// Determinar uma aproximação numérica do zero de e^x+x, com erro absoluto inferior a \epsilon=0.001

#include <stdio.h>

#include <math.h>

// Define the function f(x) = e^x + x

double f(double x) {

return exp(x) + x;

}

int main() {

double a = -1, b = 0; // Initial interval [a, b]

double tol = 1e-3; // Absolute error tolerance

double c, error;

int iterations = 1; // Start counting from the first middle point computation

// Compute the first midpoint

c = (a + b) / 2.0;

error = fabs(f(c));

// Check if the initial midpoint is already a good approximation

if (error < tol) {

printf("Solution found at x = %.10f with error %.10f after %d iteration(s).\n", c, error, iterations);

printf("Final Interval: [%.10f, %.10f]\n", a, b);

return 0;

}

// Perform further iterations if needed

while ((b - a) / 2.0 > tol) {

iterations++;

if (f(a) \* f(c) < 0) {

b = c;

} else {

a = c;

}

c = (a + b) / 2.0;

error = fabs(f(c));

if (error < tol) {

break;

}

}

printf("Solution found at x = %.10f with error %.10f after %d iteration(s).\n", c, error, iterations);

printf("Final Interval: [%.10f, %.10f]\n", a, b);

return 0;

}