# Tarea 2: Ejercicios de Programación

*Andrés Mateus Vargas Hernández*

A continación se presenta la tarea que se debía realizar, existen líneas demasiado largas en el código fuente así que no están dentro de las margenes, modifiqué las margenes ajustadas al mínimo y sin embargo no se muestran bien, las dudas que me surgieron tras elaborar los ejercicios se muestran en los códigos.

## 1. Caída no tan libre

El programa provee de manera simultanea los datos para el movimiento de caida de un cuerpo en tierra $\left(g = 9{,}79\dfrac{m}{s^2}\right)$ con y sin fricción proporcionada por el fluido donde se desplaza, debido al crecimiento exponencial de la función $\cosh(x)$ para valores elevados de su argumento $\cosh\left(\sqrt{\frac{C\rho\pi R^2}{2mg}}gt\right)$ el programa determina un valor para la coordenada $y$ de $-\infty$ lo que termina la ejecución arrojando un tiempo de vuelo errado, tenga en cuenta que en cada una de las figuras la función *y_no_air(t)* representa la solución sin aplicar la fuerza resistiva a la esfera.

### 1.1. Demostración

Ecuación planteada:

$$\frac{d^2y}{dt^2} = -g + \frac{C\rho A}{2m}\left(\frac{dy}{dt}\right)^2 \tag{1}$$

Sí:

$$y = y_0 + \frac{1}{\gamma^2 g}\ln\left(\frac{\sqrt{1+(\gamma v_0)^2}}{\cosh(\gamma g t)}\right) \tag{2}$$

Demostrar que la ecuación 2 es solución a la ecuación diferencial 1 :

$$\frac{dy}{dt} = -\frac{[1/(\gamma)]\sinh(\gamma g t)}{\cosh(\gamma g t)}$$

y:

$$\frac{d^2y}{dt^2} = \frac{g\sinh^2(\gamma g t)}{\cosh^2(\gamma g t)} - g$$

De este modo, la ecuación 1 realiza la siguiente pregunta:

- ¿ Es $\dfrac{g\sinh^2(\gamma g t)}{\cosh^2(\gamma g t)} - g$ equivalente a $-g + \dfrac{C\rho A}{2m}\left(\dfrac{[1/(\gamma)]\sinh(\gamma g t)}{\cosh(\gamma g t)}\right)^2$?

$$\frac{g\sinh^2(\gamma g t)}{\cosh^2(\gamma g t)} - g = -g + \frac{C\rho A}{2m}\left(\frac{[1/(\gamma)]\sinh(\gamma g t)}{\cosh(\gamma g t)}\right)^2$$

Sumando $g$ a la igualdad:

$$\frac{g\sinh^2(\gamma g t)}{\cosh^2(\gamma g t)} = \frac{C\rho A}{2m}\left(\frac{[1/(\gamma)]\sinh(\gamma g t)}{\cosh(\gamma g t)}\right)^2$$

Multiplicando por ${\cosh^2(\gamma g t)}/{\sinh^2(\gamma g t)}$:

$$g = \frac{C\rho A}{2m}\left(1/\gamma^2\right)$$

Finalmente, se concluye que 2 es solución a 1 siempre que:

$$\gamma = \sqrt{\frac{C\rho A}{2mg}}$$

## 1.2. Variación de la constante C.

La constante $C$ de acuerdo con 1 establece qué tanto se aleja la solución del movimiento de caída libre sin fuerza resistiva al movimiento donde el cuerpo se mueve en un fluido que ofrece una fuerza de fricción proporcional al cuadrado de la rapidez, A continuación se muestran los resultados obtenidos para diferentes valores de $C$:

