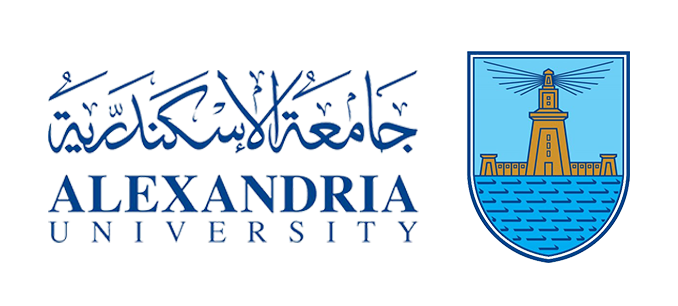
**Root Finder Program**

****

**Name 1: Abdelrahman Salem Mohamed**

**ID: 6309**

**Name 2: Reem Abdelhalim**

**ID: 6114**

**Course: Numerical Analysis**

**Instructor: Zeinab**

Contents

[**Objective** 3](#_Toc100518557)

[**Description** 3](#_Toc100518558)

[**Program specs** 3](#_Toc100518559)

[**Procedures** 3](#_Toc100518560)

[**Methods Algorithm & Pseudocode** 6](#_Toc100518561)

[Bisection Method 6](#_Toc100518562)

[False Position Method 8](#_Toc100518563)

[Fixed Point Method 9](#_Toc100518564)

[Newton – Raphson's Method 11](#_Toc100518565)

[Secant 12](#_Toc100518566)

[**Code Result Analysis** 13](#_Toc100518567)

[Expression one 14](#_Toc100518568)

[Expression two 18](#_Toc100518569)

[Expression three 22](#_Toc100518570)

[**Problems we faced** 25](#_Toc100518571)

[**Code** 25](#_Toc100518572)

# **Objective**

The aim of this assignment is to compare and analyze the behavior of numerical methods studied in class {Bisection, False-position, Fixed point, Newton Raphson, Secant}.

# **Description**

It is required to implement a root finder program which takes as an input the equation expression and choose which methods to apply to find the root and its required parameters.

# **Program specs**

Code written in python and GUI implemented using Tkinter, simple user interface just to see the results in proper format and readable way .

# **Procedures**

The program start with window and user should enter **valid** mathematical expression function as shown in figure 1.

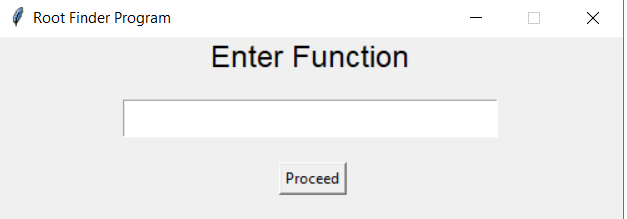


Figure 1

And if the user enter invalid expression format Error window is show as presented in figure 2.

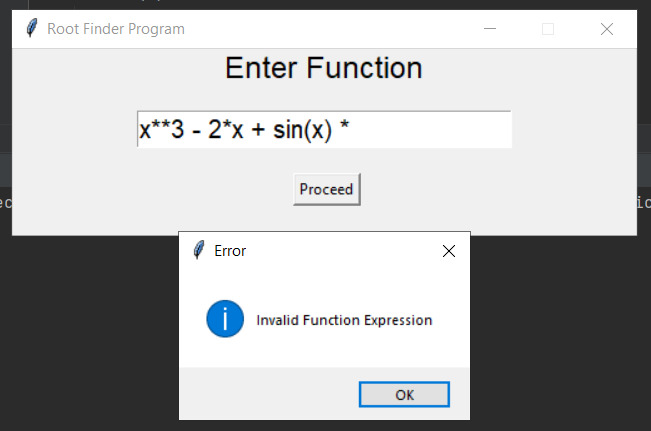


Figure 2

Let's assume the user enter the following valid expression and click proceed as shown in figure 3.

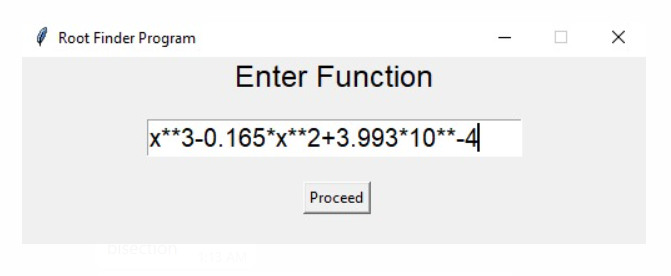


Figure 3

The user will be directed to the next window to select all available root find methods {Bisection, False position, Fixed Point, Newton & Secant} as shown in figure 4.

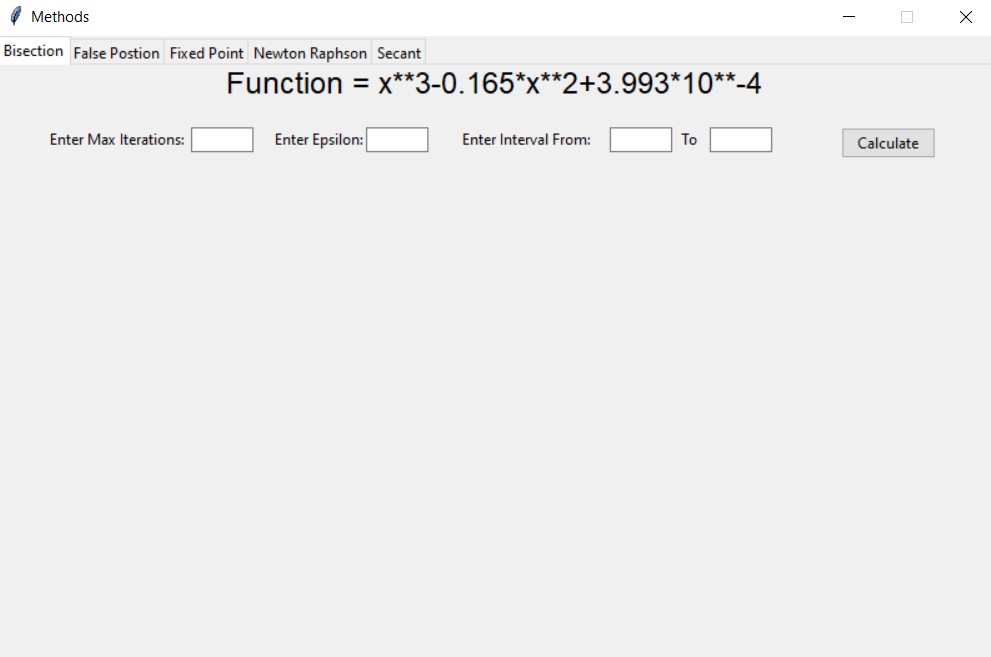


Figure 4

For every method there are couple of parameters the user can enter to apply on the algorithm and from all methods Max iterations and epsilon (approximate absolute relative error) are common and if the user ignores any of them the default pre-defined value will be used, after enter the algorithm parameter then the user can get the results of each iteration and display each calculated epsilon.

