



# Numerical analysis

Part I

---

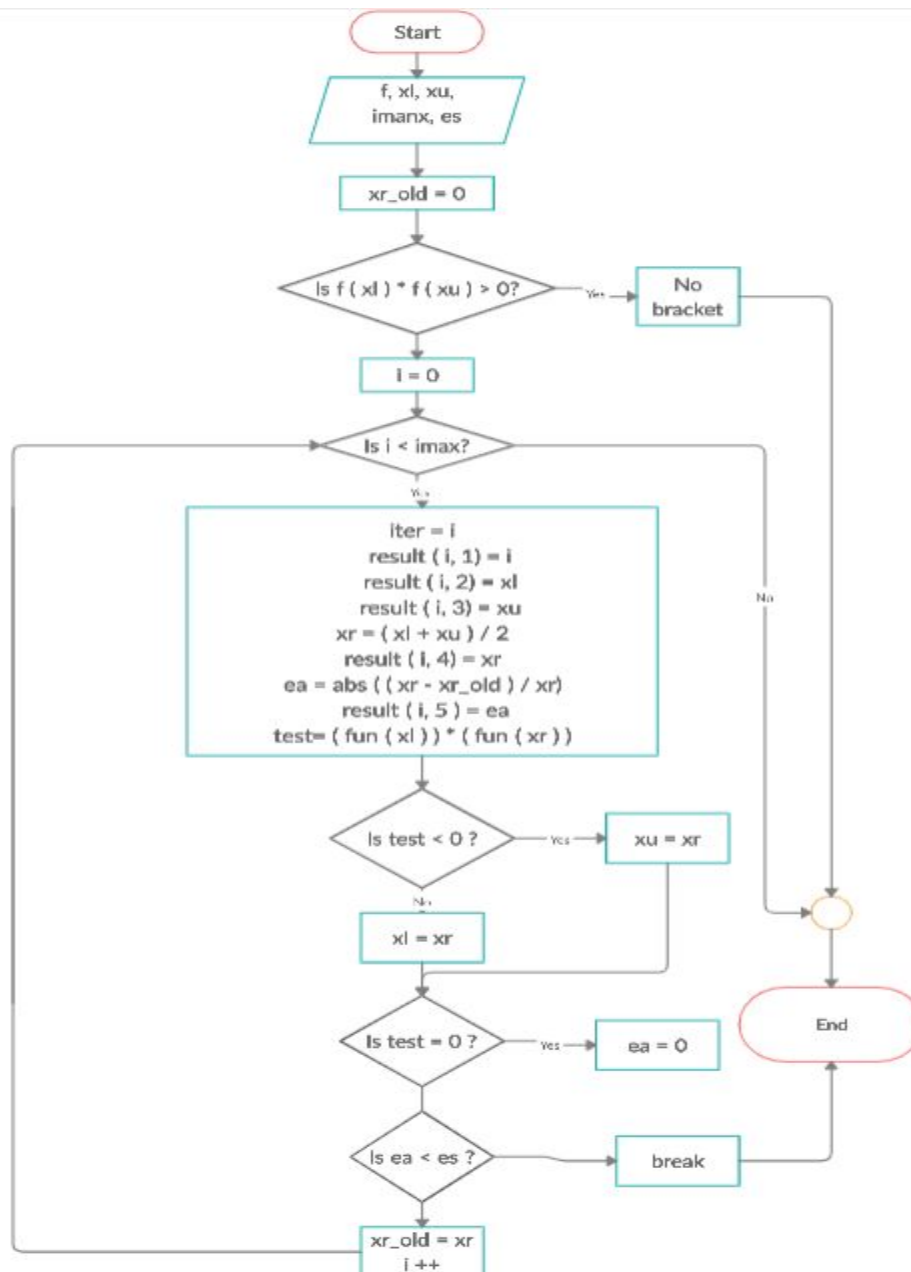
<b>Afnan Mousa</b>	<b>15</b>
<b>Enas Morsy</b>	<b>20</b>
<b>Sara Mohammad</b>	<b>31</b>
<b>Shimaa kamal</b>	<b>34</b>
<b>Nada Fathy</b>	<b>68</b>

## Attention please!!!

- This project basically is handled by matlab versions 2018, 2019 so it may cause troubles running this project with older versions.
- Bonus here is done “reading inputs from files”
- User guide is supplied at the end of this report with sample runs

# 1. Bisection

## Flowchart



## Data structure

### I. Array two dimension

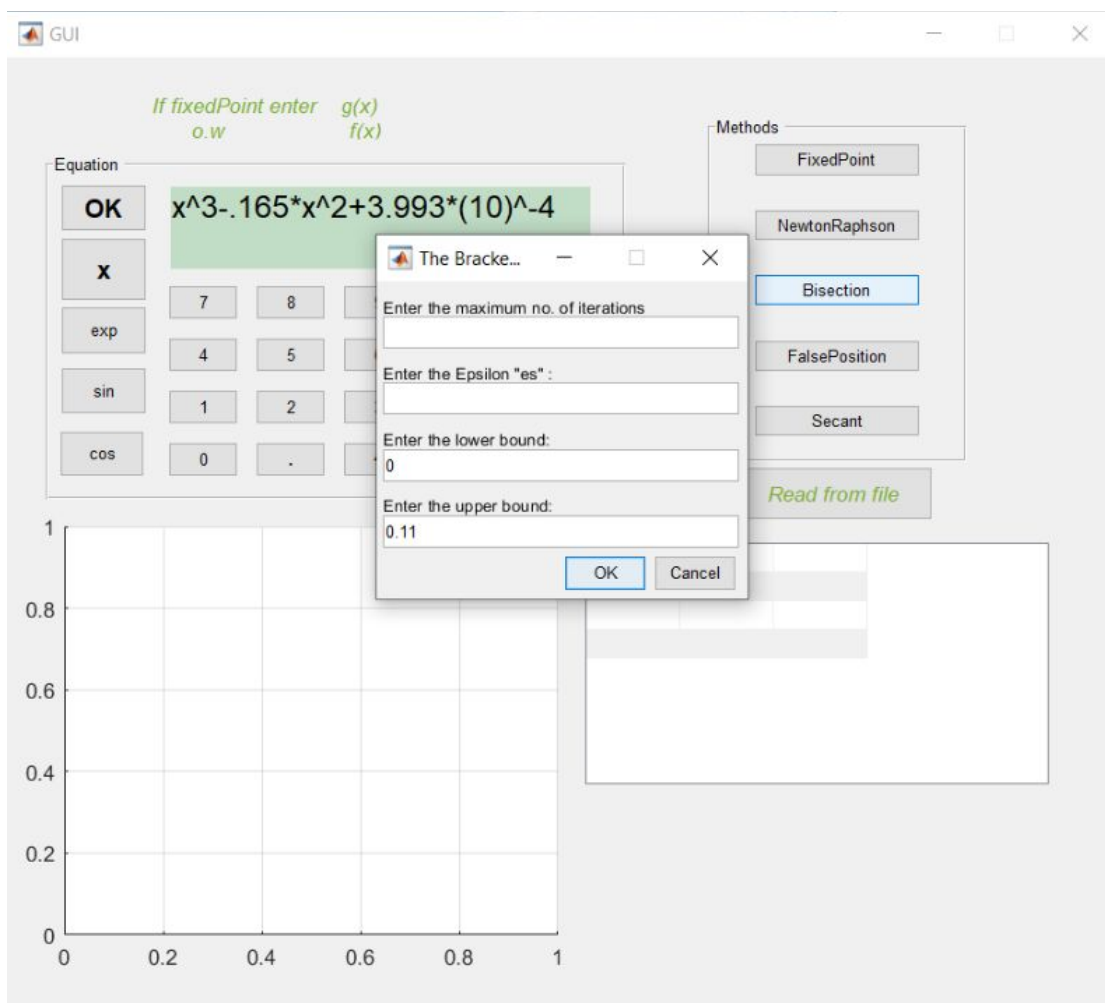
Use to create the table which contains all the data to be shown to the user in GUI.

This table contains every iteration with each calculation .

