CST 8233 – W19 - Assignment #1

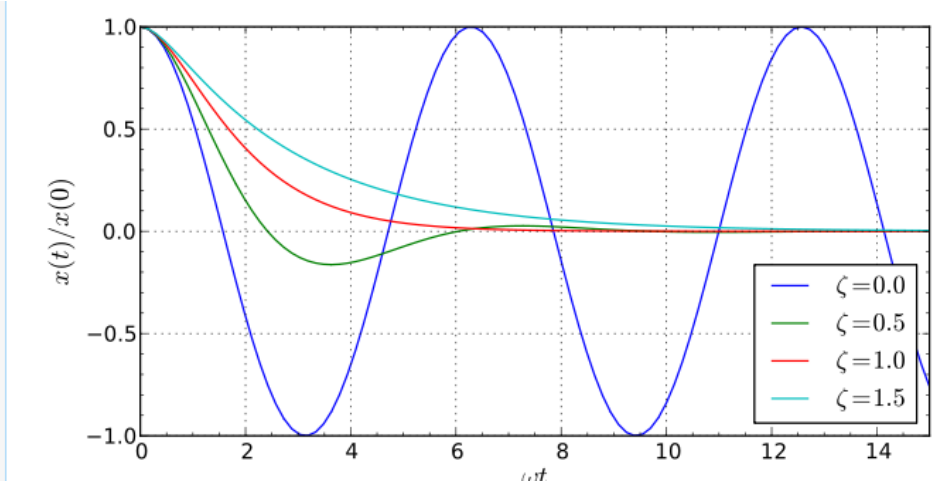
Building Motion Simulation

Due Time: 23.59, 17 February 2019 **Earnings:** 8% of your final grade

NOTE: Plan to finish a few days early to avoid last minute hardware/software holdups for which no allowance is given.

NOTE: The code in this assignment must be your own work. It must not be code taken from another student or written for you by someone else, even if you give a reference to the person you got it from (attribution); if it is not entirely your own work it will be treated as plagiarism and given a fail mark, or less.

Purpose: You are employed by an architecture company to write code for a building simulation. The company has a client that wants to model the motion response of a tall building following being hit by a sudden strong gust of wind. The simulation must run on a PC in real time and is part of a larger simulation that includes the motion of traffic and pedestrians etc. Because it must run in real-time it is too time consuming to use the more accurate but slow math library functions, so your task is to investigate a less accurate but faster Maclaurin Series approximation. The motion of the building is damped so it performs less than one complete oscillation. The following graph shows oscillation under differing degrees of damping:



The formula for the movement D of the top of the building in meters with time t in seconds after the gust hits is:

$$D(t) = e^{-t}\cos(t)$$

which is close to the red critically damped function on the graph, 2nd from the top, that decays to 0.0 without swinging negative.

Determine the Maclaurin Series approximation to $D(t) = e^{-t}\cos(t)$ as a power series in t up to 6 non-zero terms (you need the sixth term to estimate the truncation error when the user chooses a series with five terms).

As a check that you have to correct series, you should to derive it two ways to see if they agree: 1. from 1st principles by differentiation, 2. by the product of the separate e^{-t} and cos(t) series.

The purpose of the series approximation is to speed up its execution so no functions are used in the evaluation of the terms of the series, whether they be math library functions such as exp() or pow() etc., or any other function, and terms such as x^3 for example are written explicitly in your code as x*x*x and factorials are precalculated and written in as numbers.

Algorithm: While the user wishes to continue, the user is asked to choose the number of terms in the series up to a maximum of five (you should reject invalid integers). Then the user selects a range of t, somewhere between 0 and +2.0 (you should reject values outside this range) up to which the series is evaluated from 0 in ten equal increments.

For each value of t the Maclaurin series approximation to the function is output together with the exact

value calculated using the math library functions exp and cos.
The error that results from using the series approximation is calculated in two different ways.

1. From comparison with the exact value calculated using the math library functions:
   Exact % Error = 100*(exact value – series value)/exact value

2. From the first truncated term. This gives you an idea of how well the first truncated term approximates to the error.
   Truncation % Error = 100*first truncated term/series value.

Set up an empty project in Visual Studio 2017 with the name ass1. Write the code to implement the application in a file named ass1.cpp using the C (or/C++) programming language.
Example output is given at the end. Yours should be very similar, but may differ very slightly due to round-off errors from the way you have evaluated your series. Note than your assignment might be tested with different parameters than those shown in the example output.

See the Marking Sheet for how you can lose marks, but you definitely lose 60% or more if:
- the series is wrong
- Your application won't build in Visual Studio 2017
- Your application crashes in normal operation
- I can't build it because you submitted the wrong files or the files are missing, even if it's an honest mistake – this gets 100% deduction.

