In [11]:

```python
def bisection(f, a, b):

    tolerance = 1e-10
    max_iteration = 100000

    if f(a) * f(b) > 0:
        return None
    iter_count = 0
    while (b - a) / 2.0 > tolerance and iter_count < max_iteration:
        midpoint = (a + b) / 2.0
        if f(midpoint) == 0:
            return midpoint
        elif f(a) * f(midpoint) < 0:
            b = midpoint
        else:
            a = midpoint
        iter_count += 1
    return (a + b) / 2.0

import math

# Test Case 1:
f1 = lambda x: math.exp(x) + math.log(x)
root1 = bisection(f1, 0.01, 1)
print(f"Root of f1 in [0,1]: {root1}")

# Test Case 2:
f2 = lambda x: math.atan(x) - x**2
root2 = bisection(f2, 0, 2)
print(f"Root of f2 in [0,2]: {root2}")

# Test Case 3:
f3 = lambda x: math.sin(x)
root3 = bisection(f3, 3, 4)
print(f"Root of f3 in [3,4]: {root3}")

# Test Case 4:
f4 = lambda x: math.log(math.cos(x))
root4 = bisection(f4, 5, 7)
print(f"Root of f4 in [5,7]: {root4}")
```

```
Root of f1 in [0,1]: 0.2698741375870304
Root of f2 in [0,2]: 1.9999999999417923
Root of f3 in [3,4]: 3.141592653642874
Root of f4 in [5,7]: None
```