# **Read from file**

Graphical user interface, application

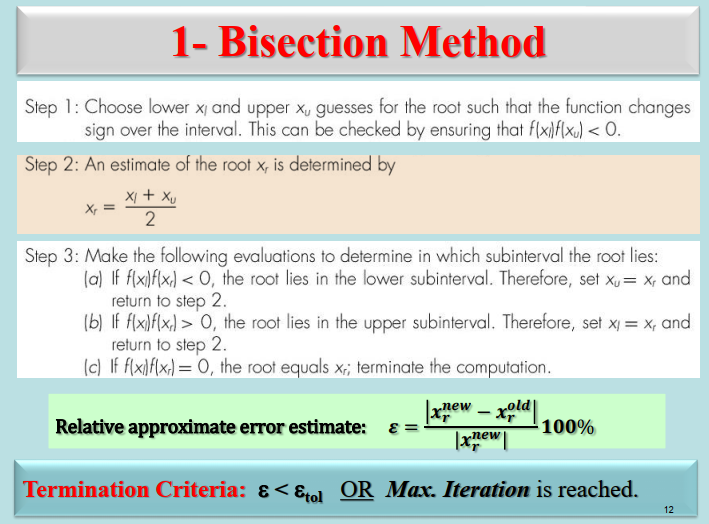
Description automatically generated

Click on browse file button to choose text file to read equation from.

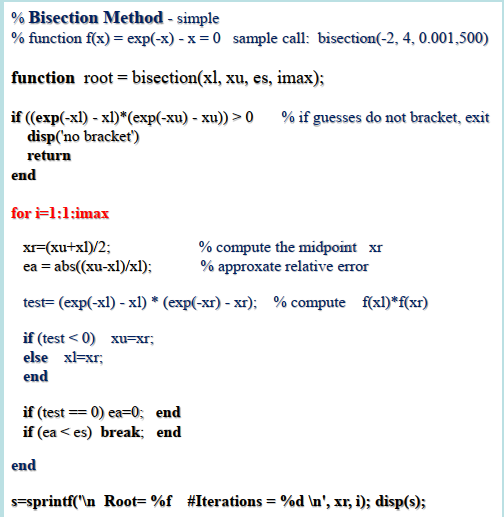
# **Methods Algorithm & Pseudocode**

## Bisection Method

Algorithm



Pseudocode



### How many Iteration needed by bisection method:

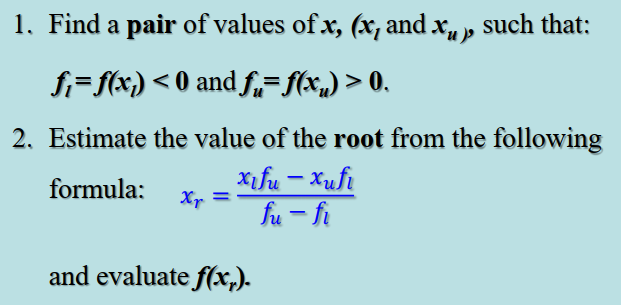
A picture containing diagram

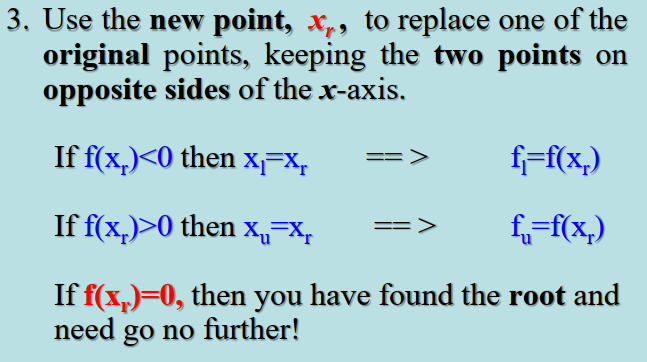
Description automatically generated

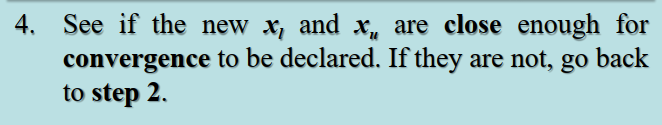
We used this equation to calculate k, which is maximum number of iterations needed to calculate the root