What to Submit : Use Blackboard to submit this assignment as a zip file (**not** RAR, not 9zip, not 7 zip) containing only the source code file (ass1.cpp). The name of the zipped folder **must** contain your name as a prefix so that I can identify it, for example using my name the file would be tyleraAss1CST8233.zip. It is also vital that you include the file header (as specified in the Submission Standard) so the file can be identified as yours. Use comment lines in the file to include the header.
There is a late penalty of 25% per day - even one minute is counted late.
Don't send me the file as an email attachment – it will get 0.

Example Output

```
Evaluate the Maclaurin Series approximation to exp(-t)*cos(t)

1: Evaluate the series
2: quit

1

Evaluating the series
Please enter the number of terms in the series (max 5 terms): 5

Please enter the range of t to evaluate in 10 increments( 0.0 < t < +2.0): 2

MACLAURIN SERIES
  x          Series         Exact           Exact % Error   Trunc. % Error
0.000e+00   1.00000e+00   1.00000e+00        0.00000e+00   0.00000e+00
2.000e-01   8.02411e-01   8.02411e-01        -2.40826e-06   -2.53205e-06
4.000e-01   6.17408e-01   6.17406e-01        -3.80965e-04   -4.21218e-04
6.000e-01   4.52992e-01   4.52954e-01        -8.43593e-03   -9.80907e-03
8.000e-01   3.13323e-01   3.13051e-01        -8.69389e-02   -1.06242e-01
1.000e+00   2.00000e-01   1.98766e-01        -6.20775e-01   -7.93651e-01
1.200e+00   1.13344e-01   1.09140e-01        -3.85188e+00   -5.01799e+00
1.400e+00   5.36747e-02   4.19134e-02        -2.80609e+01   -3.11736e+01
1.600e+00   2.25920e-02   -5.89528e-03        4.83222e+02   -1.88601e+02
1.800e+00   2.42560e-02   -3.75563e-02        1.64586e+02   -4.00634e+02
2.000e+00   6.66667e-02   -5.63193e-02        2.18373e+02   -3.04762e+02

Evaluate the Maclaurin Series approximation to exp(-t)*cos(t)

1: Evaluate the series
2: quit
```

```
1

Evaluating the series
Please enter the number of terms in the series (max 5 terms): 4

Please enter the range of t to evaluate in 10 increments( 0.0 < t < +2.0): 1

MACLAURIN SERIES
  x          Series         Exact           Exact % Error   Trunc. % Error
0.000e+00   1.00000e+00   1.00000e+00       0.00000e+00   0.00000e+00
1.000e-01   9.00317e-01   9.00317e-01       3.70068e-05   3.70240e-05
2.000e-01   8.02400e-01   8.02411e-01       1.32692e-03   1.32935e-03
3.000e-01   7.07650e-01   7.07731e-01       1.13995e-02   1.14463e-02
4.000e-01   6.17067e-01   6.17406e-01       5.49041e-02   5.53155e-02
5.000e-01   5.31250e-01   5.32281e-01       1.93644e-01   1.96078e-01
6.000e-01   4.50400e-01   4.52954e-01       5.63808e-01   5.75488e-01
7.000e-01   3.74317e-01   3.79809e-01       1.44618e+00   1.49668e+00
8.000e-01   3.02400e-01   3.13051e-01       3.40217e+00   3.61199e+00
9.000e-01   2.33650e-01   2.52728e-01       7.54874e+00   8.42414e+00
1.000e+00   1.66667e-01   1.98766e-01       1.61494e+01   2.00000e+01

Evaluate the Maclaurin Series approximation to exp(-t)*cos(t)

1: Evaluate the series
2: quit

1

Evaluating the series
Please enter the number of terms in the series (max 5 terms): 2

Please enter the range of t to evaluate in 10 increments( 0.0 < t < +2.0): 2

MACLAURIN SERIES
  x          Series         Exact           Exact % Error   Trunc. % Error
0.000e+00   1.00000e+00   1.00000e+00       0.00000e+00   0.00000e+00
2.000e-01   8.00000e-01   8.02411e-01       3.00426e-01   3.33333e-01
4.000e-01   6.00000e-01   6.17406e-01       2.81916e+00   3.55556e+00
6.000e-01   4.00000e-01   4.52954e-01       1.16908e+01   1.80000e+01
8.000e-01   2.00000e-01   3.13051e-01       3.61125e+01   8.53333e+01
1.000e+00   0.00000e+00   1.98766e-01       1.00000e+02   inf
1.200e+00   -2.00000e-01   1.09140e-01      2.83251e+02   -2.88000e+02
1.400e+00   -4.00000e-01   4.19134e-02      1.05435e+03   -2.28667e+02
1.600e+00   -6.00000e-01   -5.89528e-03    -1.00776e+04   -2.27556e+02
1.800e+00   -8.00000e-01   -3.75563e-02    -2.03014e+03   -2.43000e+02
2.000e+00   -1.00000e+00   -5.63193e-02    -1.67559e+03   -2.66667e+02

Evaluate the Maclaurin Series approximation to exp(-t)*cos(t)

1: Evaluate the series
2: quit
```