**PROBLEM STATEMENT**

This program will calculate the end time and cost of a call given the start time, duration, and the different pay rates at certain times of the day.

**Output Display**

N = 2 e(1) = 2 e(2) = 2.5 delta = 0.5

N = 3 e(2) = 2.5 e(3) = 2.67 delta = 0.17

N = 4 e(3) = 2.667 e(4) = 2.708 delta = 0.041

N = 5 e(4) = 2.7083 e(5) = 2.7167 delta = 0.0084

N = 6 e(5) = 2.71667 e(6) = 2.71805 delta = 0.00138

N = 7 e(6) = 2.71805 e(7) = 2.71825 delta = 0.00020

N = 8 e(7) = 2.71825 e(8) = 2.71827 delta = 0.00002

N = 9 e(8) = 2.71827 e(9) = 2.71828 delta = 0.00001

N = 10 e(9) = 2.7182815 e(10) = 2.7182818 delta = 0.0000003

**Structure Chart**

Main

Factorial

Flowchart for Main

Start

Set n = 0  
Set old\_e = 0.0  
Set new\_e = 1.0

Pseudo Code for Main

End

While (new\_e – old\_e > 0.000001)

Display (new\_e – old\_e)

new\_e = new\_e + 1.0 / (factorial(n++))

old\_e = new\_e

Start of algorithm for Main

1. Define variables n = 0, old\_e = 0, new\_e = 1
2. While(new\_e – old\_e > 0.000001)  
   - old\_e = new\_e  
   - new\_e = new\_e + 1.0 / (factorial(n++))  
   - Display output
3. Display final output

End of algorithm for Main

Flowchart for Factorial

Start

Pseudo Code for Factorial

End

Return 1

Return (n \* factorial(n-1))

else

If (n == 0)

Start of algorithm for Factorial

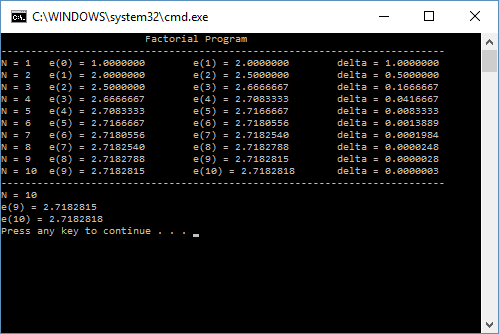
1. If (n == 0)

- return 1

1. Else  
   - return(n \* factorial(n-1))

End of algorithm for Factorial

**Output**



**User Instructions**

This program requires no input from the user. Simply run the program and it will calculate the difference between two sequential factorial numbers until the difference is less than 0.000001.

**Comment**

This lab was real nice, quick and easy. It was cool to see how a recursive function works and it is interesting that a function can call itself. I’m interested in how I am going to use this new technique in the future.