## False Position Method

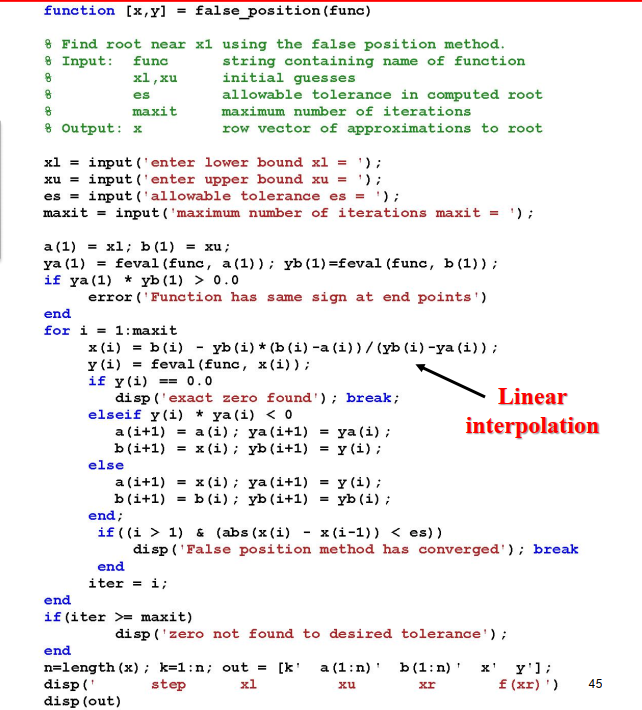
Algorithm





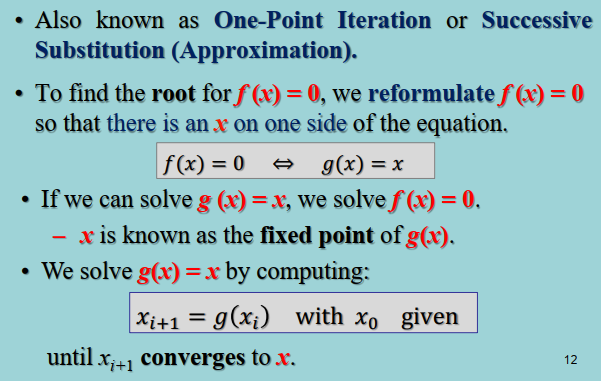


Pseudocode

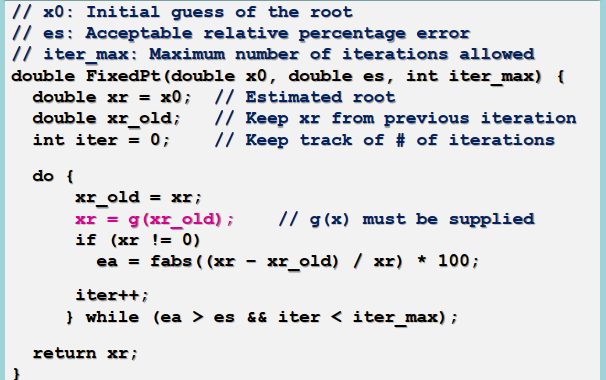


## Fixed Point Method

Algorithm



Pseudocode

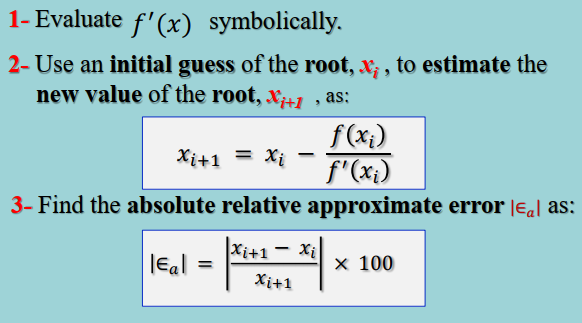


### Check convergence of equation:

Diagram

Description automatically generated

## Newton – Raphson's Method

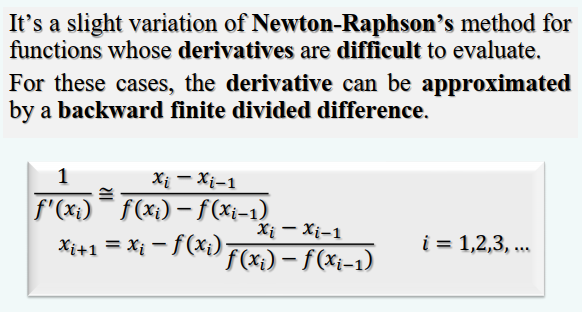