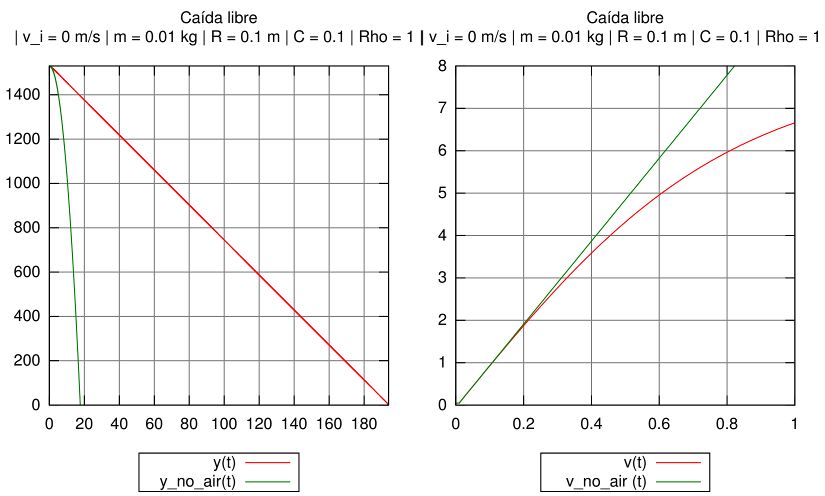


Fig. 1: Gráficas de $v(t)$ y $y(t)$ para $C = 0{,}1$ SI.

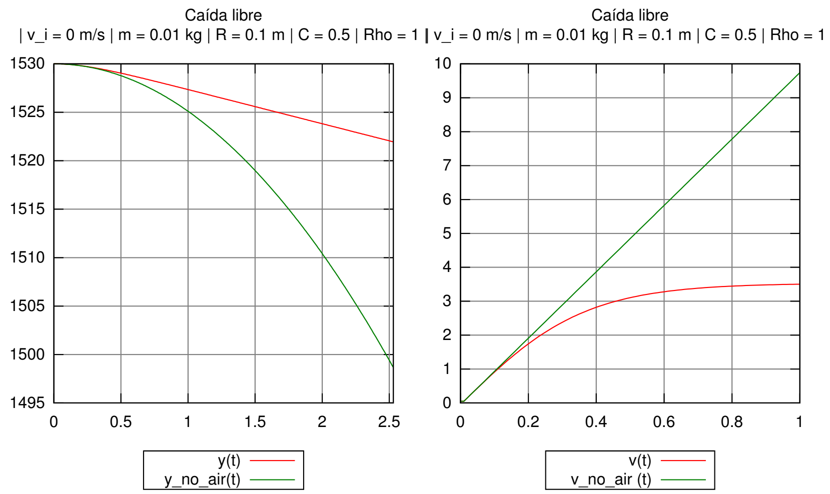


Fig. 2: Gráficas de $v(t)$ y $y(t)$ para $C = 0{,}5$ SI.

En la figura anterior se evidencia que la solución queda indefinida ya que la función $\cosh(x)$ crece al infinito de manera exponencial, se necesitaría mayor precisión, se realizó la definición de una variable que contenía la evaluación de ésta función, declarada como tipo *long double* sin tener cambio alguno por lo que se eliminó.

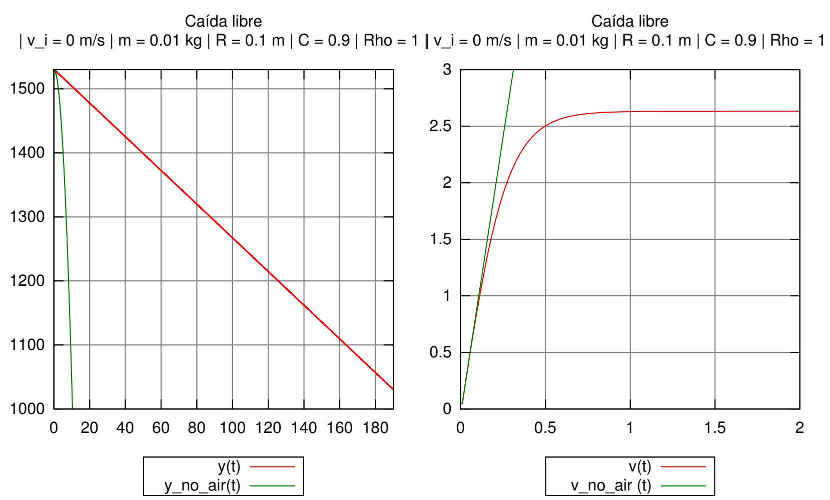


Fig. 3: Gráficas de $v(t)$ y $y(t)$ para $C = 0{,}9$ SI.

