**REPORT 1**

**Lab Name:** Solutions to Non-linear Equations: Bisection Method.

**Objectives:**

• To solve Non-linear Equation Using Bisection method.

**Introduction:**

The bisection method is a simple numerical method used to find the root of an equation. **algorithm:**

1. First, we need to choose two initial guesses for the root, let's call them a and b. These two values should be such that the function f(a) and f(b) have opposite signs. i.e.

f(a)\* f(b) < 0

1. Next, you find the midpoint c between a and b. This is done using the formula:

c = (a + b) / 2

1. You then evaluate the function at the midpoint, i.e., calculate f(c).
2. If f(c) is zero or very close to zero, then c is the root of the equation. Otherwise, repeat the process by choosing a new interval [a, c] or [c, b], depending on whether f(c) has the same sign as f(a) or f(b).
3. Repeat steps 2-4 until we find the root to the desired level of accuracy.

To implement this method in Python, you can write a function that takes as input the function f(x), the initial guesses a and b, and the desired level of accuracy. Here's some sample code:

**Source Code:**

def func(x):

return x\*x\*x - x\*x + 2

def bisection(a,b):

if (func(a) \* func(b) >= 0):

print("You have not assumed right a and b")

return None

c = (a+b)/2

if (func(c) == 0.0):

return c

elif (func(c)\*func(a) < 0):

return bisection(a, c)

else:

return bisection(c, b)

a = float(input("Enter the lower bound of the interval: "))

b = float(input("Enter the upper bound of the interval: "))

root = bisection(a, b)

if root is None:

print("Bisection method failed to find a root.")

else:

print("The value of root is: ", root)

**REPORT 2**

**False Position:**

def false\_position(f, a, b, tol):

c = (a\*f(b) - b\*f(a)) / (f(b) - f(a))

if abs(f(c)) < tol:

return c

elif f(c) \* f(a) < 0:

return false\_position(f, a, c, tol)

else:

return false\_position(f, c, b, tol)

expr = input("Enter the function f(x): ")

f = lambda x: eval(expr)

a = float(input("Enter the lower limit of the interval: "))

b = float(input("Enter the upper limit of the interval: "))

tol = float(input("Enter the desired level of accuracy: "))

root = false\_position(f, a, b, tol)

if root:

print(f"The root of {expr} in [{a}, {b}] is approximately {root:.6f}")

else:

print(f"No root of {expr} found in [{a}, {b}]")

**REPORT 3**