Pseudocode

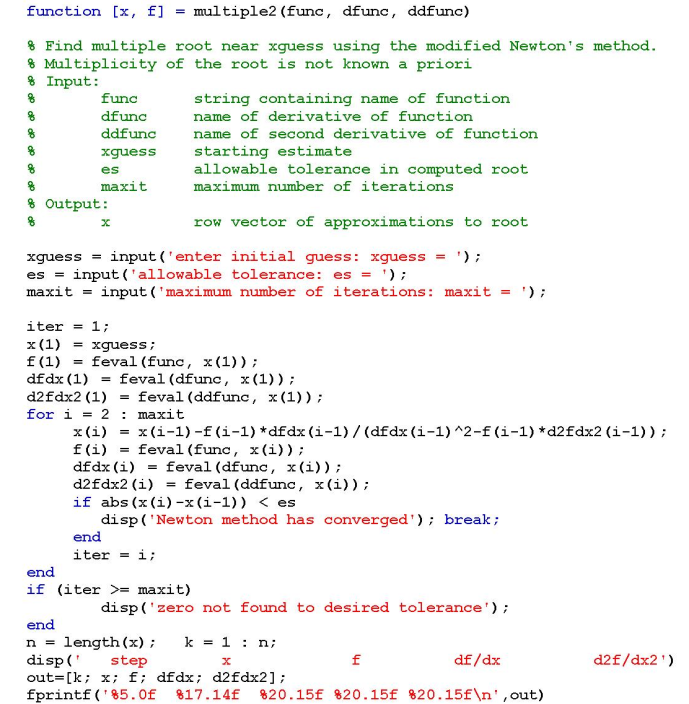


## Secant

Algorithm



Pseudocode

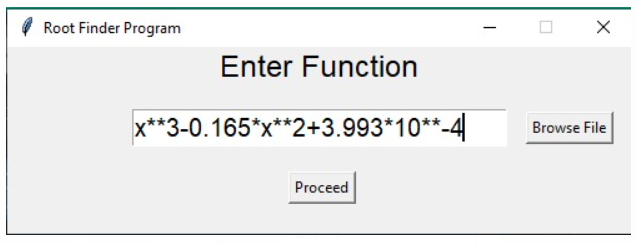


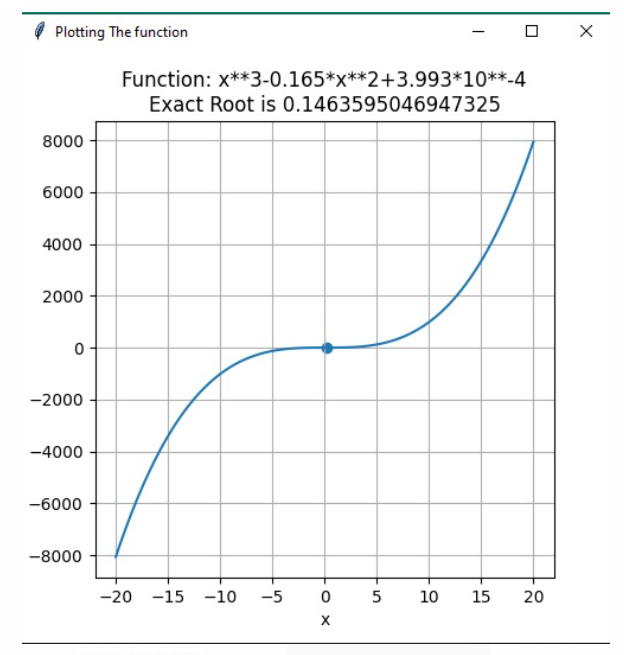
# **Code Result Analysis**

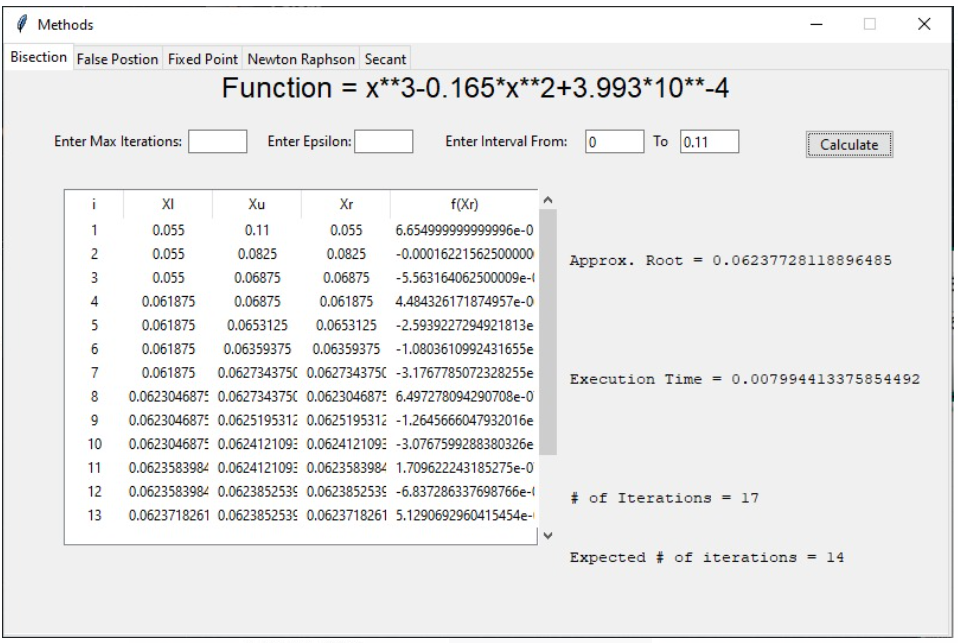
We have picked three function expressions to test on the code from the class lecture based on basic mathematical functions {polynomial, sin, cosine, exponential}.