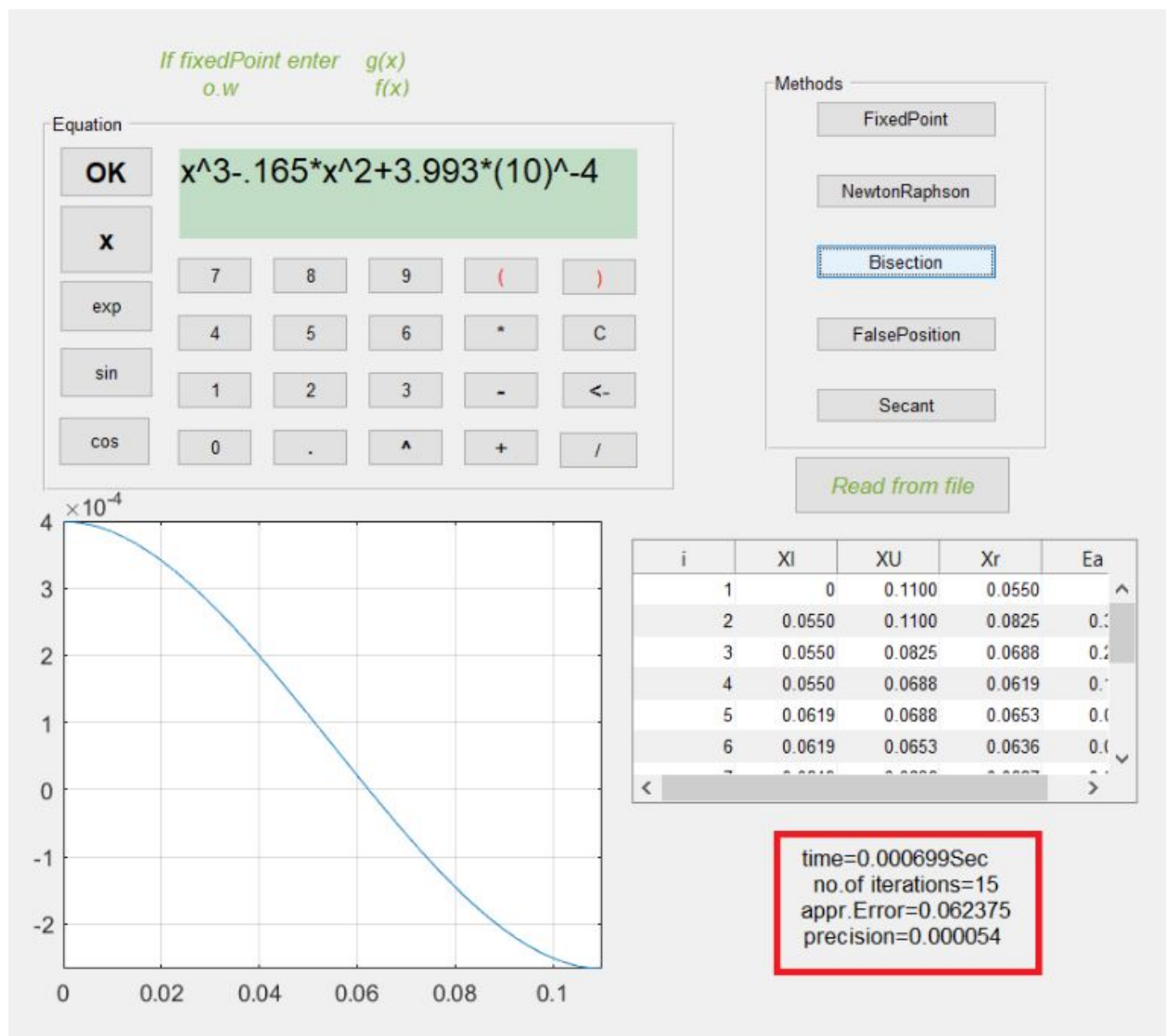
## Analysis for the behavior

### I. example one

Function with default maximum no. iteration & the estimated error:

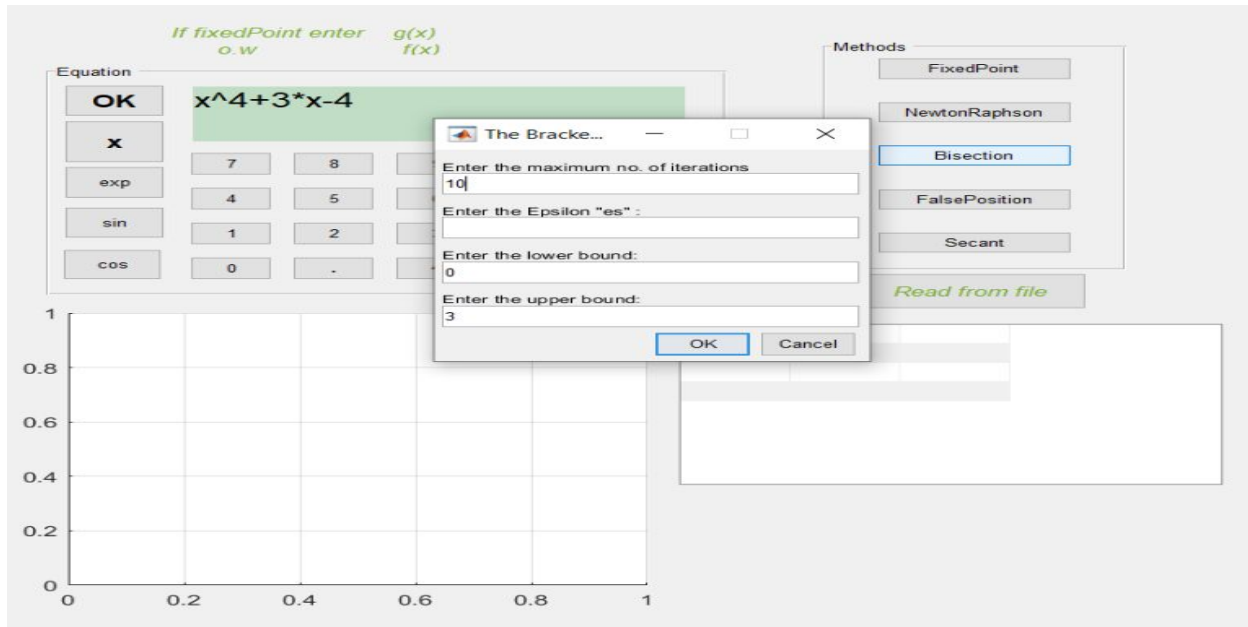


Output :

