MACM 316 Computing Assignment 3:
Newton's Method and Convergence Analysis

This report investigates the computation of root finding using Newton's Method and convergence behaviour of the function,

$$f(x) = x\sin(\pi x) + \cos(\pi x) \cdot \sin(\pi x)$$

A) In the study, trials are conducted using two initial cases: $p_0 = 0.77$ and $p_0 = 0.99$, where the respective results of Newton's Method, modeled by the equation,

$$p_{n+1} = p_n - \frac{f(x)}{f'(x)}$$

iterations can be found in the tables on the left. The tolerance accuracy used in this experiment is 1^{-6}decimal places. Newton's method continues to iterate until a break condition, when the error is less than the set tolerance, which indicates stable convergence.

With the initial estimate of $p_0 = 0.77$, we observe that the function, $f(x)$, converges at $p = 0.7893$ very rapidly and the error begins to stabilize after 5 iterations. By Newton's method, we can confirm that the initial estimate of $p_0 = 0.77$ converges to 0.7893 rapidly after 5 iterations.

Applying Newton's method using the initial estimate of $p_0 = 0.99$, $f(x)$ converges to 1 after 14 iterations, once error has stabilized and meets tolerance accuracy.

Iteration	p_in	Error P_in	Error P1	Alpha	Lambda
1	0.78732	0.01983	0.0025119	NaN	NaN
2	0.78978	0.0025119	4.7714e-05	NaN	NaN
3	0.78983	4.7714e-05	2.6075e-06	1.9183	4.6372
4	0.78983	2.6075e-06	2.6284e-06	1.0852	0.089389
5	0.78983	2.6284e-06	2.6284e-06	0.74202	0.013009

```
For: 0.77 Convergence reached at run: 5
```

Iteration	p_in	Error P_in	Error P1	Alpha	Lambda
1	0.99513	0.01	0.0048659	NaN	NaN
2	0.9976	0.0048659	0.0024026	NaN	NaN
3	0.99881	0.0024026	0.001194	0.97972	0.44321
4	0.9994	0.001194	0.00059524	0.98517	0.45507
5	0.9997	0.00059524	0.00029718	0.9888	0.46354
6	0.99985	0.00029718	0.00014848	0.99132	0.46978
7	0.99993	0.00014848	7.4213e-05	0.99312	0.4745
8	0.99996	7.4213e-05	3.71e-05	0.99444	0.47813
9	0.99998	3.71e-05	1.8548e-05	0.99543	0.48099
10	0.99999	1.8548e-05	9.2736e-06	0.99619	0.48327
11	1	9.2736e-06	4.6367e-06	0.99678	0.48512
12	1	4.6367e-06	2.3183e-06	0.99724	0.48665
13	1	2.3183e-06	1.1592e-06	0.99762	0.48792
14	1	1.1592e-06	5.7958e-07	0.99792	0.48898

```
For: 0.99 Convergence reached at run: 14
Convergence to: 1
```

B) Now that we have found the roots for $f(x)$, we must now analyze the convergence behaviour of these roots, using the relation:

$$\left|p_{n+1} - p\right| \approx \lambda\left|p_n - p\right|^{\alpha}$$

Where λ is the asymptotic error constant, α is the order of convergence, and the associated error terms (E_n, E_{n+1}). If we convert this relation to be in the form log-log, we arrive at:

$$log(E_{n+1}) \approx \alpha \cdot log(E_n) + C \text{ where } C = log(\lambda)$$

In the above form, we can now apply log-log/linear regression, using polyfit() in MATLAB, where we can then solve for α directly by finding the slope of the regression line, and indirectly solve for λ by finding the constant C. Rearranging for λ:

$$C = log(\lambda) \rightarrow \lambda = e^C$$

The reader may observe in the table, the first two readings appear as NaN. This is expected as for log-log regression we need at least two data points to form a line.

For $p_0 = 0.77$, we can apply the above log-log regression method for each iteration of Newton's method execution, yielding $\alpha = 0.74202$, $\lambda = 0.013$. What should be noted is the mismatch between the expected order of convergence for Newton's, 2, versus the close to linear order obtained here is the lack of data points. For this estimate, we have established that it converges accurately extremely quickly, in 5 iterations, which leave little to no data points for the regression line to act upon. There may also be numerical instability at such small values that can also cause errors in the computer's calculations.

For $p_0 = 0.99$, we again can apply log-log regression, yielding the results $\alpha = 0.99792$, $\lambda = 0.48898$. This again, deviates from the expected α value of 2 for Newton's method. However, here the root of 1 has a multiplicity of two. The ideal case for Newton's method is 2; however, this should stand true for values with a multiplicity of one. We can verify using:

$$\lambda = \frac{m-1}{m} \text{ where m is multiplicity}$$

For roots with a multiplicity of two, the error constant, λ, is 0.5 which indicates that for each iteration, the error is halved. The results obtained follow this order, therefore, the results are valid following theoretical expectations of Newton's method for roots with multiplicity of 2.