En la figura anterior se evidencia nuevamente que la solución queda indefinida ya que la función $\cosh(x)$ crece al infinito de manera exponencial, también se puede evidenciar la validez de la aproximación de *Stokes* donde se afirma que luego de un corto intervalo de tiempo la rapidez de un cuerpo que cae sumergido en un fluido permanece constante.

## 1.3. Variación del radio de la esfera

A continuación se muestran los resultados obtenidos para diferentes valores de $R$, el radio de la esfera:

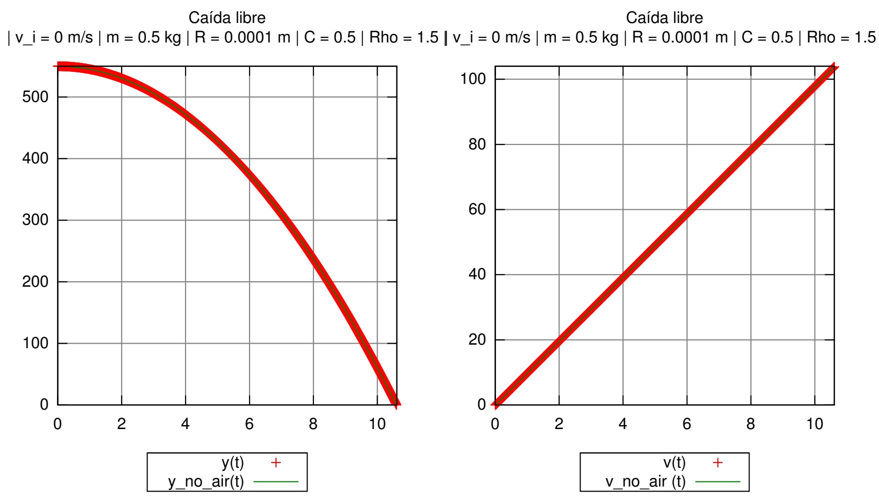


Fig. 4: Gráficas de $v(t)$ y $y(t)$ para $R = 1 * 10^{-4} m$ .

En la figura anterior es posible observar como para radios lo suficientemente pequeños la diferencia entre la caída con y sin resistencia del fluido de densidad $\rho = 1{,}5$ S.I. es prácticamente nula, ya que las curvas para los dos movimientos se superponen.

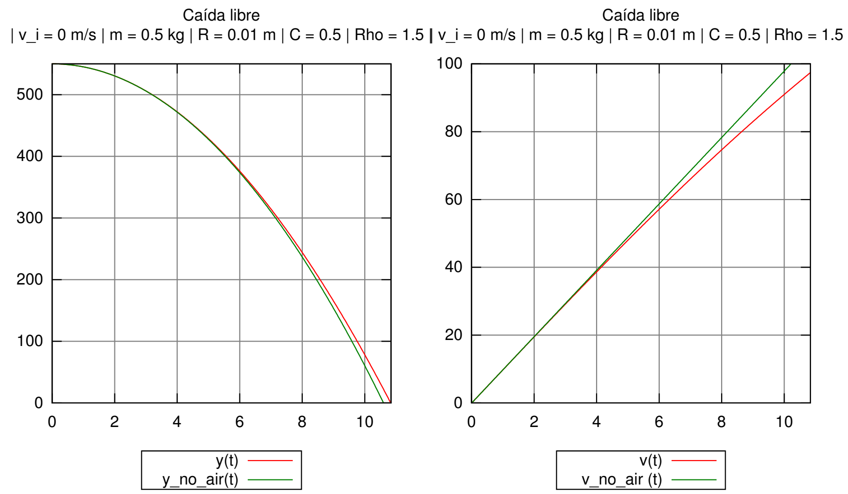


Fig. 5: Gráficas de $v(t)$ y $y(t)$ para $R = 0{,}01m$ .

