C++ Mathematical Calculations: Falling Bodies

Suppose an object is dropped from a point at a known distance above the ground and allowed to fall without any further interference; for example, a skydiver leaps from an airplane flying at an altitude of 5000 feet. An object that falls in this manner is said to be in free fall.

The altitude of the object, its distance above the ground, will change as time passes. The velocity at which the object falls will also change (increase, actually) as time passes. Let *H* represent the altitude of the object and *V* represent its velocity. It's also clear, intuitively, that the altitude of the object depends on how its velocity changes; in fact, there's a nice mathematical relationship between the altitude and the velocity involving a definite integral. But we're interested only in the velocity of a free-falling object.

It turns out that the velocity of a free-falling object is modeled by the formula

$$V = \sqrt{\frac{m \times g}{k}} \tanh\left(\sqrt{\frac{g \times k}{m}} \times t \right)$$

where:

- *m* is the mass of the object
- *g* is the acceleration due to gravity
- *t* is the number of seconds the object has been in free fall
- *k* is the drag coeffiient, determined by the shape of the object and the density of the air through which it falls
- tanh( ) is the hyperbolic tangent function

For simplicity, we will call k the drag constant. We also assume that the distance the object falls is small enough that the density of air doesn't change appreciably from its starting point down to the last point of interest.

For example, suppose a skydiver weighing 130 lb exits her plane and falls for 3 seconds. Her velocity could be computed as follows. First of all, weight equals mass times *g*. Second, in English units, *g* is about 32.2. So, her mass m would be about 130/32.2 ≈ 4.037. Applying the given formula:

$$V = \sqrt{\frac{4.037 \times 32.2}{0.005}} \tanh\left(\sqrt{\frac{32.2 \times 0.005}{4.037}} \times 3 \right) \approx 161.24 \tanh(0.599) \approx 86.48$$

giving her a velocity of about 86.48 feet per second (or about 58.96 miles per hour), three seconds after her jump began.

This problem is adapted from an exercise in *Calculus*, 9th Ed., by Thomas and Finney, page 527. It's not necessary, or even useful to understand the derivation of the formula given above to complete this assignment. However, if you're interested, see the Appendix for Project 2 on the course website.

In this project, you will complete a program that reads the data needed to apply the given formula to calculate the velocity over a given number of seconds. The partial implementation given below manages reading the input data and writing some of the output data. Your task is to implement the necessary calculations and complete the output. Be sure to follow the specification below carefully. As was the case with the previous project, the given code uses some C++ features that have not been covered yet; the comments should make the logic reasonably clear.

To perform these calculations, you will need to use the formula given above. Use the value 32.2 for *g*. The formula requires taking square roots and also using the hyperbolic tangent function. Fortunately, both of those may be found among the standard library functions in C++. The square root function is called `sqrt( )`; it takes one parameter of type `double` and returns the square root of that parameter. So, the statement:

```
double aRoot = sqrt(3.24);
```

would assign the value 1.8 to the variable `aRoot`. The hyperbolic tangent function is called `tanh( )`; it also takes one parameter of type `double` and returns the hyperbolic tangent of that parameter. So, the statement:

```
double hypTangent = tanh(2.72);
```

would assign the value 0.9914 to the variable `hypTangent` (approximately).

Sample input and corresponding output:

Here is a sample input file for the program (named "`FreefallData.txt`"). The first line contains the name of the skydiver. The next three lines specify the skydiver's weight (in pounds), the drag coefficient, and the number of seconds for which a report is to be generated.

```
Cedric the Entertainer
269.1
0.02
4
```

The weight, drag coefficient and time value will always be positive, so you don't have to worry about square roots of negative numbers or division by zero when you apply the given formula.

Here is a corresponding output file for the program, which must be named "`FreefallLog.txt`". It begins with two lines identifying the programmer (you) and the specific project, followed by a blank line. We will always require that information at the beginning of your output file. The remainder of the output file reports the results computed by the program.

The next three lines specify the name of the skydiver, the skydiver's mass (which you must calculate), and the drag coefficient of the skydiver. That is followed by a single blank line, and then a row of column labels corresponding to the number of velocity values to be reported, and a separator line.

The final line contains the computed velocity of the skydiver at each second from 1 up to and including the time value given in the input file. If you have read the *Student Guide to the Curator*, you already know that all the fixed text must be precisely as shown in the sample output. The output should be aligned for easy readability.

```
Programmer: <your name here>
CS 1044:  Falling Bodies

Skydiver:     Cedric the Entertainer
Mass:                          8.36
Drag constant:                 0.02

Seconds:        1       2       3       4
----------------------------------------
Velocity:    31.4    58.5    79.1    93.3
```

Additional samples of input and correct output will be available on the course website.

