Lab 3 - Analytical and Numerical Solutions

Dr. Balakumar Balasingam

As we delve further in programming, every week will introduce new concepts. Each lab will expect that you understand concepts introduced in the previous lab. That being said, let us continue on and introduce the concepts will use in this lab. This week will remain basic as it will be the first real program that you write.

1 New Concepts

1.1 const Variables

A const variable should be used whenever a variable will never be changed or altered by our program. A good example of this is G, the gravitational constant. The syntax for this is typing the qualification const before the variable. It is essentially the same as a regular variable with the exception of the const qualification. This qualification makes it so the variable cannot be modified by the program. An example of a constant double would be declared as const double foo.

1.2 bool Variables

A boolean variable can only store 2 possible values, true or false. In memory, this is represented as a 1 for true and a 0 for false. The syntax for this is the same as declaring say, a double variable; one would just preclude the variable name with a bool instead.

1.3 Escape Characters

An escape character can be placed in a print literal (the quotation part of the cout call), allowing for things such as tabs or new line commands to be simplified. To add a tab into a print literal the syntax is \t. To incorporate a new line command the syntax is \n. There are more escape characters than can be utilized and a quick google search will show you the rest.

2 Background Theory

You will need to go over the Analytical and Numerical Solutions to the falling object equations that were discussed in lecture 1. If you are not familiar with them, you will need to re read the lecture notes. You will notice as the course progresses that falling behind on the theory behind the methods being used in the lab will make the labs unnecessarily difficult.

3 Assignment Instructions

1. Declare the packages that you will need. You will need to include iostream and cmath.
2. Since we are only making calls to the std library, we can type *using namespace std* below where we included the packages.
3. In this program, we are going to use *const* variables that do not change. Underneath where you declared your namespace, declare the following as *const* qualified doubles.
   (a) G = 9.81
   (b) M = 10
   (c) DRAG_COEFF = 12.5
   (d) MAX_TIME = 10
   (e) STEP_SIZE = 1
4. Next, we will need to get into our main function. Begin by declaring the variables that we will modify during our program inside of our main function. You will need two doubles, one called numericalSolution and one called analyticalSolution. In their declaration, set them equal to 0.
5. Once that is all done, we are ready to print the header of our table. Using the cout statement just like in Lab 1, print the three column headers Time, Analytical and Numerical. Seperate the three headers using the tab escape character ("ẗ) direclty in the string that cout is printing.
6. Now we are ready for the meat and potatoes of our program. Declare a for loop that uses a counter that starts at 0, loops until that counter hits MAX_TIME and counts by STEP_SIZE each iteration.
7. Inside of that for loop, calculate the analytical and numerical solutions using the time counter variable from your for loop and your const doubles your declared at the top of your program. Store the result in the analyticalSolution and numericalSolution variables that were declared inside of the main function.
8. Even though the computer now knows internally the value of the solutions we are generating at each step, this is no good to a human operator. Still inside of the for loop, create a print statement that, in every iteration, prints the Time, anaylticalSolution and numericalSolution in accordance with your table header. Dont forget to use the tab escape character in order to space your table properly.

This finished program will be what you show the GA. You will need to answer a random question on how your code works.
This is what your finished output should look like:

```
Time Analytical Numerical
-------------------------------------------------
0 0 9.81
1 5.59951 7.3575
2 7.2038 7.97063
3 7.66343 7.81734
4 7.79512 7.85566
5 7.83285 7.84608
6 7.84366 7.84848
7 7.84676 7.84788
8 7.84764 7.84803
9 7.8479 7.84799
```

You will also need to submit a .cpp with your code on Blackboard. Failure to submit this file will result in a mark of 0 for the lab.