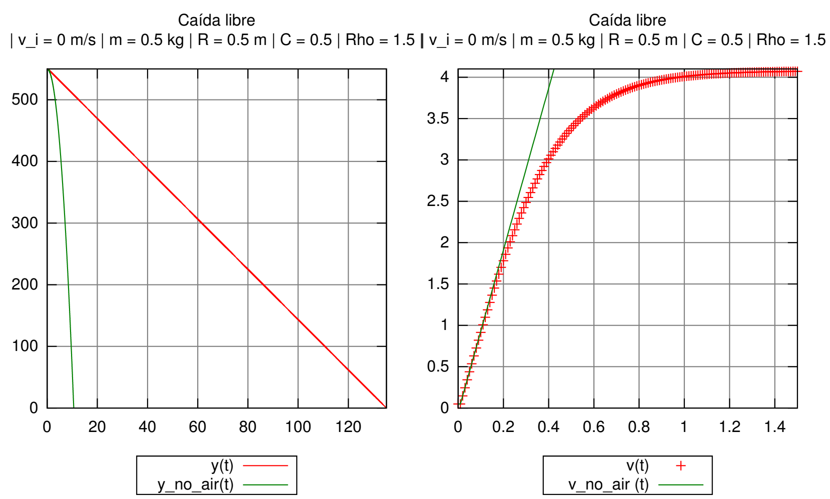


Fig. 6: Gráficas de $v(t)$ y $y(t)$ para $R = 0{,}5m$ .

A partir de las tres figuras anteriores y de los datos suministrados en los archivos anexos es posible afirmar que a medida que se aumenta el radio $R$ de la esfera, el tiempo que dura en caer aumenta de manera no lineal, observe la siguiente tabla:

| $R(m)$ | Tiempo de vuelo (s) |
|---|---|
| $1 * 10^{-4}$ | 10,61 |
| $1 * 10^{-2}$ | 10,84 |
| 0,5 | 135,2 |

Tab. 1: Tiempo de Vuelo vs Radio (Ver datos anexos)

## 1.4. Variación de la masa de la esfera

A continuación se muestran los resultados obtenidos para deiferentes valores de $M$, la masa de la esfera:

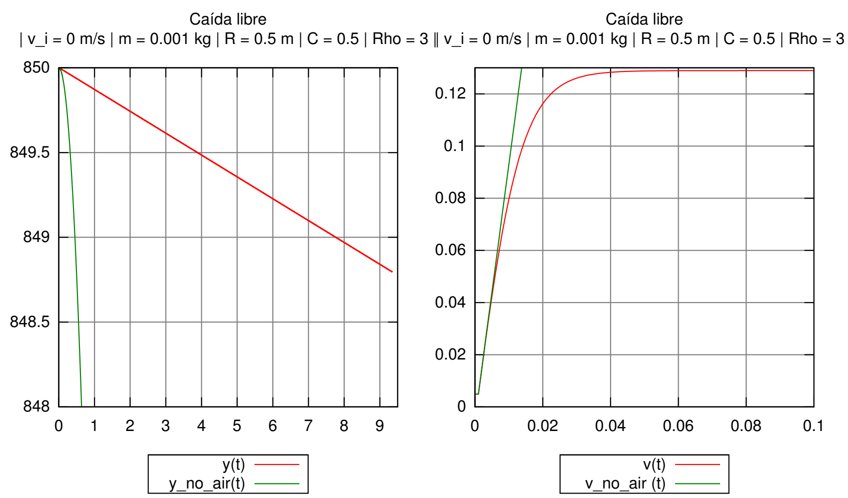


Fig. 7: Gráficas de $v(t)$ y $y(t)$ para $M = 0{,}001kg$ .