The program source code:

The following program shell is provided as a starting point for the project. You are not required to use the given code; you may modify it as much or as little as you wish. Regardless, be sure that your final implementation meets the requirements given on the *Programming Standards* page on the course website, especially those for comments.

```
//  Programmer:     Bill McQuain and <your name here>
//  Last modified:  January 27, 2003
//  Compiler:       Visual C++ .NET
//
//  This program applies the velocity formula for a falling body to
//  create a table of data for a skydiver falling for a given number
//  of seconds.  The formula takes into account air resistance and
//  the coefficient of drag of the falling body.
//
#include <fstream>       // for file streams
#include <iomanip>       // for output format manipulators
#include <string>        // for string variables
#include <cmath>         // for sqrt() and tanh() functions
using namespace std;     // put all of the above in scope

const double g           =   32.2;   // gravitational accelereration
const int    COLUMNWIDTH =      8;   // table column setting

int main() {

   string DiverName    = "Anonymous"; // name of skydiver
   double Weight       = 0.0,         // weight of skydiver in pounds
          Mass         = 0.0,         // mass of skydiver
          DragCoeff    = 0.0,         // coeff of drag for skydiver
          fpsV         = 0.0;         // skydiver velocity in feet/second
   int    TimeInterval = 0;           // number of seconds to report

   ifstream Data("FreefallData.txt");  // Open the input file stream.

   getline(Data, DiverName);           // Read the name of sky diver
   Data >> Weight                      //      and weight
        >> DragCoeff                   //      and drag coefficient
        >> TimeInterval;               //      and time interval

   Data.close();                       // Close the input stream.

   ofstream Log("FreefallLog.txt");    // Open the output file stream.
   Log << fixed << showpoint;          // Prepare it for decimal output.

   // Write an identification header to the output file:
   Log << "Programmer: <your name here>" << endl;
   Log << "CS 1044:  Falling Bodies" << endl;
   Log << endl;

   // Calculate the mass of the skydiver.

   // Write the header data to the output file:
   Log << "Skydiver:     "
       << setw(20) << DiverName << endl;
   Log << "Mass:         "
       << setw(20) << setprecision(2) << Mass << endl;
   Log << "Drag constant:"
       << setw(20) << setprecision(2) << DragCoeff << endl;
   Log << endl;
```

```
    // Write the column header label:
    Log << "Seconds: ";
    // Write the column headers:
    int Second = 1;
    while ( Second <= TimeInterval ) {
       Log << setw(COLUMNWIDTH) << Second;
       Second = Second + 1;
    }
    Log << endl;

    int SeparatorWidth = 9 + TimeInterval * COLUMNWIDTH;
    // Write the separator:
    Log << setw(SeparatorWidth) << setfill('-') << "-"
        << setfill(' ') << endl;

    // Write the table data label:
    Log << "Velocity:";

    // Write the sky diver velocity data:
    Second = 1;
    while ( Second <= TimeInterval ) {

       // Calculate the velocity in feet per second:

       // Write the velocity at the current current time:

       // Step to the next time unit:
       Second = Second + 1;
    }
    Log << endl;

    Log.close();              // Close the log file.

    return 0;
}
```

Submitting your program:

You will submit this assignment to the Curator System (read the *Student Guide*), and it will be graded automatically. Instructions for submitting, and a description of how the grading is done, are contained in the *Student Guide*.

You will be allowed up to five submissions for this assignment. Use them wisely. Test your program thoroughly before submitting it. Make sure that your program produces correct results for every sample input file posted on the course website. If you do not get a perfect score, analyze the problem carefully and test your fix with the input file returned as part of the Curator e-mail message, before submitting again. The highest score you achieve will be counted.

The *Student Guide* can be found at: http://www.cs.vt.edu/curator/

The submission client can be found at: http://eags.cs.vt.edu:8080/curator/

Pledge:

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the header comment for your program:

```
//    On my honor:
//
//    - I have not discussed the C++ language code in my program with
//      anyone other than my instructor or the teaching assistants
//      assigned to this course.
//
//    - I have not used C++ language code obtained from another student,
//      or any other unauthorized source, either modified or unmodified.
//
//    - If any C++ language code or documentation used in my program
//      was obtained from another source, such as a text book or course
//      notes, that has been clearly noted with a proper citation in
//      the comments of my program.
//
//    - I have not designed this program in such a way as to defeat or
//      interfere with the normal operation of the Curator System.
//
//    <Student Name>
```

Failure to include this pledge in a submission is a violation of the Honor Code.