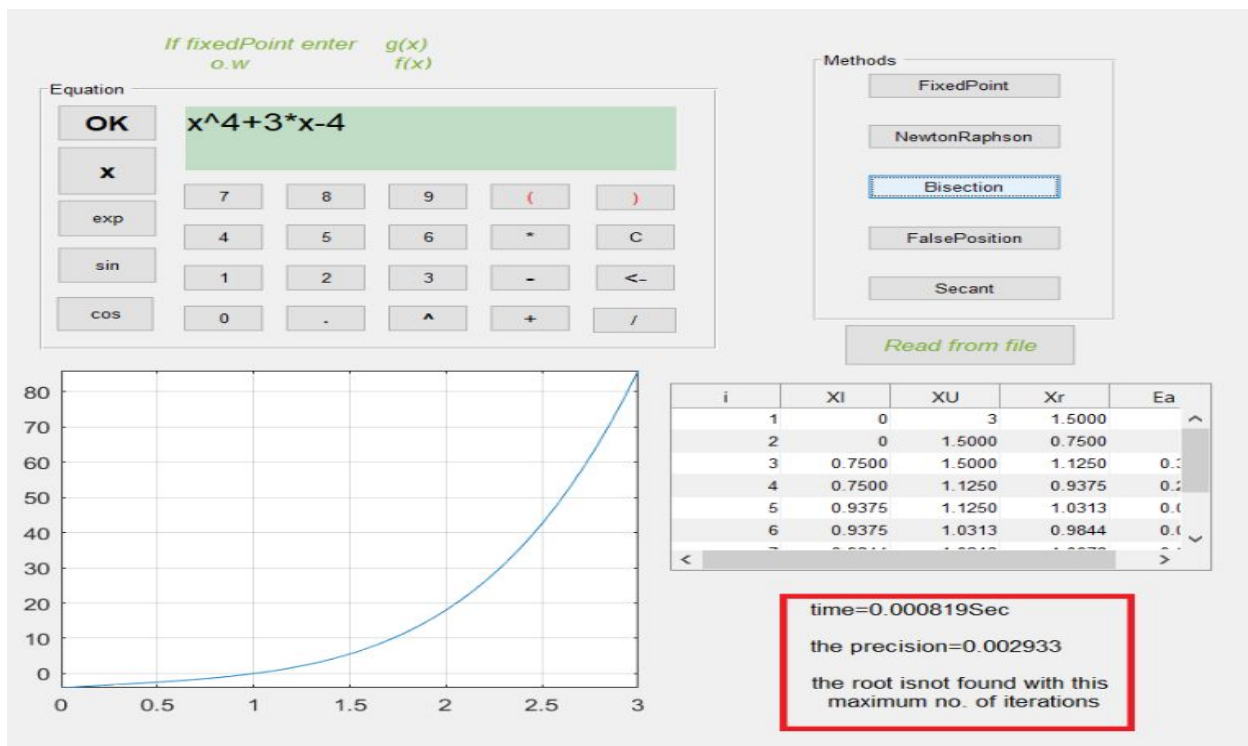


## II. example two

No. of iteration is bigger than the maximum iterations

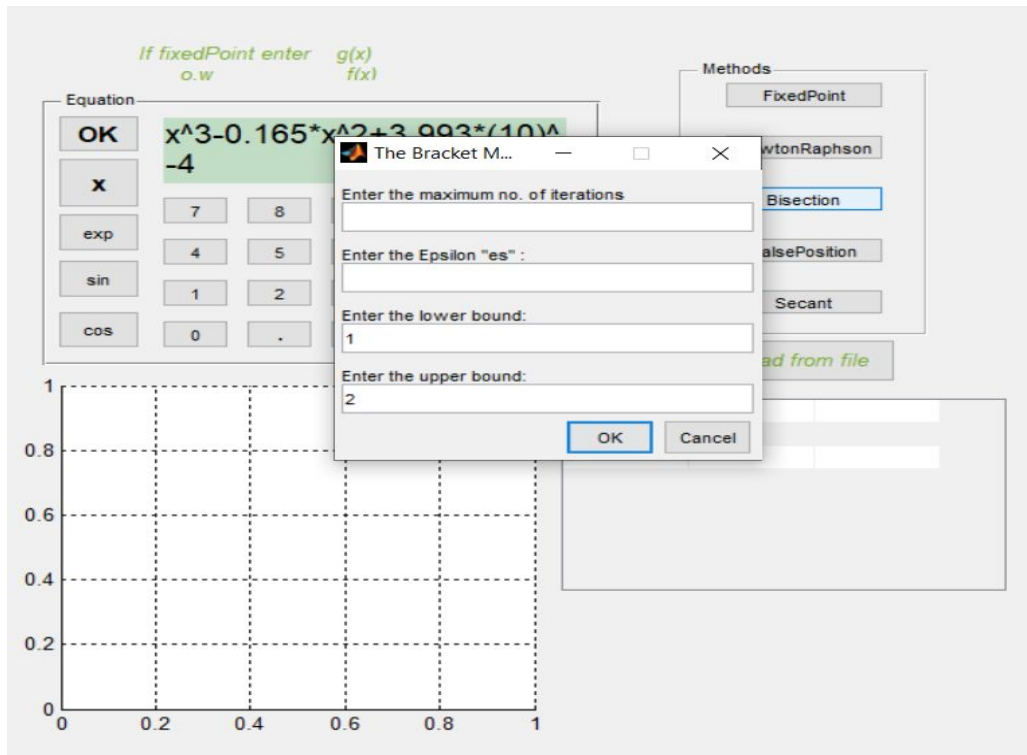


Output:

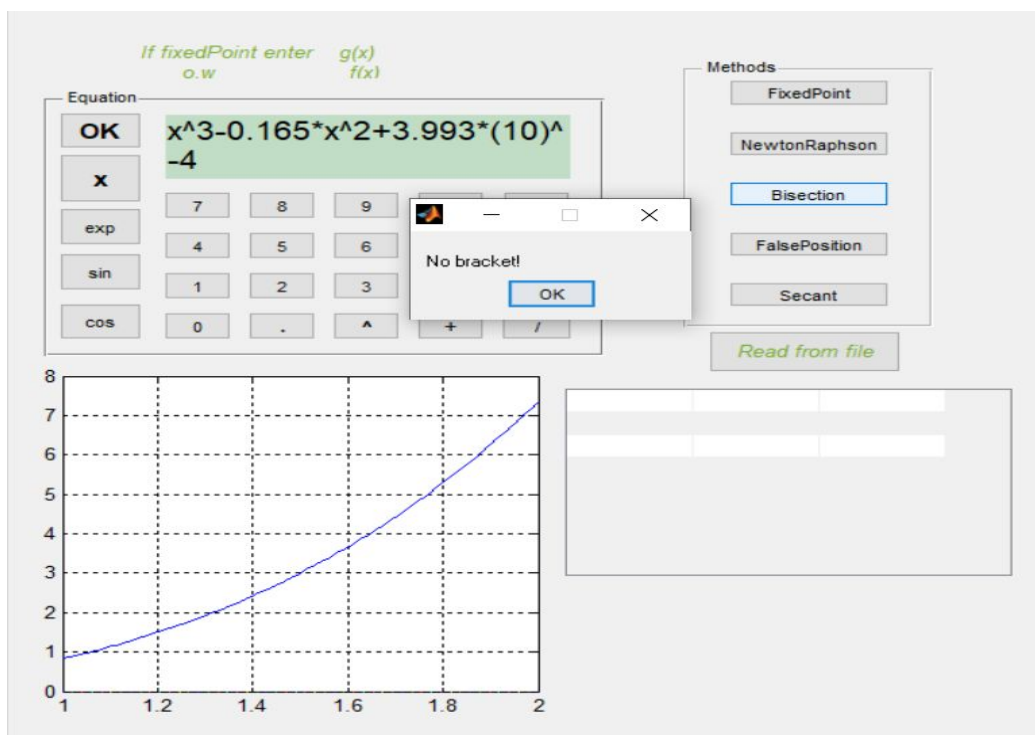


### III. example three

The root isn't between xl and xu



### Output



## Conclusion

Bisection method is easy and always finds the root but it is also slow and if one of the initial guesses is close to the root, the convergence becomes slower.

## Problematic functions

- One of the problematic functions :

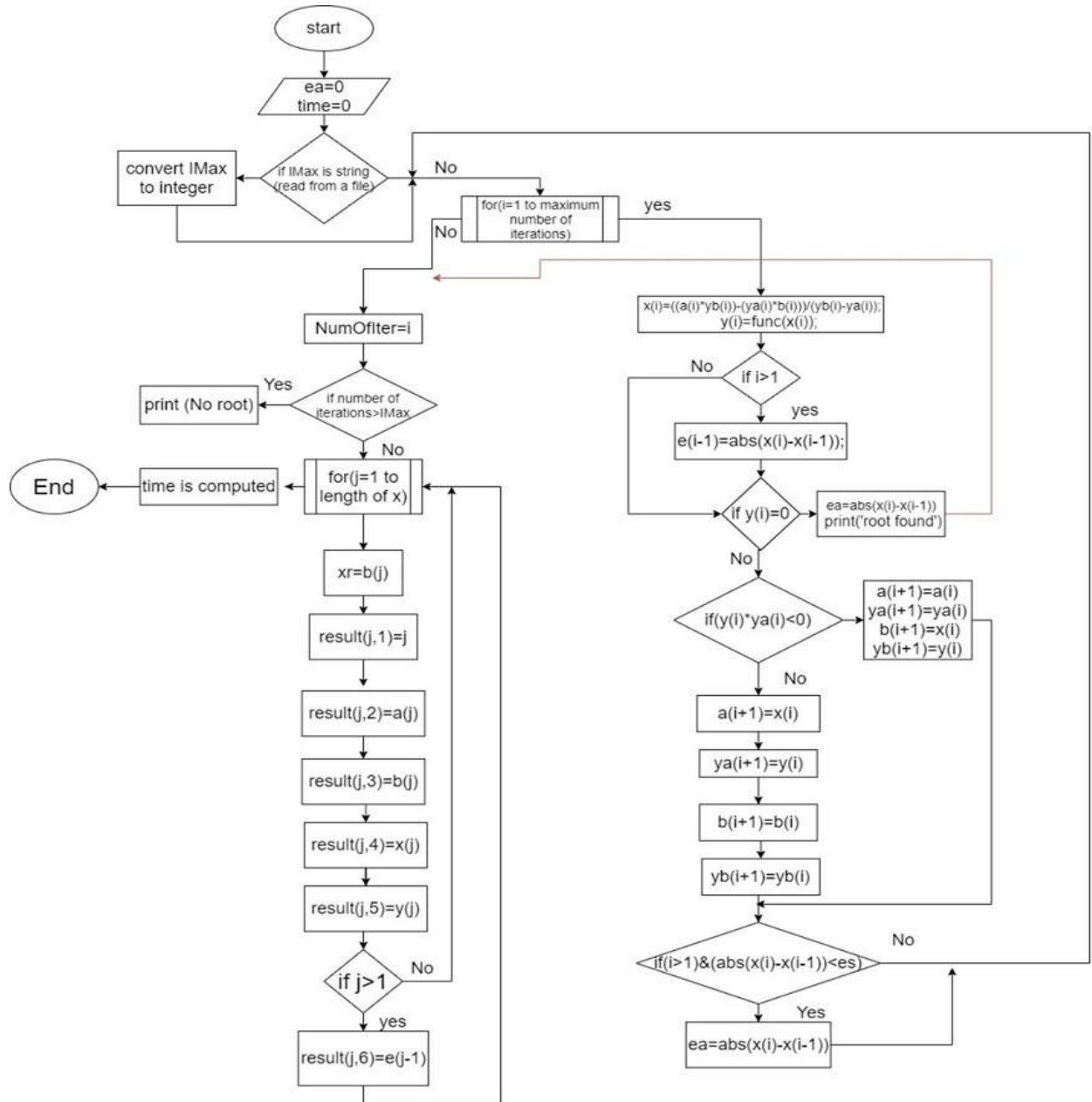
Convergence slowly, solve it :

- by using another Open method such as Newton Raphson, secant.
- Need to find initial guess for  $x$  lower and upper.
- If a function  $f(x)$  is such that it just touches the  $x$ -axis it will be unable to find the lower and upper guesses.
- Function changes sign but root does not exist.