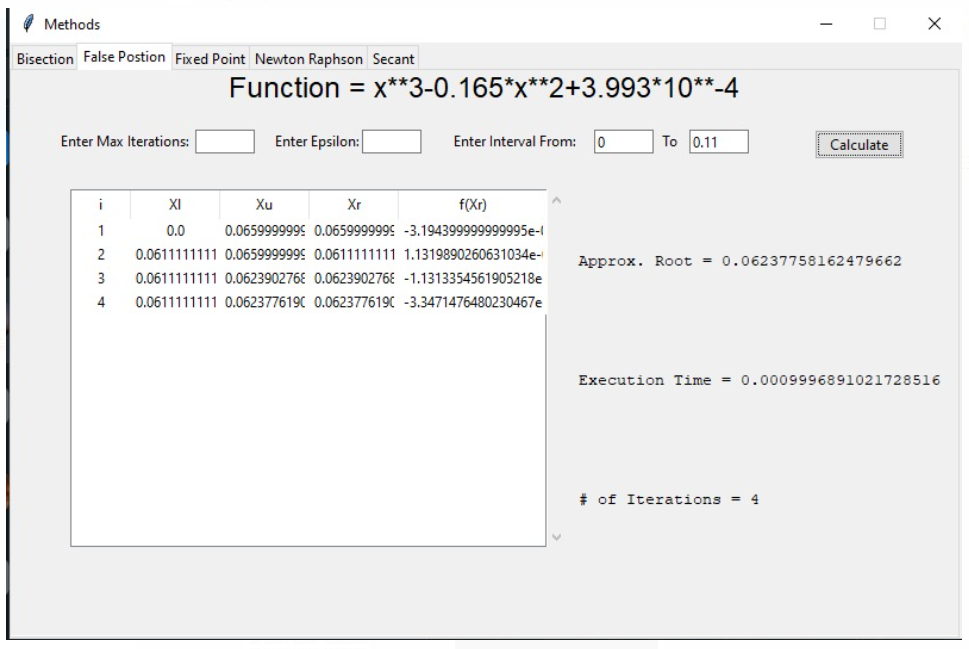
## Expression one

Assuming the user enter the equation in valid way and with no error

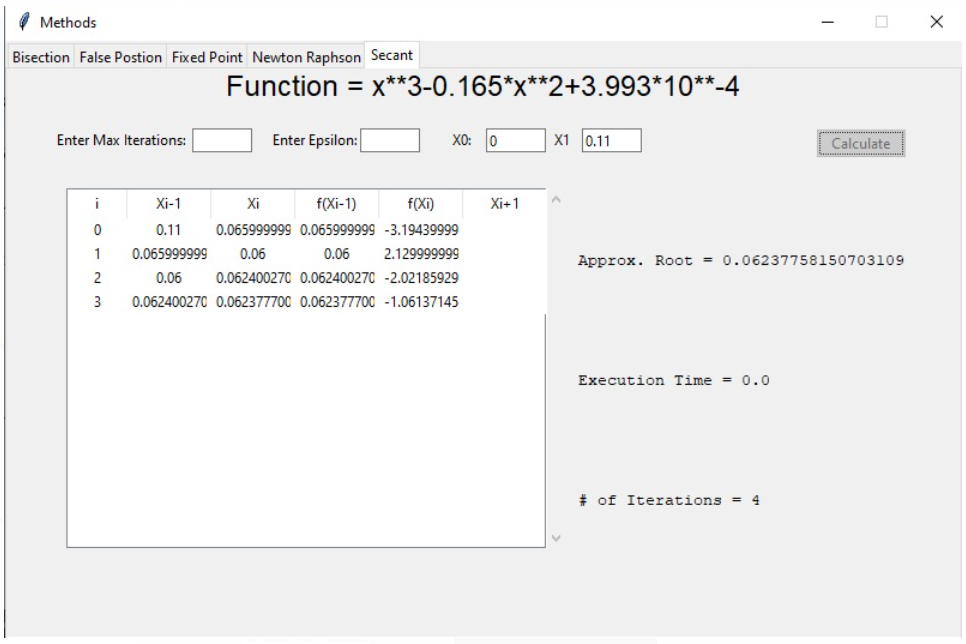




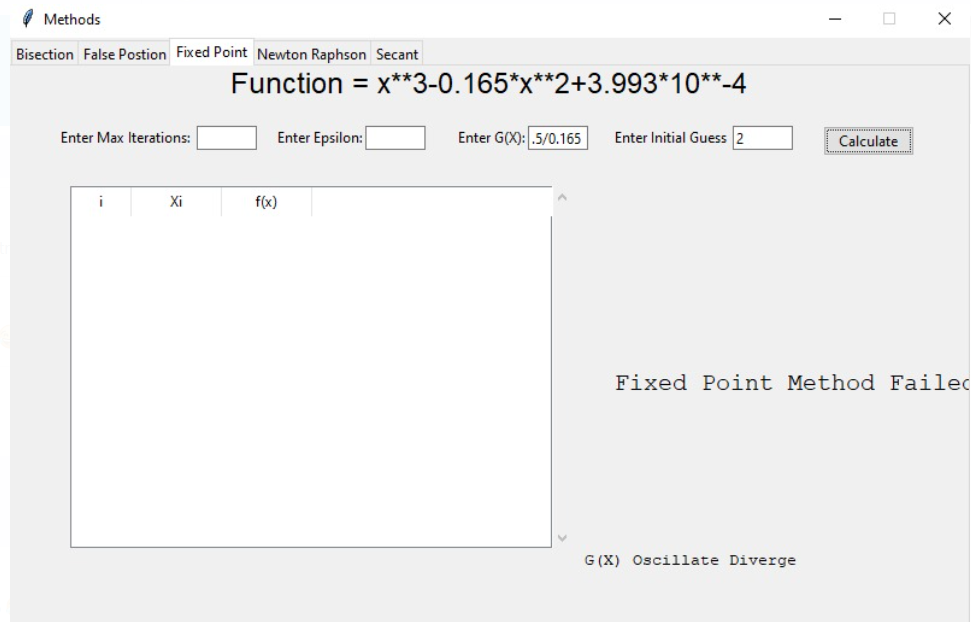






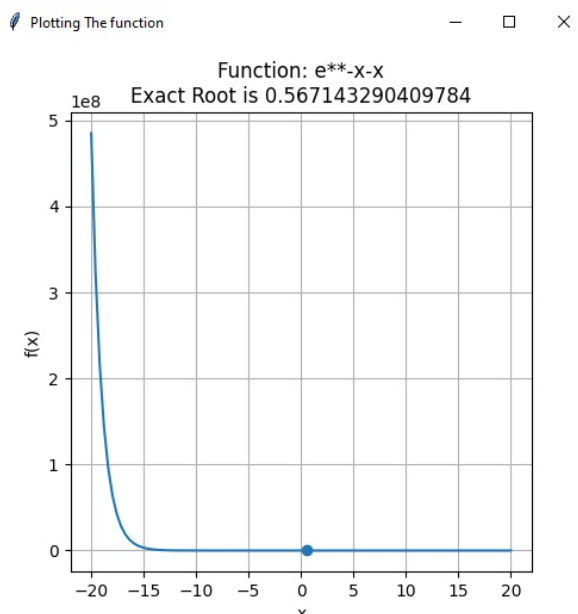


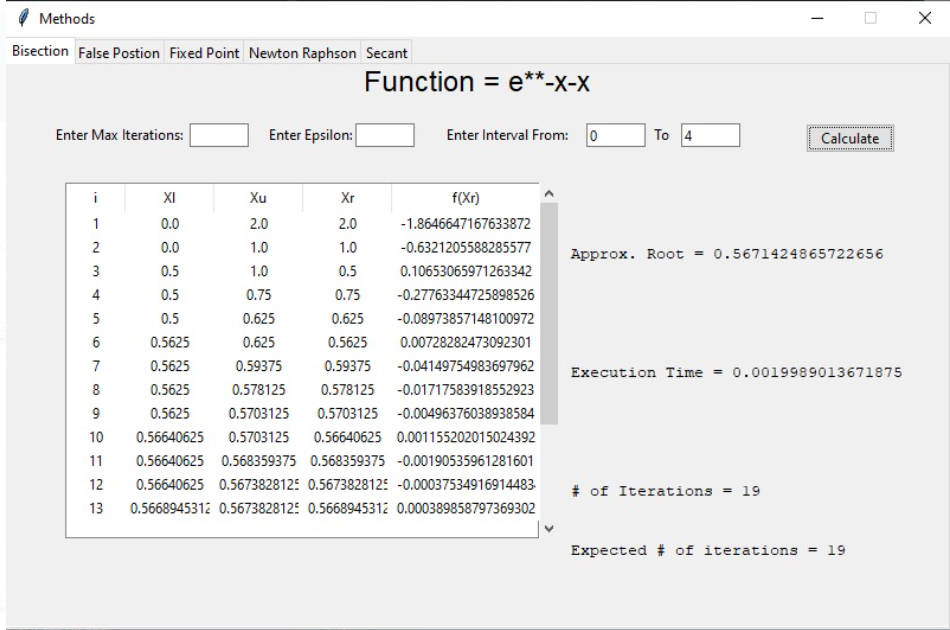
Comment: Since the equation from the third degree and some roots are different 🡪 some methods find get the same root bisection, false position & secant but newton method finds another different root depends on the initial guess, unfortunately the fixed-point method required G(x) with result fail case and couldn’t find the root due to diverge based on the mean-value theorem .