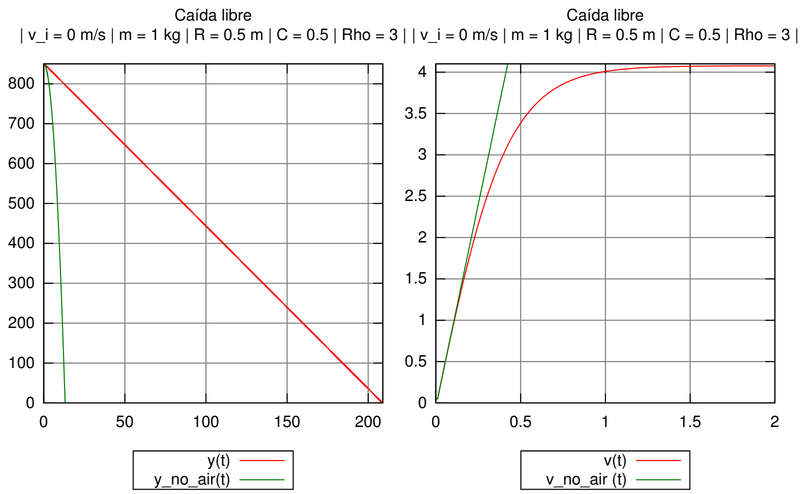


Fig. 8: Gráficas de $v(t)$ y $y(t)$ para $M = 1kg$ .

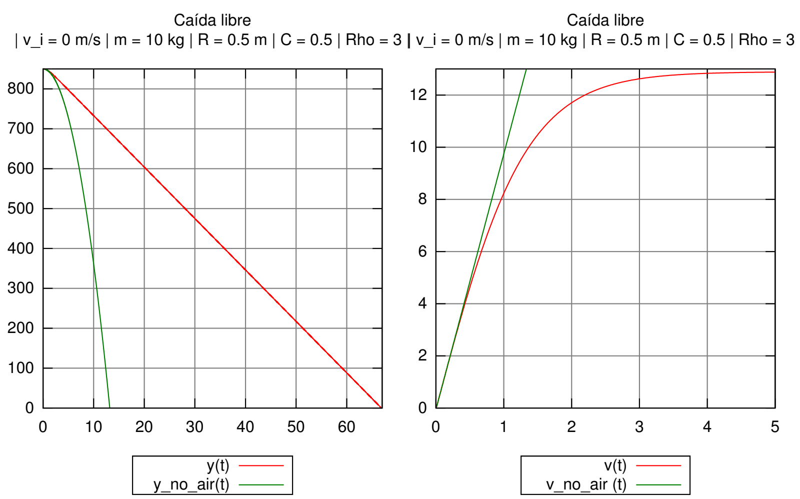


Fig. 9: Gráficas de $v(t)$ y $y(t)$ para $M = 10kg$ .

El tiempo de vuelo decrece a medida que aumenta la masa $M$ de la esfera, como se observa en la siguiente tabla (ver archivos de datos anexos).

| $M(kg)$ | Tiempo de vuelo (s) |
|---|---|
| $1 * 10^{-3}$ | 208,79 |
| 1 | 66,85 |
| 10 | 9,356 |

Tab. 2: Tiempo de Vuelo vs Radio (Ver datos anexos)