## 2.False-position

### Flowchart



## Data structure

### I. Array two dimension

Use to create the table which contains all the data to be shown to the user in GUI.  
This table contains every iteration with each calculation .

### II. Array one dimension

To store the value of XR after each iteration .

Another one to store the value of function ( XR )after each iteration .

Another one to store the change in the upper and lower bound and then corresponding values ( f(x lower ) , f(x upper)).

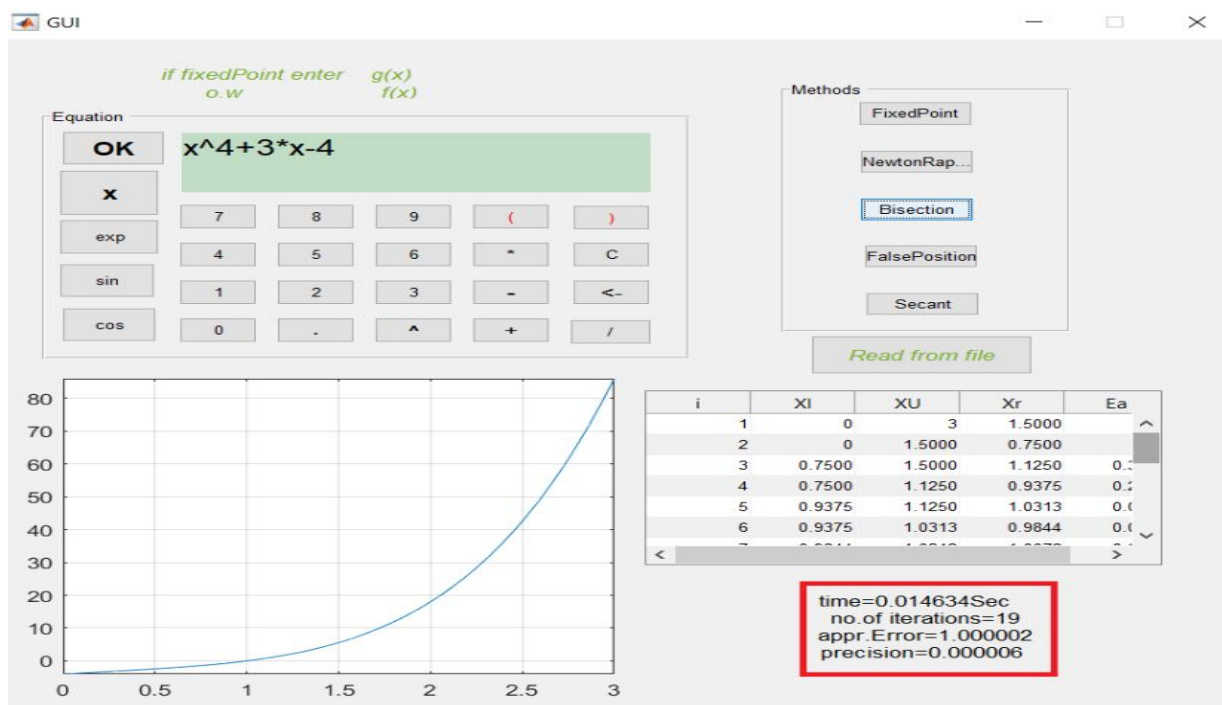
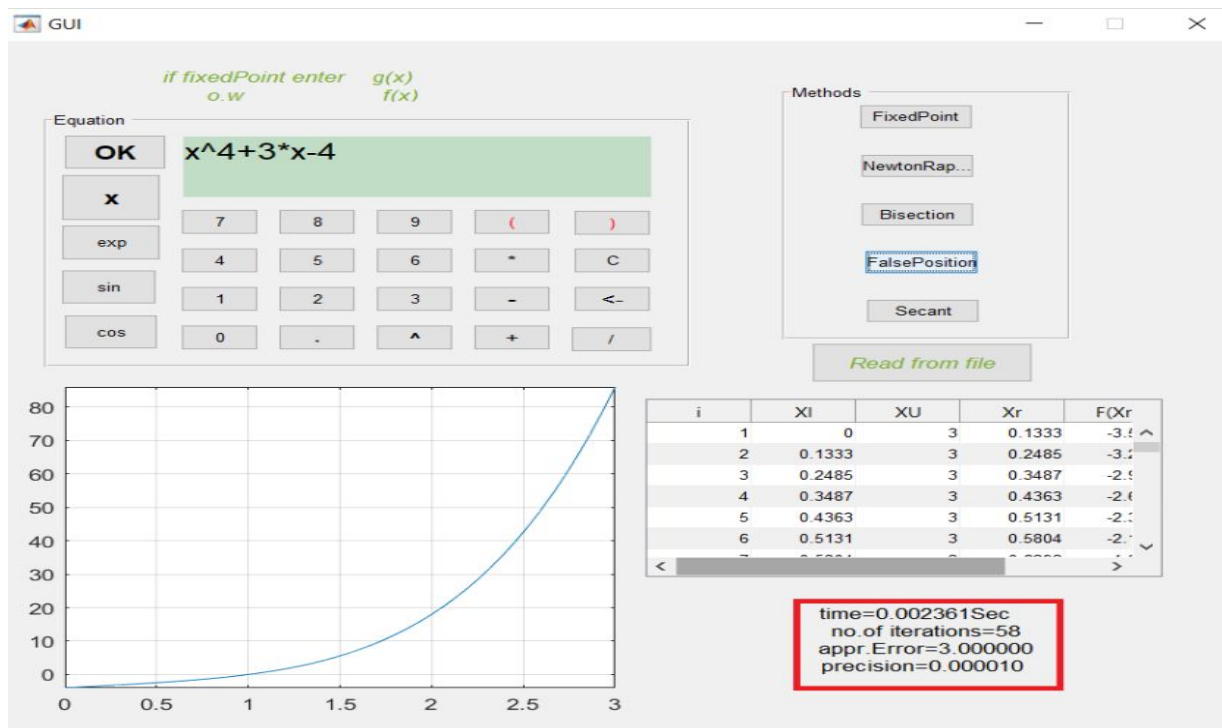
## Analysis for the behavior

### I. example one

In case of this equation:  $X^4+3X-4$

Lower bound=0 , Upper bound=3 ,Maximum number of iterations=100 ,precision: 0.00001 .

False position method takes 58 iterations to reach the root. However ,the bisection method takes only 19 iterations .



## II. example two

In case of this equation:  $X^3 - 3X + 1$

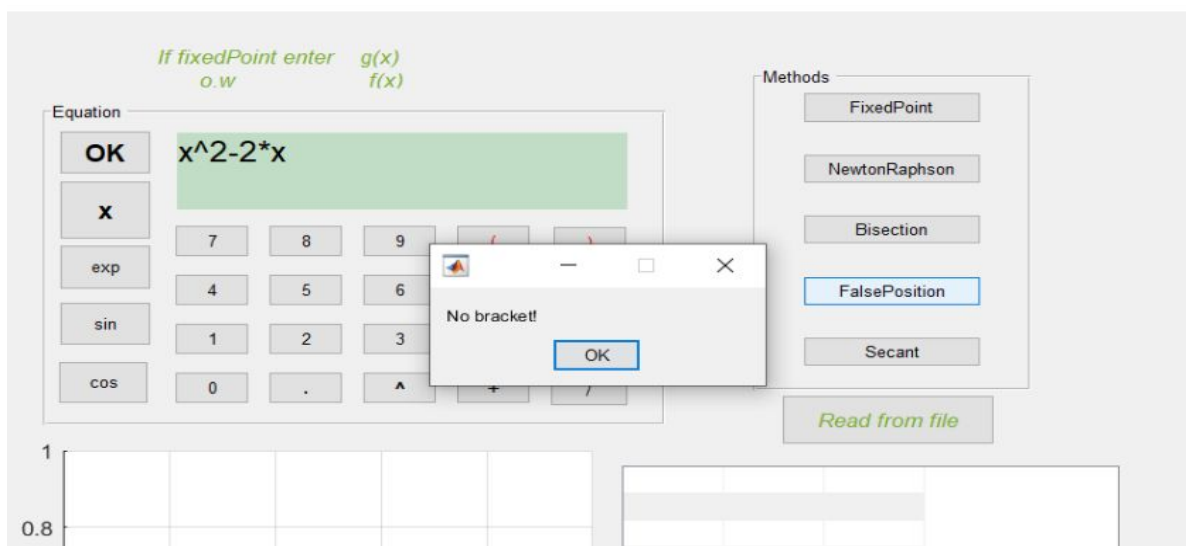
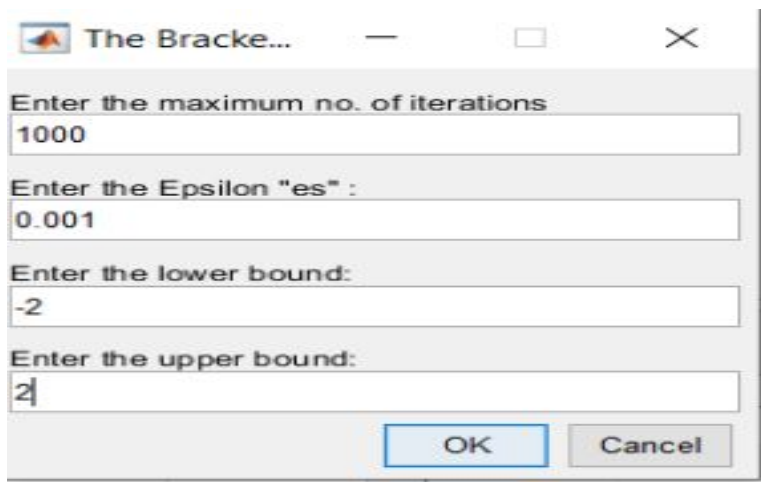
Lower bound=-2 , Upper bound=2 ,Maximum number of iterations=100 ,precision: 0.001 .