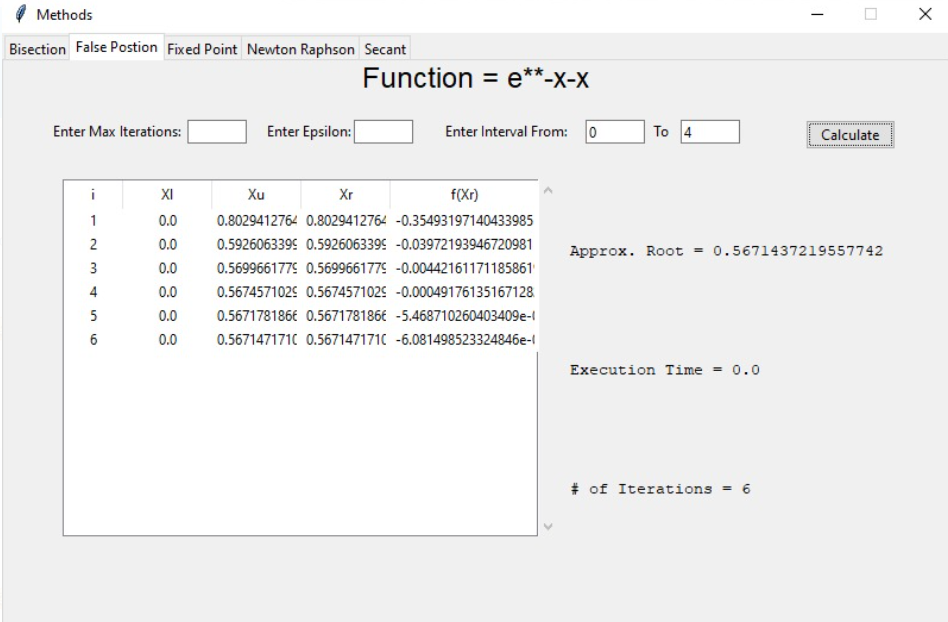


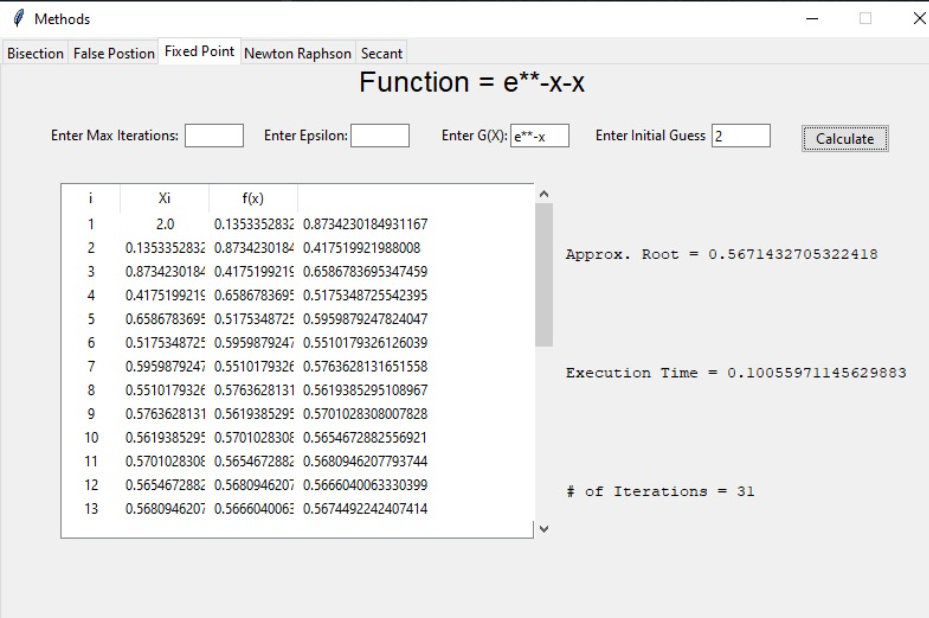
## Expression two

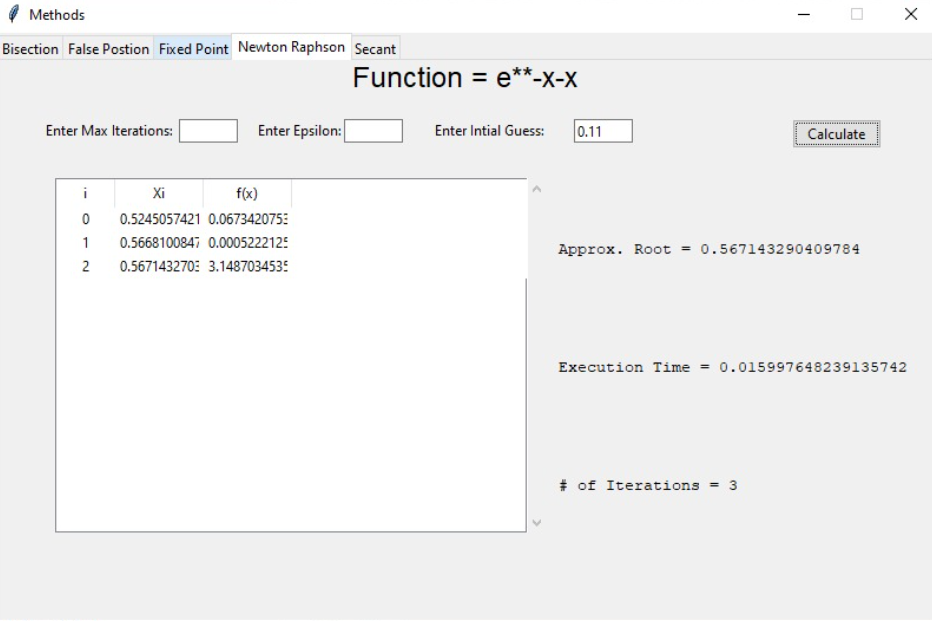


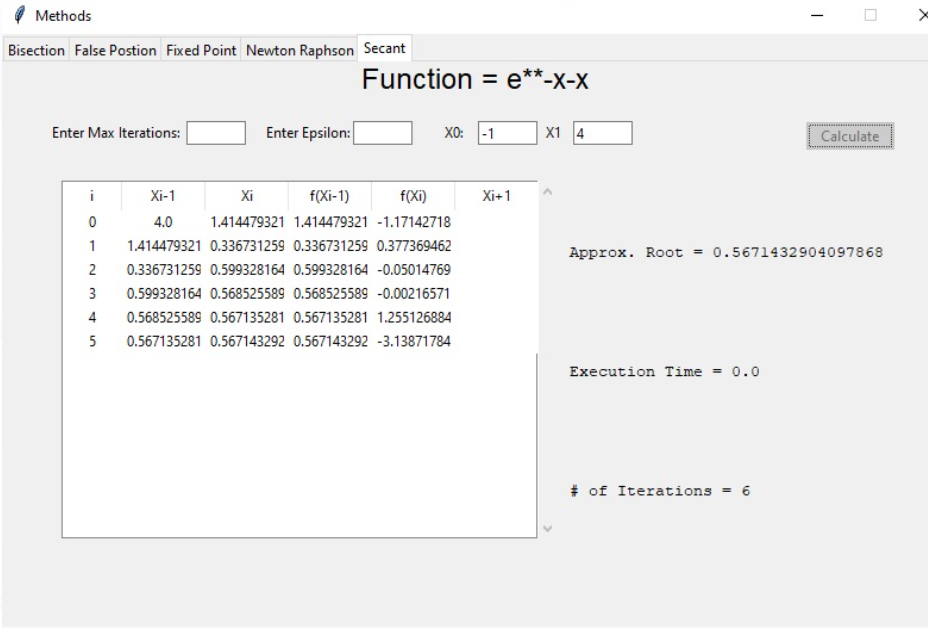






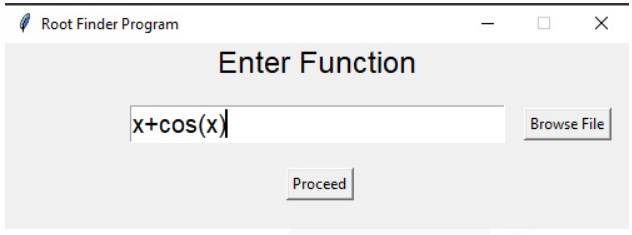


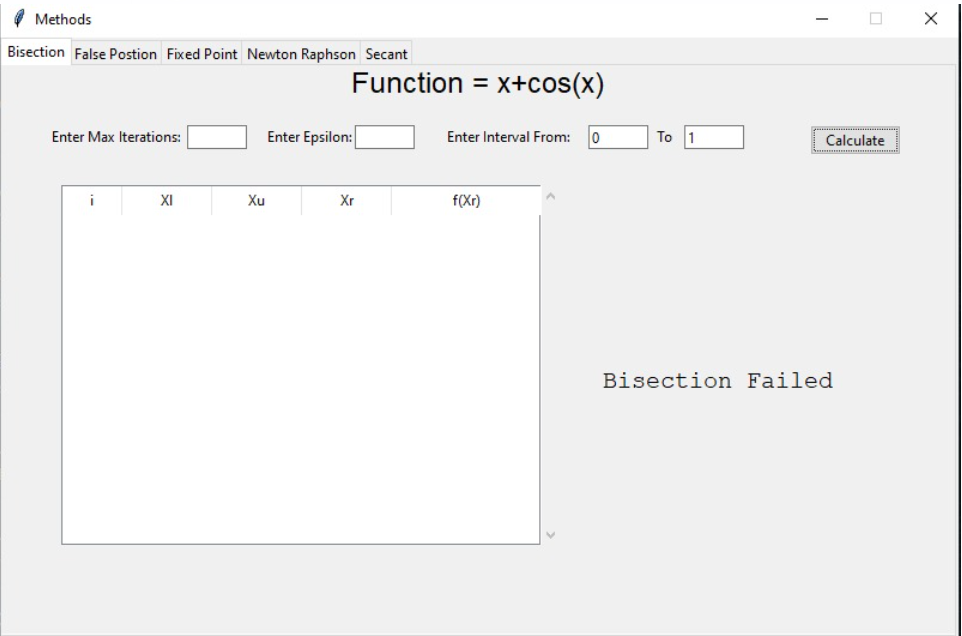


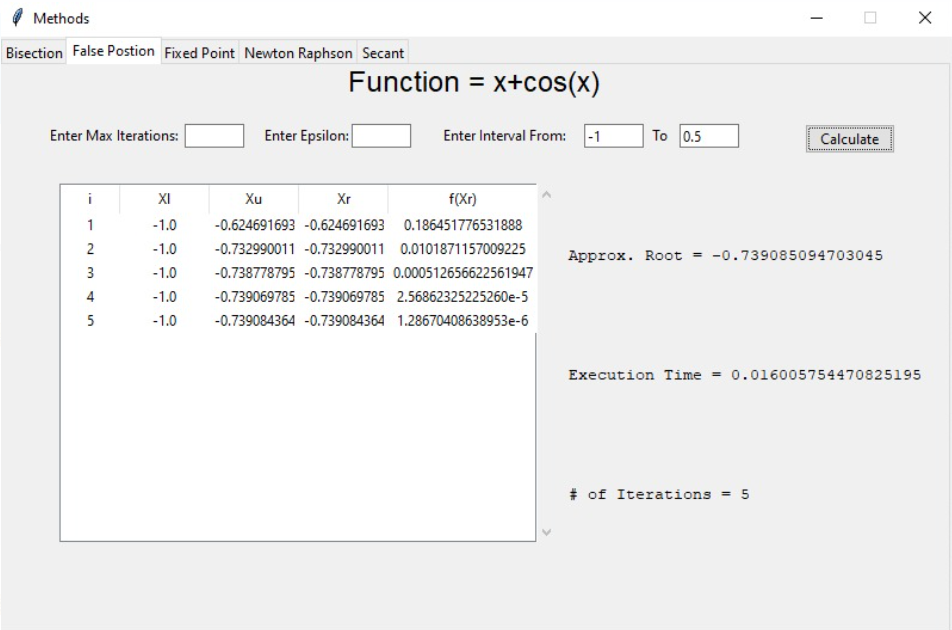


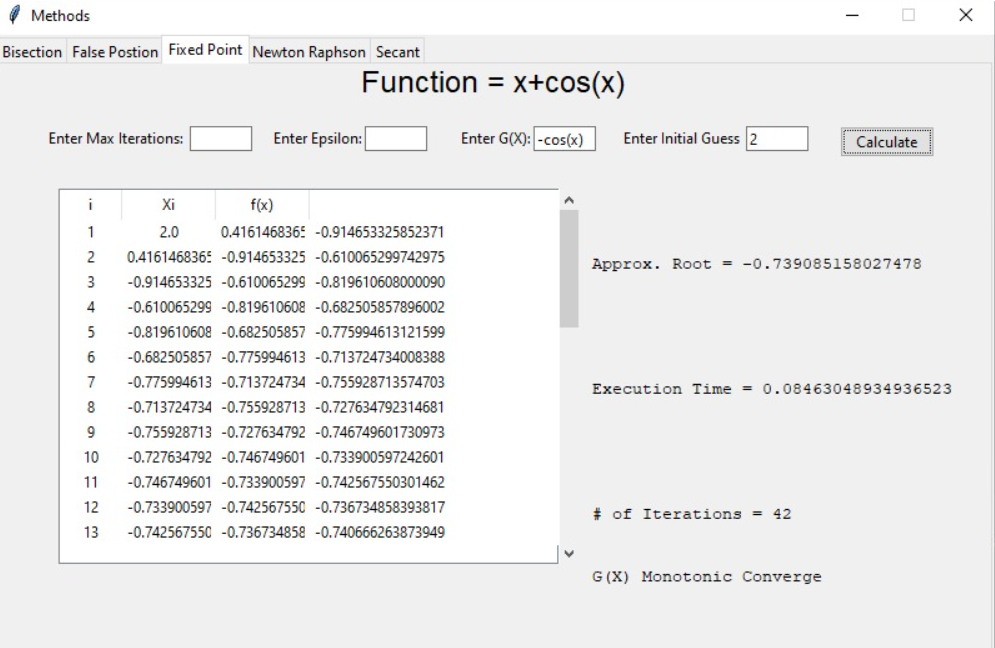
Comment: all the methods returned almost the same root but secant and newton found it faster as we use the graphical plotting for the guessing .

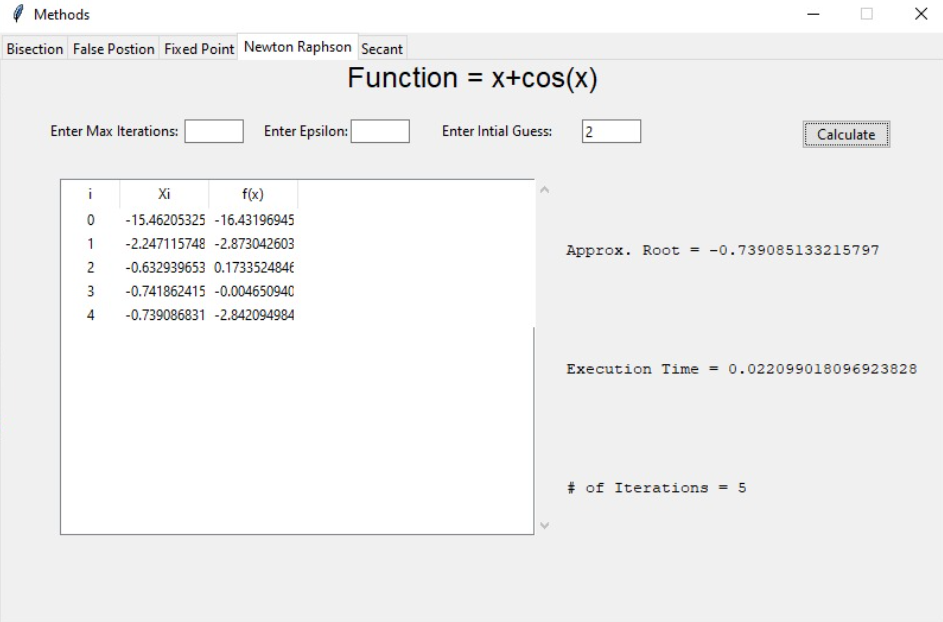
## Expression three

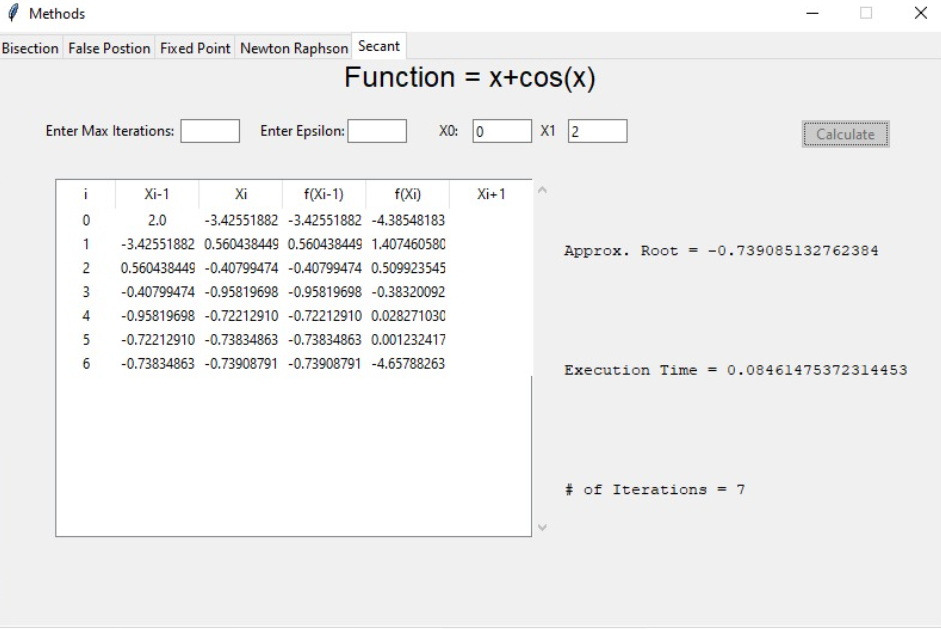












Comment: we have faced problem with plotting the trigonometric function due to the eval python function that has been updated on recent python version that can't compute the lambda expression for plotting so we couldn't predict the interval for the bisection method so no result produced. However, all others methods returned the same root fixed point takes more iteration to find but it converges after all.

# **Problems we faced**

1. Due to some versions update for python we couldn’t plotting any trigonometric functions so the initial guess and any parameters we choose based on the graphical solution was hard to use.
2. We found it really challenging to find the magic function for the fixed-point method to perform the root finder algorithm so we assumed that the user can choose any magic function format can be generated from the expression and embed it as parameter in the algorithm.
3. Some functions for newton – Raphson's method produce division by zero due to the denominator may be number very close from the zero due to the differentiation of the original expression , we tried to prevent dividing by exact zero.

# **Code**

We will provide the code in the submission form and we upload into GitHub [link](https://github.com/ReemAbdelhaleemHassan/Root-Finder-Program.git) .