Student ID: 8444507 Name: M. Kashif Hussain

Scientific Computing Coursework 1

Q1)

Given that the Maclaurin series for exp is as follows:

$$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

By denoting the nth term as $T_n(x)$ where $T_0(x) = 1$ and $T_1(x) = x\ x$, we can define a recurrence relation as follows (for n > 0):

$$T_n(x) = T_{n-1}(x) \times \frac{x}{n}$$

Q2)

A C++ function to approximate the value of $\exp(x)$ can be written as follows:

```
//Generates an approximation to e^x
double ExpSeries(double x, int N)
{
	//Base case
	if (N == 0)
		return 1;
	else
		return ExpSeries(x, N - 1) + pow(x, N) / factorial(N);
}
```

With the factorial function defined as follows:

```
//Implements the factorial function
int factorial(int n)
{
	if (n == 1 || n == 0)
		return 1;
	else
		return factorial(n - 1) * n;
}
```

Below are some test cases:

Value of X	Value of N	Expected Result	Observed Result
0.0	0	1	1
2.0	0	1	1
1.0	2	2.5	2.5
-1.0	1	0	0
6.0	3	61	61
0.0	20	1	1

Q3) The code on the following page produces five columns which display the following:

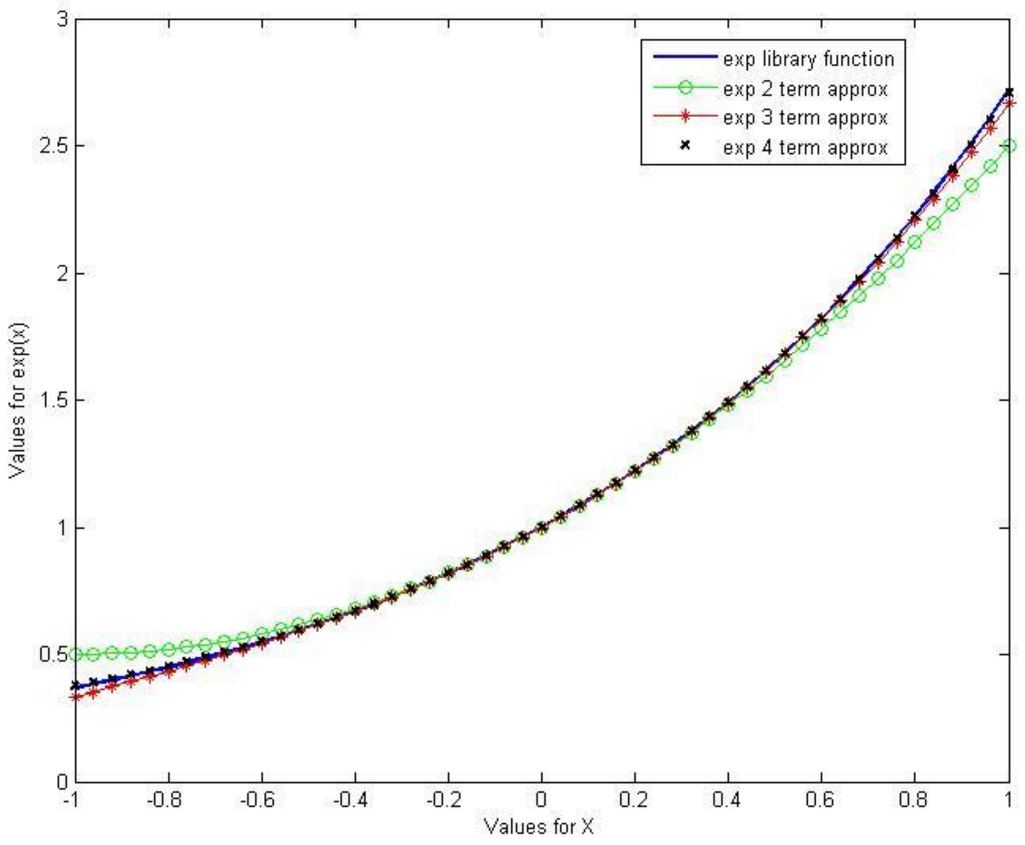
1. The value for x.
2. The value of $exp(x)$, as evaluated by the standard library function exp.
3. The Two-term Maclaurin series approximation to $exp(x)$.
4. The Three-term Maclaurin series approximation to $exp(x)$.
5. The Four-term Maclaurin series approximation to $exp(x)$.

Student ID: 8444507   Name: M. Kashif Hussain

```
#include <iostream>
#include <math.h>
#include <fstream>
using namespace std;

//Implements the factorial function
int factorial(int n)
{
	if (n == 1 || n == 0)
		return 1;
	else
		return factorial(n - 1) * n;
}

//Generates an approximation to e^x
double ExpSeries(double x, int N)
{
	//Base case
	if (N == 0)
		return 1;
	else
		return ExpSeries(x, N - 1) + pow(x, N) / factorial(N);
}

int main()
{

	//variable of type ofstream called expFile
	ofstream expFile;
	//Opens specified text document for writing
	expFile.open("Table of Exp Values.txt");
	//If attempt to open file failed, return 1
	if (!expFile) return 1;

	double xValue = -1;
	//Loop to get atleast 50 rows of data for comparing exp(x) approximations
	for (int i = 0; i <= 50; i++)
	{
		//setting value to 0 if almost 0 (double precision offset)
		if (xValue < 0.01 && xValue > -0.01)
			xValue = 0.0;
		//Write 5 columns to file with values listed below
		expFile.width(10); expFile << xValue;
		expFile.width(10); expFile << exp(xValue);
		expFile.width(10); expFile << ExpSeries(xValue, 2);
		expFile.width(10); expFile << ExpSeries(xValue, 3);
		expFile.width(10); expFile << ExpSeries(xValue, 4) << endl;
		//step size for xValue
		xValue += 2.0/50;
	}
	//close file
	expFile.close();

	return 0;
}
```

Q4)

The image below displays the comparison of the exp function and my own approximation function for varying values of x. This plot was constructed using MATLAB. The code can be found below



```
function compareExp()
data = load('Table of Exp Values.txt');
plot(data(:,1), data(:,2), 'b', 'LineWidth', 2);
hold on;
plot(data(:,1), data(:,3), 'g-o');
plot(data(:,1), data(:,4), 'r-*');
plot(data(:,1), data(:,5), 'kx', 'LineWidth', 2);

legend('exp library function', 'exp 2 term approx', 'exp 3 term approx',...
        'exp 4 term approx');
xlabel('Values for X');
ylabel('Values for exp(x)');

end
```