False position method takes 5 iterations to reach the root. However ,the bisection method takes only 12 iterations .

## III. example three

When there are even number of roots or no root between lower and upper bounds

Function :  $x^2 - 2x$  , lower bound=-2 , upper bound=4



## conclusion

False position method is slower than bisection method in some cases (**flat functions**).

False position method is faster than bisection method in some cases (**when the slope intersects x axis near the root**).

## Problematic functions

- One of the problematic functions :

Is the method slower than the open method in case convergence , solve it :

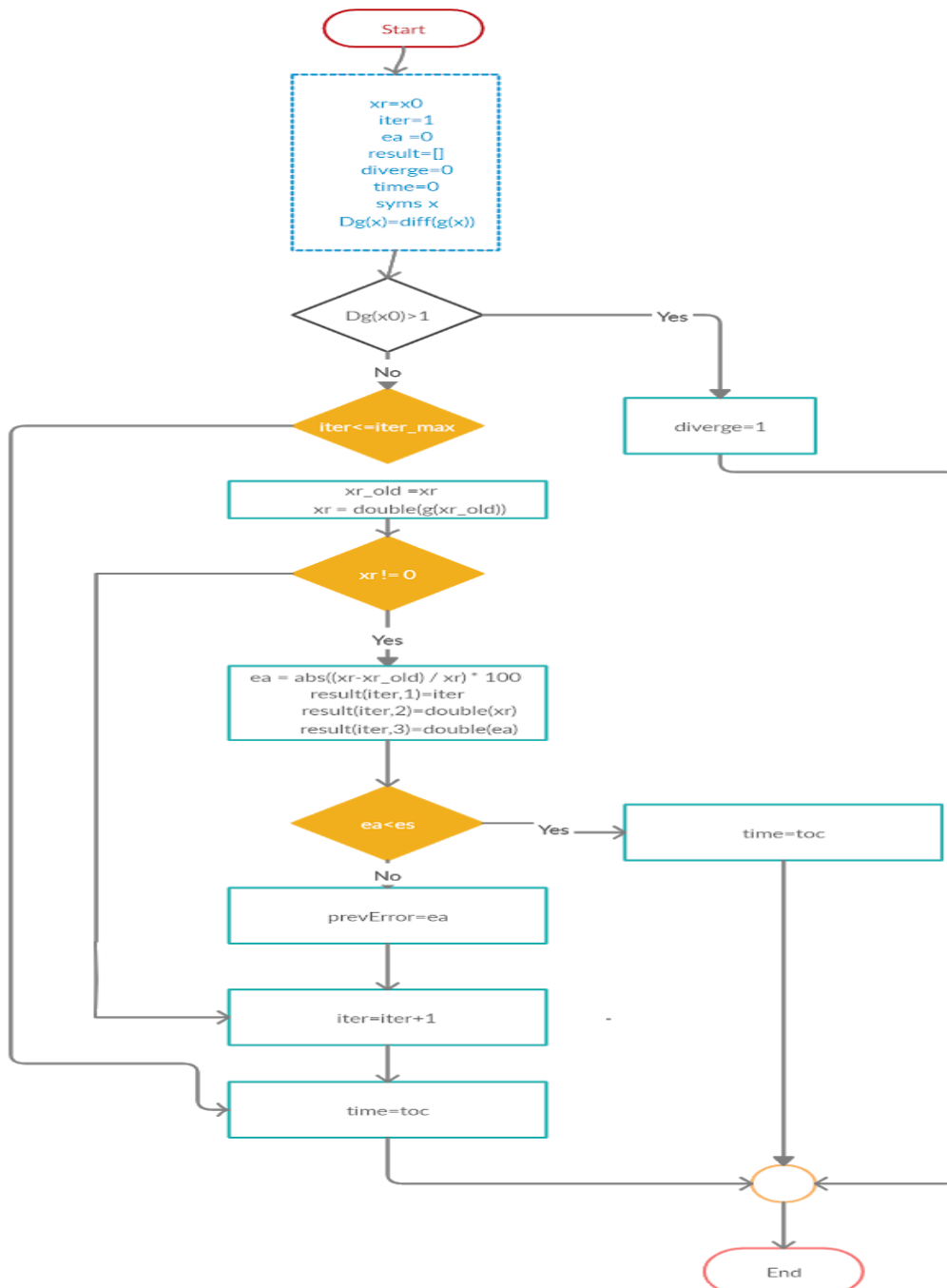
- by using another Open method such as Newton Raphson ,secant .
- Need to find initial guess for x lower and upper .
- Convergence slower than bisection method .

To solve by :

- We can use the bisection method in the first iteration to abbroch the root and then use false position.

### 3.Fixed point

#### Flowchart



## Data structure

### I. Array two dimension

Use to create the table which contains all the data to be shown to the user in GUI.

This table contains every iteration with each calculation .

## Analysis for the behavior

### I. example one

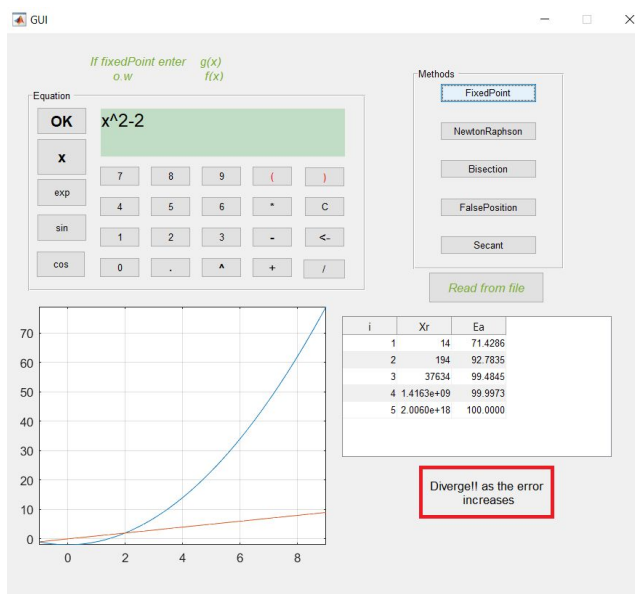
In case of this equation:  $g(X) = x^2 - 2$

Fixed point method diverges after a few iterations !

$$g'(c) = \frac{g(\alpha) - g(X)}{\alpha - X} \quad \text{where } \alpha \text{ is the root, } x < c < \alpha$$

So, X is greater than g(X) and  $|g'(c)|$  is greater than 1 . than is why the error increases and the method diverges.

Sample run with x initial equals to 4

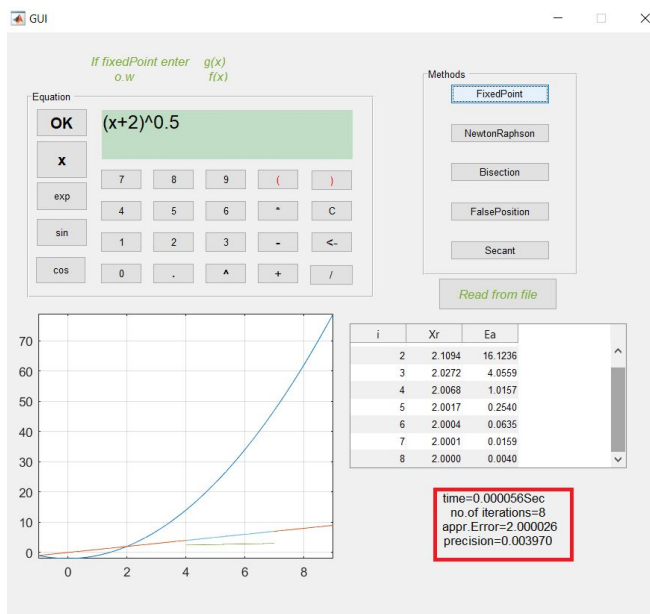


## II. example two

In case of this equation:  $g(X) = \sqrt{x+2}$

Fixed point method converges, because  $g(X)$  is greater than  $X$ . And according to the previous relation of  $g'(c)$ ,  $|g'(c)|$  is smaller than 1. so, the error decreases and the method converges.