## 1.5. Código fuente

El siguiente es el código fuente de la aplicación, se presentan unas dudas al respecto:

```
/*
    Genera datos para la caída vertical de un cuerpo en un fluido

    CHANGELOG:

    2011-09-17:   Se agregaron funciones que permiten hacer la
            comparación entre la caida con y sin fricción
            tanto la posición en $y como la rapidez;

    El archivo generado será "data_friction.dat" y contendrá las
    siguientes columnas:
        * Tiempo
        * $y con resistencia del fluido
        * $v con resistencia del fluido
        * $y sin resistencia del fluido
        * $v sin resistencia del fluido
    Nota:

    Por comodidad se graficarán los valores absolutos de la
    rapidez.

    Preguntas:

    * fstream.open no crea el archivo si no existe, ¿Cómo se
    soluciona esto?
*/

#include <iostream>
#include <fstream>
#include <math.h>
#include <stdlib.h>

#define pi 3.14159
#define g 9.79
```

```
using namespace std;

//Declaro globalmente las condiciones iniciales
double y_i;
double v_i;
double delta_t;

double y_noair (int i) {
    //Esta función calcula el valor de y \
    para el caso en que no existe resistencia del fluido
    double y_na; //y no air
    y_na = y_i + v_i*(i*delta_t) - 0.5*g*pow((i*delta_t),2);
    return y_na;
}

double v_noair (int i) {
    double v_na;
    v_na = ((y_i + v_i*(i*delta_t) - 0.5*g*pow((i*delta_t),2))-\
    (y_i + v_i*((i-1)*delta_t) - 0.5*g*pow(((i-1)*delta_t),2)))/delta_t;
    return v_na;
}

int main () {

     double y;

    double m;
    double R;
    double rho;
    double C;

    double v;

    double gamma;

    int j=1; //Contador

    cout << "Se solicitarán los datos necesarios para resolver el \
    problema \nPor favor, introduzca las variales en el sistema \
    internacional de medidas (SI).\n";


    cout << "Introduzca el intervalo de tiempo: ";
    cin >> delta_t;
    cout << "Introduzca la coordenada Y inicial: ";
```

```
cin >> y_i;
cout << "Introduzca la rapidez inicial: ";
cin >> v_i;
cout << "Introduzca la masa de la esféra: ";
cin >> m;
cout << "Introduzca el radio de la esféra: ";
cin >> R;
cout << "Introduzca la constante C: ";
cin >> C;
cout << "Intruduzca la densidad del fluido (Rho) : ";
cin >> rho;


fstream f_data;

/*Aquí va la cuestión de mi pregunta
  Debí lanzar un comando que me creara el archivo
  para después abrirlo*/

system("echo '# | t | y | v | y_na | v_na |'> data_friction.dat");

f_data.open ("data_friction.dat");

f_data << "#Los valores asignados son los siguientes: \n";
f_data << "# | delta_t = " << delta_t << " s | y_i = " << y_i \
<< " m | v_i = " << v_i << " m/s | m = " << m << " kg | R = " << \
R << " m | C = " << C << " | Rho = " << rho << endl;

for ( int i = 0; i < j ; i++) {
    //se sustituye el gasto de memoria\
    por el trabajo de procesador\
    lo cual no es muy inteligente

    gamma = sqrt((C*rho*pi*(pow(R,2)))/(2*m*g));
    //Calculo y
    y = y_i + (1/((pow(gamma,2))*g))*(log((sqrt\
    (1+(pow((gamma*v_i),2))))/(cosh(gamma*g*(i*delta_t)))));

    //Calculo V mediante el Método de Euler \
    v aprx = (y2 - y1)/delta_t
    v = (y - (y_i + (1/((pow(gamma,2))*g))*(log((sqrt(1+\
    (pow((gamma*v_i),2))))/(cosh(gamma*g*(i-1)*delta_t))))))/(delta_t);

    //Escribo los datos en el archivo incluyendo cómo \
    variaría y si no hubiera fluido
    f_data << i*delta_t << "\t" << y << "\t" << v << "\t" <<\
```

```
         y_noair(i) << "\t" << v_noair(i) <<endl;

        //Continúa el ciclo mientras que $y permanezca mayor que cero
        if ( y > 0 ) {
            j++;
            cout << "Se han tomado: " << j << " datos.\n";
        }
        else {
            cout << "La esfera tardó: " << i*delta_t << \
            " s en llegar al suelo.\n";
            f_data << "#La esfera tardó: " << i*delta_t << \
            " s en llegar al suelo." << endl;
        }
    }

    f_data.close();

    return 0;

}
```

## 2. Números enteros

El siguiente programa realiza diversas operaciones con números enteros introducidos por el usuario.

```
/*
En éste código se visualizará la utilidad de in-
cluir <using namespace std;> después de incluir
las cabeceras.
*/

#include <iostream>

void minus(int a, int b, int c) {

    if ( a < b && a < c ) {
        std::cout << "El número más \
        pequeño es: " << a << std::endl;
    }

    if ( b < a && b < c ) {
        std::cout << "El número más \
        pequeño es: " << b << std::endl;
    }
```

```
    if ( c < a && c < b ) {
        std::cout << "El número más \
        pequeño es: " << c << std::endl;
    }

}

void plus (int a, int b, int c) {

    if ( a > b && a > c ) {
        std::cout << "El número más \
        grande es: " << a << std::endl;
    }

    if ( b > a && b > c ) {
        std::cout << "El número más \
        grande es: " << b << std::endl;
    }

    if ( c > a && c > b ) {
        std::cout << "El número más \
        grande es: " << c << std::endl;
    }
}

void main () {

    int a,b,c;

    std::cout << "Por favor, Ingrese tres \
    números enteros diferentes:" ;
    std::cin >> a >> b >> c ;

    std::cout << "La suma es: " << a+b+c << std::endl;
    std::cout << "El promedio es: " << (float)(a+b+c)/3.0 << std::endl;
    std::cout << "El producto es: " << a*b*c << std::endl;

    minus(a,b,c);
    plus(a,b,c);

}
```

# 3. Ecuación Cuadrática

El siguiente programa obtiene soluciones reales de la ecuación cuadrática:

## 3.1. Código fuente

```
/* Resolviendo cuadráticas al estilo CPP */

#include <iostream>
#include <math.h> //La cabecera sabia
using namespace std;

int givemearesult(int a, int b, int c){
    double x;

    x = (-1*b + sqrt((b*b)-4*a*c))/(2*double(a));

    cout << "X + = " << x << endl;
    cout << "Probemos: \n";
    cout << (((double)a*x*x)+((double)b*x)+
    (double)c) << " ¿Es próximo a cero?, Sí!" << endl;
    x = (-1*b - sqrt((b*b)-4*a*c))/(2*double(a));
    cout << "X - = " << x << endl;
    cout << "Probemos: \n";
    cout << (((double)a*x*x)+((double)b*x)+\
    (double)c) << " ¿Es próximo a cero?, Sí!" << endl;
}

int main() {

    int a,b,c;

    cout << "Por favor, ingrese los valores \
    de a, b, y c respectivamente:\n" ;
    cout << "a: " ;
    cin >> a ;
    cout << "b: " ;
    cin >> b ;
    cout << "c: " ;
    cin >> c ;

    if ( (b*b-4*a*c)< 0 ) {
        cout << "No, Aun no me han enseñado \
        números complejos,\n Adios!\n";
        return -1;
    }
```

```
    cout << "La ecuación cuadrática a resolver es:\n" ;
    cout << a << " * (X^2) + " << b << " * X + " << c << " = 0\n";
    cout << "La solución de la cuadrática es: \n" ;

    givemearesult(a,b,c);

    return 0;
}
```

## 4. Conclusiones:

- Valores lo suficientemente grandes para el argumento de cosh $\left(\sqrt{\frac{C\rho\pi R^2}{2mg}}gt\right)$ no se pueden resolver a través del programa realizado para la caída libre, por ejemplo el valor 710,70 en dicho argumento genera que la función devuelva un valor infinito, se realizó el mismo ejercicio a través de octave sin tener éxito.
- Resulta conveniente ejecutar la sentencia *using namespace std;* luego de incluir las cabeceras de los programas en C++ como se pudo observar en el primer programa.

## 5. Dudas

- ¿Cómo a través de *fstream.open* se puede crear el archivo a abrir cuando éste no existe?
- ¿Cómo evito que los códigos incluidos mediante *verbatim* burlen las margenes sin tener que modificar la longitud de las líneas?

## 6. Archivos necesarios para leer el documento (ANEXOS)

- *data_friction*.dat*

## 7. Otros archivos incluidos:

- *gplot.gp* Script que genera las gráficas de gnuplot.
- *friction* Ejecutable del ejercicio 3.