
| 3.44 | 893.21 | 617.74 | 116.29 | 284.64 |
| 3.78 | 982.53 | 657.14 | 112.91 | 283.28 |
| 4.13 | 1071.85 | 695.38 | 109.54 | 281.96 |
| 4.47 | 1161.17 | 732.46 | 106.17 | 280.66 |
| 4.81 | 1250.50 | 768.38 | 102.80 | 279.41 |
| 5.16 | 1339.82 | 803.14 | 99.43 | 278.18 |
| 5.50 | 1429.14 | 836.75 | 96.06 | 277.00 |
| 5.84 | 1518.46 | 869.19 | 92.68 | 275.85 |
| 6.19 | 1607.78 | 900.48 | 89.31 | 274.73 |
| 6.53 | 1697.10 | 930.60 | 85.94 | 273.65 |
| 6.88 | 1786.42 | 959.57 | 82.57 | 272.61 |
| 7.22 | 1875.74 | 987.38 | 79.20 | 271.61 |
| 7.56 | 1965.07 | 1014.03 | 75.83 | 270.65 |
| 7.91 | 2054.39 | 1039.52 | 72.46 | 269.72 |
| 8.25 | 2143.71 | 1063.85 | 69.08 | 268.84 |
| 8.59 | 2233.03 | 1087.02 | 65.71 | 267.99 |
| 8.94 | 2322.35 | 1109.03 | 62.34 | 267.18 |
| 9.28 | 2411.67 | 1129.88 | 58.97 | 266.42 |
| 9.63 | 2500.99 | 1149.58 | 55.60 | 265.69 |
| 9.97 | 2590.31 | 1168.11 | 52.23 | 265.01 |
| 10.31 | 2679.63 | 1185.49 | 48.86 | 264.36 |
| 10.66 | 2768.96 | 1201.70 | 45.48 | 263.76 |
| 11.00 | 2858.28 | 1216.76 | 42.11 | 263.20 |
| 11.35 | 2947.60 | 1230.66 | 38.74 | 262.68 |
| 11.69 | 3036.92 | 1243.40 | 35.37 | 262.20 |
| 12.03 | 3126.24 | 1254.98 | 32.00 | 261.77 |
| 12.38 | 3215.56 | 1265.40 | 28.63 | 261.38 |
| 12.72 | 3304.88 | 1274.66 | 25.25 | 261.03 |
| 13.06 | 3394.20 | 1282.77 | 21.88 | 260.73 |
| 13.41 | 3483.52 | 1289.71 | 18.51 | 260.47 |
| 13.75 | 3572.85 | 1295.49 | 15.14 | 260.25 |
| 14.10 | 3662.17 | 1300.12 | 11.77 | 260.07 |
| 14.44 | 3751.49 | 1303.59 | 8.40 | 259.94 |
| 14.78 | 3840.81 | 1305.89 | 5.03 | 259.86 |
| 15.13 | 3930.13 | 1307.04 | 1.65 | 259.81 |
| 15.47 | 4019.45 | 1307.03 | -1.72 | 259.81 |
| 15.81 | 4108.77 | 1305.86 | -5.09 | 259.86 |
| 16.16 | 4198.09 | 1303.53 | -8.46 | 259.95 |
| 16.50 | 4287.41 | 1300.04 | -11.83 | 260.08 |
| 16.85 | 4376.74 | 1295.40 | -15.20 | 260.25 |
| 17.19 | 4466.06 | 1289.59 | -18.57 | 260.47 |
| 17.53 | 4555.38 | 1282.62 | -21.95 | 260.73 |
| 17.88 | 4644.70 | 1274.50 | -25.32 | 261.04 |
| 18.22 | 4734.02 | 1265.22 | -28.69 | 261.39 |
| 18.57 | 4823.34 | 1254.77 | -32.06 | 261.78 |
| 18.91 | 4912.66 | 1243.17 | -35.43 | 262.21 |
| 19.25 | 5001.98 | 1230.41 | -38.80 | 262.69 |
| 19.60 | 5091.30 | 1216.49 | -42.18 | 263.21 |
| 19.94 | 5180.63 | 1201.41 | -45.55 | 263.77 |
| 20.28 | 5269.95 | 1185.17 | -48.92 | 264.37 |
| 20.63 | 5359.27 | 1167.77 | -52.29 | 265.02 |
| 20.97 | 5448.59 | 1149.22 | -55.66 | 265.70 |