Sample run with  $x$  initial equals to 4



## III. example three

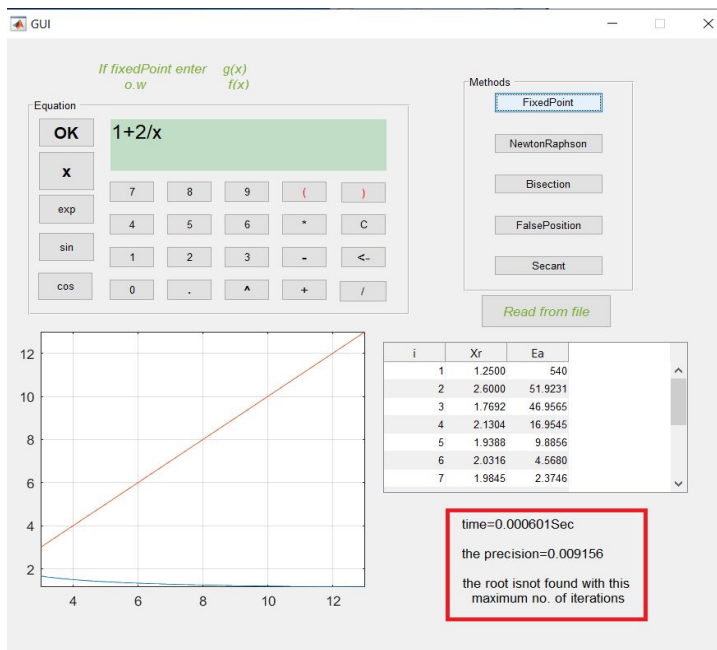
In case of this equation:  $g(X) = 1 + \frac{2}{x}$

Fixed point method converges but it's oscillating.

The function  $g(X)$  is greater than  $X$ . And according to the previous relation of  $g'(c)$ ,  $|g'(c)|$  is smaller than 1. so, the error decreases and the method converges. However, it's oscillating because  $g'(c)$  is negative ( $x$  is always greater than  $\alpha$ )

Sample run with max iteration=15 and  $x$  initial=8





## conclusion

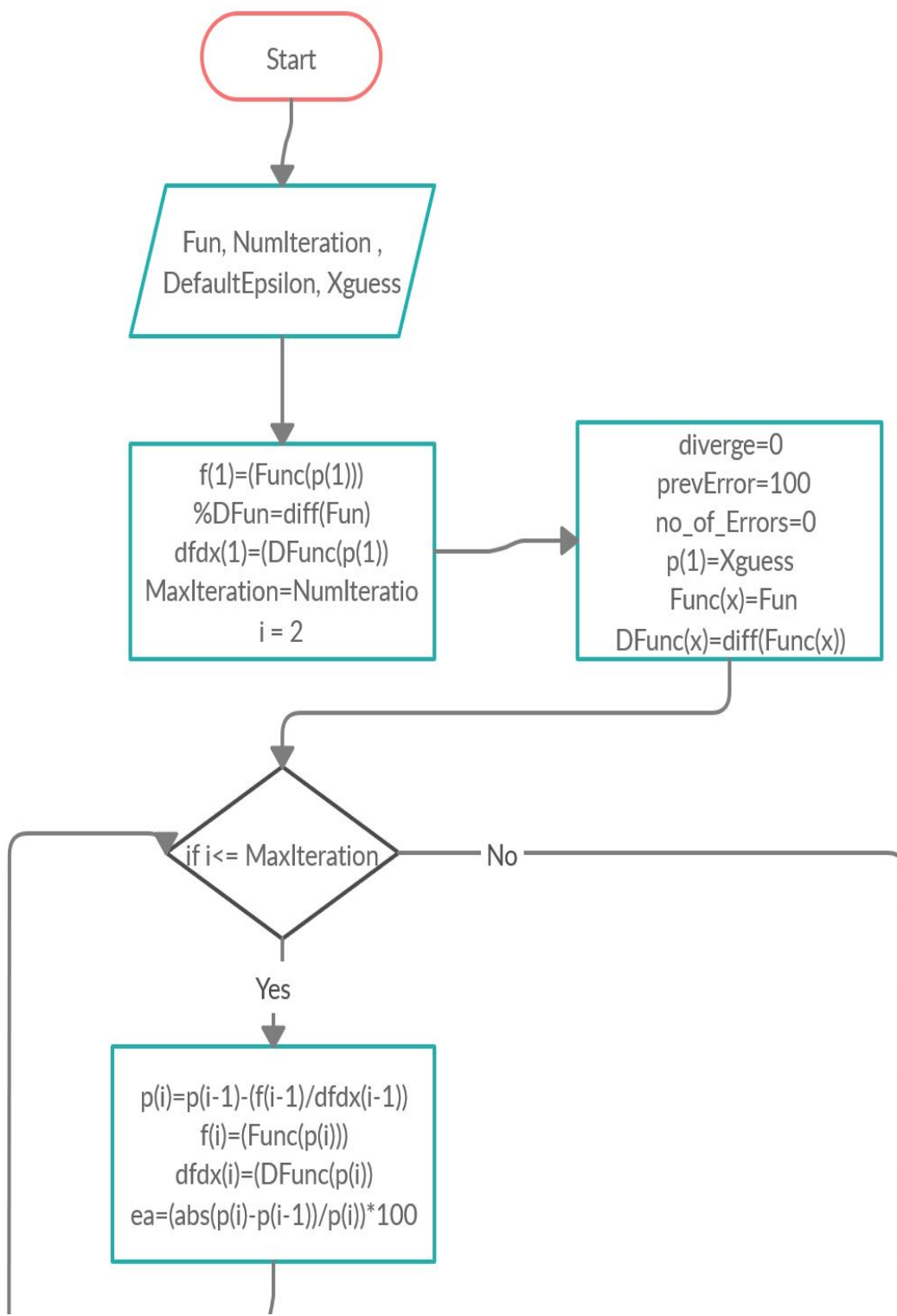
Fixed point iteration method diverges or converges depending on the auxiliary function  $g(X)$ .

