Consider the integral
\[ I_n = \int_0^1 x^n e^{x-1} \, dx \]  
(1)

where \( n \) is a non-negative integer. Note that \( x^n e^{x-1} > 0 \) for all non-negative integers \( n \) and for all \( x \in (0, 1) \). Therefore, \( I_n > 0 \) for all non-negative integers \( n \). Moreover, \( x^n > x^{n+1} \) for all non-negative integers \( n \) and for all \( x \in (0, 1) \). Hence, \( x^n e^{x-1} > x^{n+1} e^{x-1} \) for all non-negative integers \( n \) and for all \( x \in (0, 1) \). Therefore, \( I_n > I_{n+1} \) for all non-negative integers \( n \). That is, the sequence \( I_0, I_1, I_2, \ldots \) is positive and monotonically decreasing.

One way to compute \( I_n \) for any non-negative integer \( n \) is as follows. Note that
\[ I_0 = \int_0^1 e^{x-1} \, dx = [e^{x-1}]_0^1 = 1 - e^{-1} = 1 - 1/e \]  
(2)

Moreover, for \( n \geq 1 \), integrating (1) by parts leads to
\[ I_n = \int_0^1 x^n e^{x-1} \, dx = [x^n e^{x-1}]_0^1 - \int_0^1 nx^{n-1} e^{x-1} \, dx = 1 - n I_{n-1} \]  
(3)

Therefore, we can use the recurrence
\[
\begin{align*}
I_0 &= 1 - 1/e \\
I_n &= 1 - nI_{n-1} & \text{for } n = 1, 2, 3, \ldots 
\end{align*} 
\]  
(4)

to evaluate \( I_n \) for any non-negative integer \( n \). You can use the MatLab function \( \text{exp}(1.0) \) to compute \( e \) accurately in (4). If you use (4) in a little MatLab program to compute \( I_n \) for \( n = 0, 1, \ldots, 25 \), you get the values listed on the next page.

You should try to write a little MatLab program yourself to compute \( I_n \) for \( n = 0, 1, \ldots, 25 \).
The values for $I_n$ look reasonable for $n = 0, 1, 2, \ldots, 15$ or so, but the values are clearly wrong for $n = 20, 21, \ldots, 25$, since the computed $I_n$ values are not all positive and monotonically decreasing, as they should be.

1. Explain why my MatLab program computes such inaccurate values for $I_n$ for $n = 20, 21, \ldots, 25$.

My program is a correct implementation of the recurrence (4). That is, the inaccuracy in the table above is not a result of a programming bug. The problem is with the recurrence (4) itself and the rounding errors that occur when you implement it in floating-point arithmetic.

In this explanation, it is not sufficient to say that there is round-off error in the computation. Although this is true and should play a part in your explanation, there is round-off error in almost all floating-point computations and most of them produce accurate results. You need to explain why the round-off error produces such bad results in this case.
2. Re-arrange the recurrence

\[ I_n = 1 - nI_{n-1} \]

starting it from a different initial value so that your new recurrence computes accurate values for \( I_n \) for \( n = 0, 1, 2, \ldots, 25 \).

Explain why you believe your new method produces accurate results.

Write a little MatLab program that uses your new method to compute \( I_n \) for \( n = 0, 1, 2, \ldots, 25 \). Your program should also print \( n \) and \( I_n \) for \( n = 0, 1, 2, \ldots, 25 \) in a nicely formatted table.