| 21.32 | 5537.91 | 1129.50 | -59.03 | 266.43 |
| 21.66 | 5627.23 | 1108.63 | -62.40 | 267.20 |
| 22.00 | 5716.55 | 1086.59 | -65.78 | 268.01 |
| 22.35 | 5805.87 | 1063.40 | -69.15 | 268.85 |
| 22.69 | 5895.20 | 1039.05 | -72.52 | 269.74 |
| 23.03 | 5984.52 | 1013.54 | -75.89 | 270.67 |
| 23.38 | 6073.84 | 986.87 | -79.26 | 271.63 |
| 23.72 | 6163.16 | 959.04 | -82.63 | 272.63 |
| 24.07 | 6252.48 | 930.05 | -86.00 | 273.67 |
| 24.41 | 6341.80 | 899.90 | -89.38 | 274.75 |
| 24.75 | 6431.12 | 868.59 | -92.75 | 275.87 |
| 25.10 | 6520.44 | 836.13 | -96.12 | 277.02 |
| 25.44 | 6609.76 | 802.50 | -99.49 | 278.21 |
| 25.78 | 6699.09 | 767.72 | -102.86 | 279.43 |
| 26.13 | 6788.41 | 731.77 | -106.23 | 280.69 |
| 26.47 | 6877.73 | 694.67 | -109.61 | 281.98 |
| 26.82 | 6967.05 | 656.41 | -112.98 | 283.31 |
| 27.16 | 7056.37 | 616.99 | -116.35 | 284.67 |
| 27.50 | 7145.69 | 576.41 | -119.72 | 286.06 |
| 27.85 | 7235.01 | 534.67 | -123.09 | 287.49 |
| 28.19 | 7324.33 | 491.77 | -126.46 | 288.95 |
| 28.54 | 7413.65 | 447.71 | -129.83 | 290.44 |
| 28.88 | 7502.98 | 402.50 | -133.21 | 291.97 |
| 29.22 | 7592.30 | 356.12 | -136.58 | 293.52 |
| 29.57 | 7681.62 | 308.59 | -139.95 | 295.10 |
| 29.91 | 7770.94 | 259.89 | -143.32 | 296.72 |
| 30.25 | 7860.26 | 210.04 | -146.69 | 298.36 |
| 30.60 | 7949.58 | 159.03 | -150.06 | 300.03 |
| 30.94 | 8038.90 | 106.86 | -153.44 | 301.73 |
| 31.29 | 8128.22 | 53.53 | -156.81 | 303.46 |
| 31.63 | 8215.90 | 0.96 | -160.18 | 305.18 |

Program performance:

*The program’s performance was adequate. It compiled in 4.02 seconds and it is possible to run through and quit in under 30 seconds, with no noticeable lag. The UI, while barebones, conveys the information to the user quickly and adequately*.

Extra credit features include:

1. Allowed the user to select the units that he or she wishes to work with (e.g., Vo could be in m/s or ft/s. The output prints the appropriate units.
2. Instead of using arrays, used the vector class to store the values of t, x, and y. Specified the vector size to be N after the user entered the value of N.
3. Displayed: Time of Flight, Maximum Height, Displacement, Range, and Final Velocity.
4. Displayed in the list and file: Time, X position, Y position, Velocity in Y, and Velocity
5. Had additional graphs with X position, Y position, Velocity in Y, and Velocity (with respect to time.)

Potential improvements: