Given:

$g(x) = \cos(x) - x^3$ {has root on (0, 1)}

Root $r \sim 0.865474$

Find answer within 4 decimal places; Satisfy $|x_c - r| < 0.5 \times 10^{-4}$

Problem Statement 2: Use Newton's method with the given initial root $x_0 = 0.3$.

Algorithm Description: Instead of finding the root by approximating via midpoint and setting up continuously smaller bounds ourselves to interval containing the root, we can try to determine the root faster by using the derivative of the function to draw up values of x closer to the root r via the formula $g(x) = g'(x)(x - r)$ where we solve for r so $r = x - (g(x) / g'(x))$ such that x approaches r to acceptable level via continuous iterations of the updated x we get within error defined in the problem resulting in a more precise, faster approach.

Pseudocode:

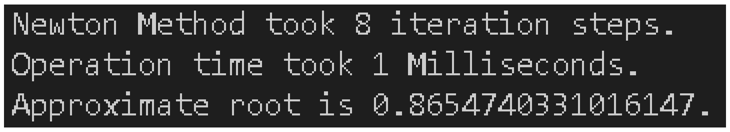
```
Function newtonMethod(x, previousSoln, count):
    If abs(previousSoln - x) < tolerance AND count > 1:
        print("Newton Method took " + count + " iterations.")
        Return x

    Return newtonMethod(x - (g(x) / gPrime(x)), x, count + 1)
End

Function g(x):
    Return cos(x) - $x^3$
End

Function gPrime(x):
    Return -sin(x) - $3x^2$;
End
```

Results: (p12.java)



```
Newton Method took 8 iteration steps.
Operation time took 1 Milliseconds.
Approximate root is 0.8654740331016147.
```

Observations & Analysis

The root estimated by *p12.java* is approximately 0.86547403 which is well within 4 decimal points of the approximately correct root of 0.865474. As for the program efficiency, I would say it is sufficient as it only took no more than a millisecond to conduct 8 iterations of 12 floating point operations at most (last iteration could be fewer operations returning on first if statement) on each iteration which is about at least 96,000 operations per second. The algorithm is good because it only takes no more than 1 millisecond to figure out the root. I am guessing there are ways to write algorithms using bitwise operators for probably faster results and other more ways unknown to me that could result in faster runtimes. This program is scalable to any predefined initial root generally and runs with any java configured environment.