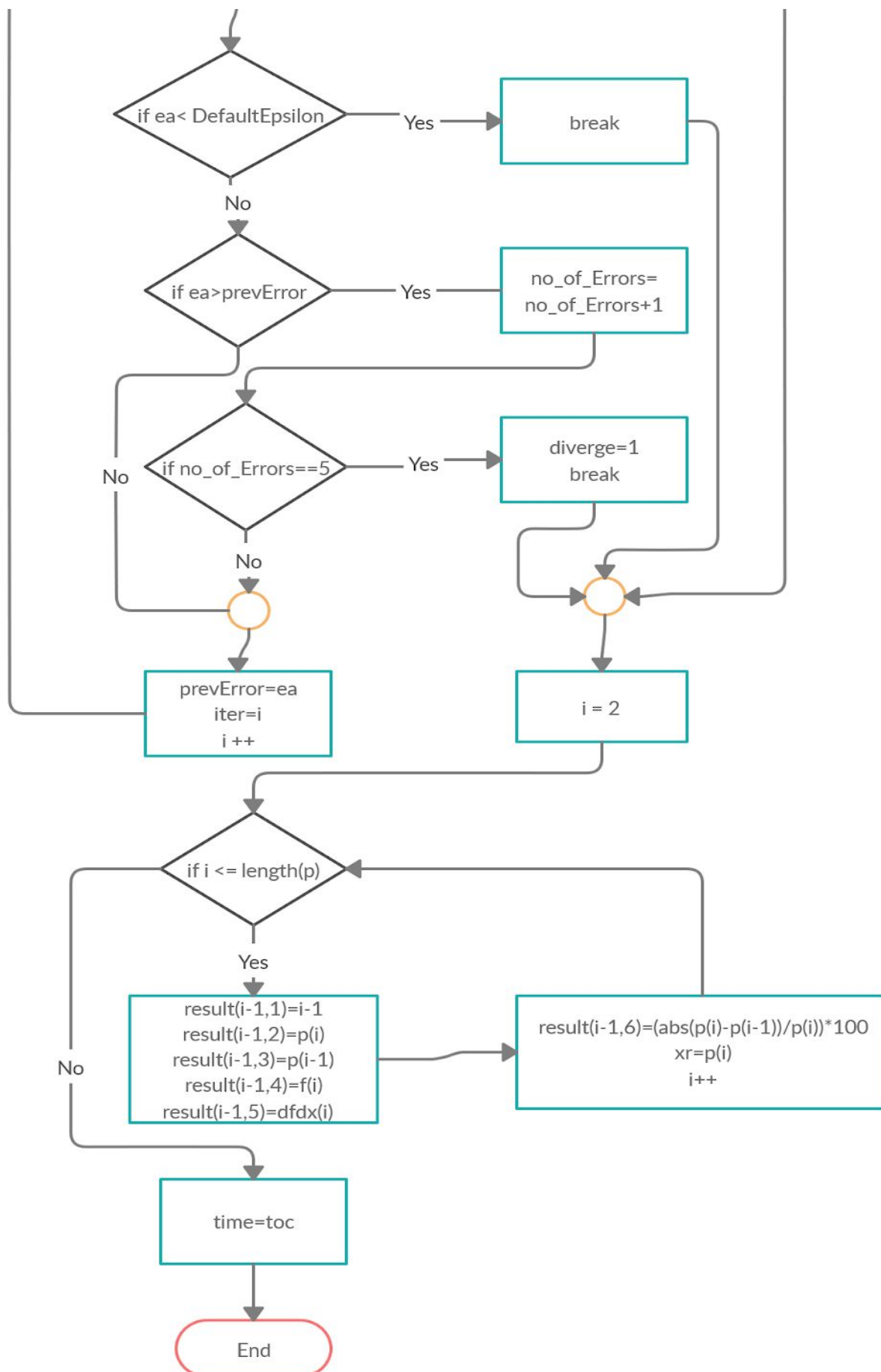
## Problematic functions

- One of the problematic functions :  
If  $X_r$  is equal zero , solve it by skipping this iteration .
- $g(x)$  may diverge or converge so check if  $g'(x_{\text{initial}}) > 1$  then it will diverge .

## 4. Newton-Raphson

### Flowchart





## Data structure

### I. Array two dimension

Use to create the table which contains all the data to be shown to the user in GUI.  
This table contains every iteration with each calculation .

### II. Array One Dimension

To store the value of  $X(i)$  after each iteration .

Another one to store the value of function (  $x(i)$  )after each iteration .

Another one to store the value of function of differentiation(  $x(i)$  )after each iteration .

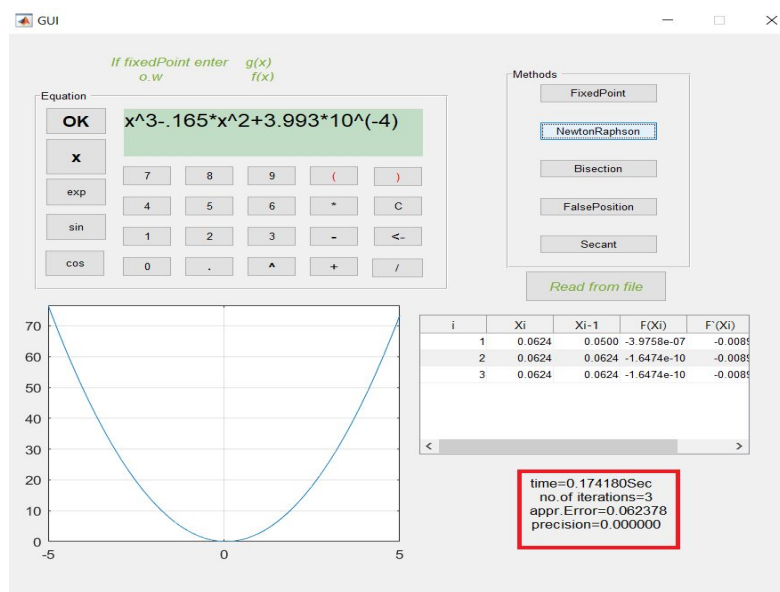
## Analysis for the behavior

### I. example one

➤ If the equation is :

$$f(x) = x^3 - 0.165x^2 + 3.993 \times 10^{-4}$$

- Assume the initial guess of the root  **$x=0.05$**
- So the number of iterations to find the root is 2 Which is less than the number of iterations in case of bisection method which equal **15**.

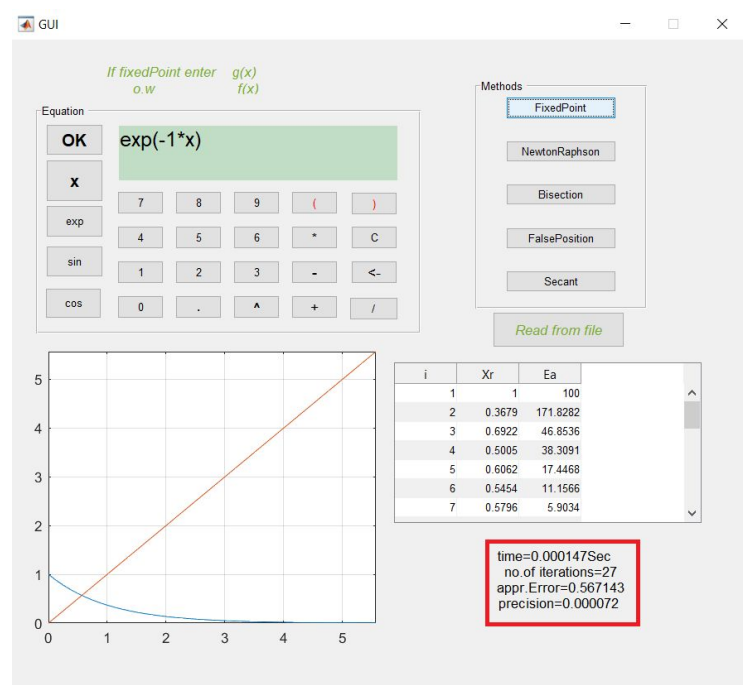
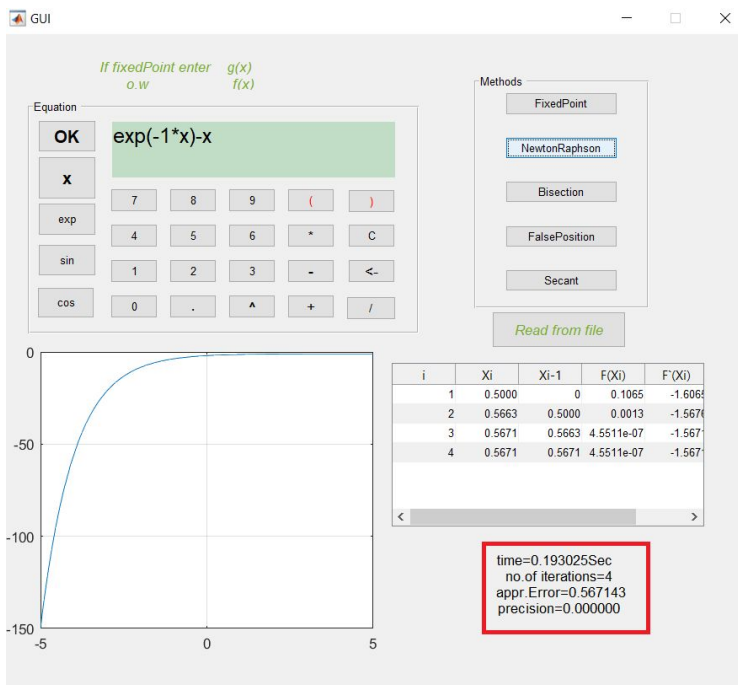


## II. example two

- If the equation is :

$$f(x) = e^{-x} - x = 0.$$

- Assume the initial guess of the root  **$x=0$**
- So the number of iterations to find the root by Newton Raphson is **4** Which is less than the number of iterations in case of Fixed Point method which equals **27** iteration.

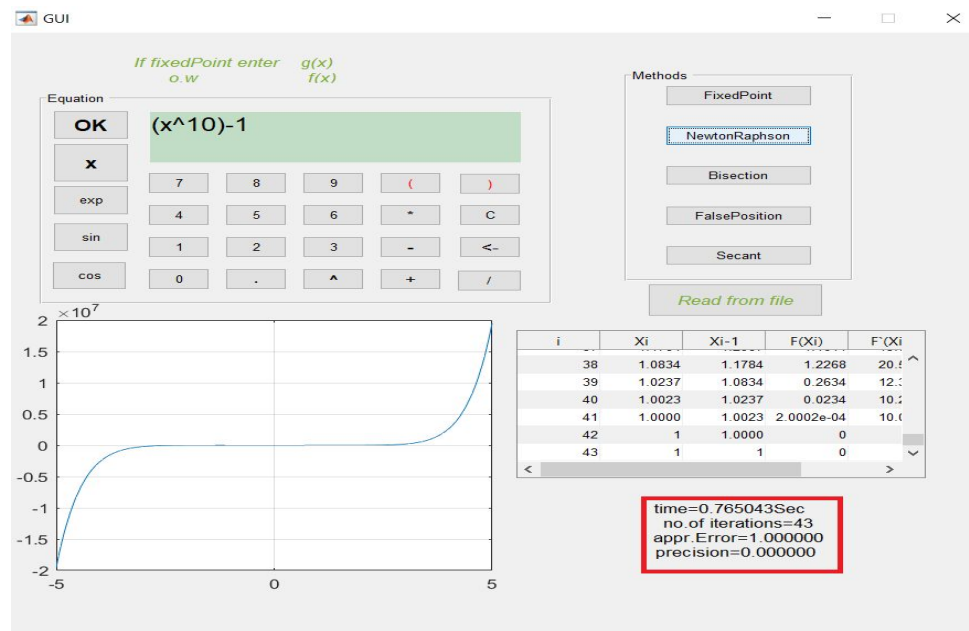


## III. example three

- If the equation is :

$$f(x) = x^{10} - 1$$

- Assume the initial guess of the root  **$x=0.5$**
- This is the Pitfalls of the Newton-Raphson Method.
- Newton-Raphson Method Sometimes be slow .
- The number of iterations is **43** iterations .



## conclusion

- It converges faster than Bracketing Methods, such as the bisection and false position methods.
- It converges faster than some method of type open method , such as the fixed point methods.
- It may diverge.
- Be active slowly ,sometimes.

## Problematic functions

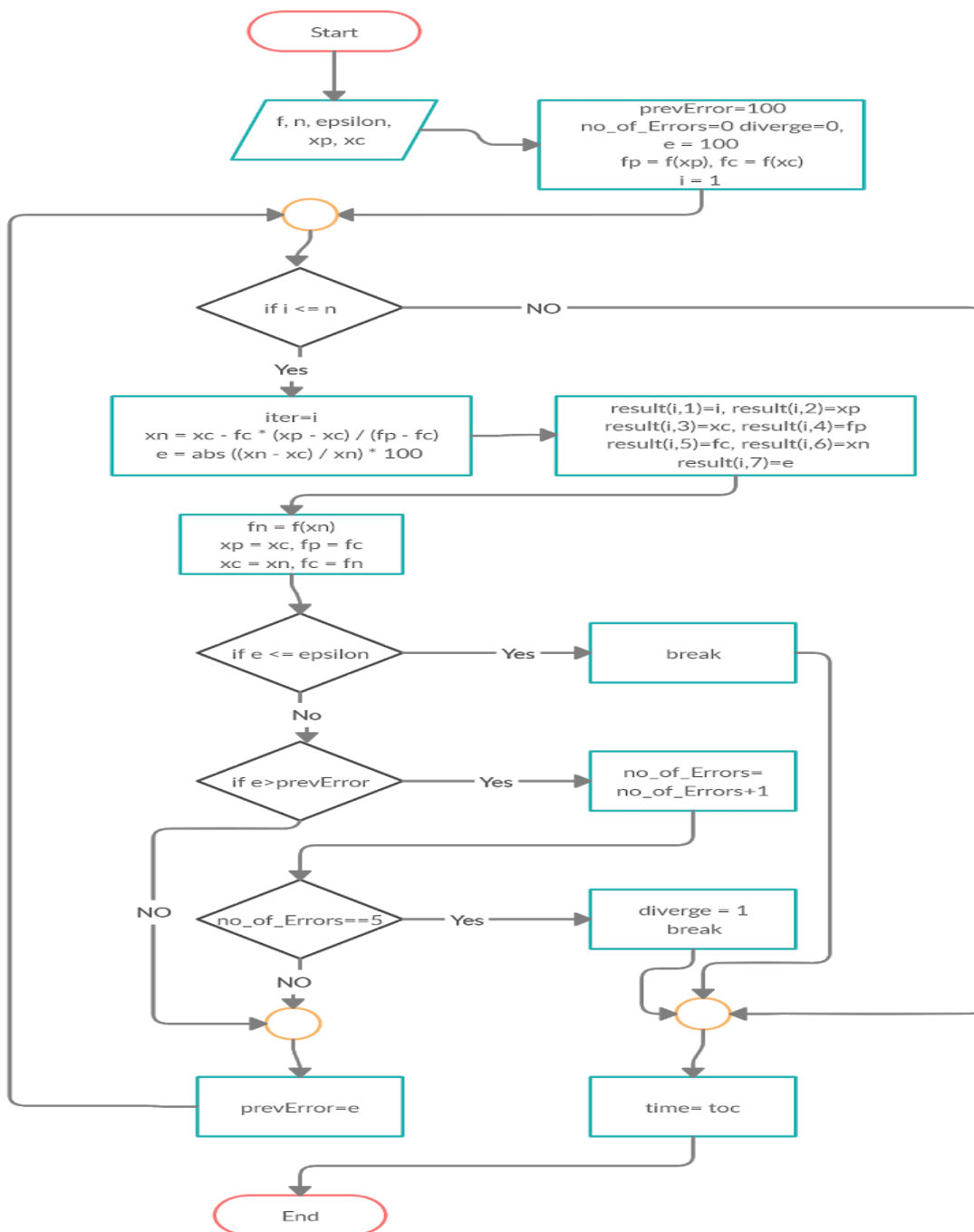
- One of the problematic functions :

When the initial guess is close to the root, Newton-Rahpson method usually converges :

- To improve the chance of convergence, we could use a bracketing method to locate the initial value for the Newton-Raphson method.
- A zero slope causes division by zero
- Need to find an initial guess for XR .
- Sometimes slow when Pitfalls of the function happen.
  - Use another method.

## 5.Secant

### Flowchart



## Data structure

### I. Matrices (2D arrays)

Used to create the table which contains all the data to be shown to the user in GUI.

This table contains every iteration with each calculation.

## Analysis for the behavior

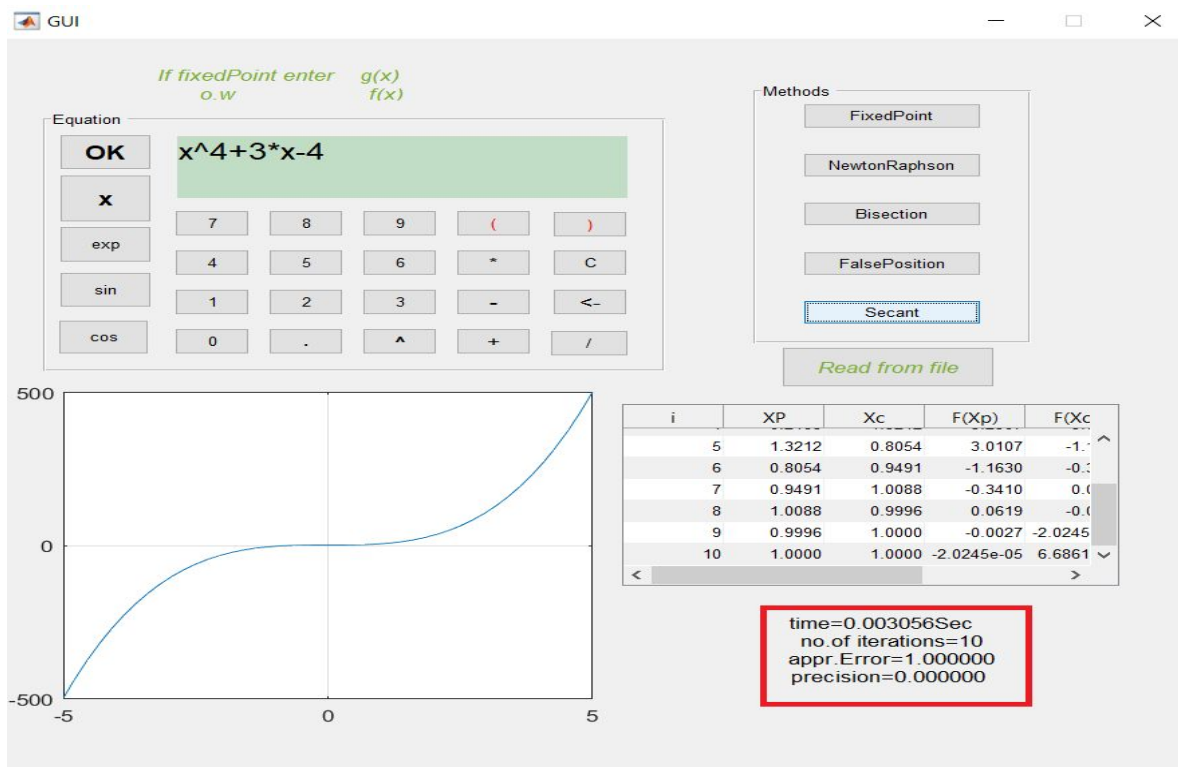
### I. example one

In case of this equation:  $X^4+3X-4$

$x_p=0$  ,  $x_c=3$  , Maximum number of iterations=100 , precision: 0.00001.

From the previous methods mentioned above:

False position method takes **58** iterations to reach the root, the bisection method takes **19** iterations, however Secant here takes only **10** iterations!



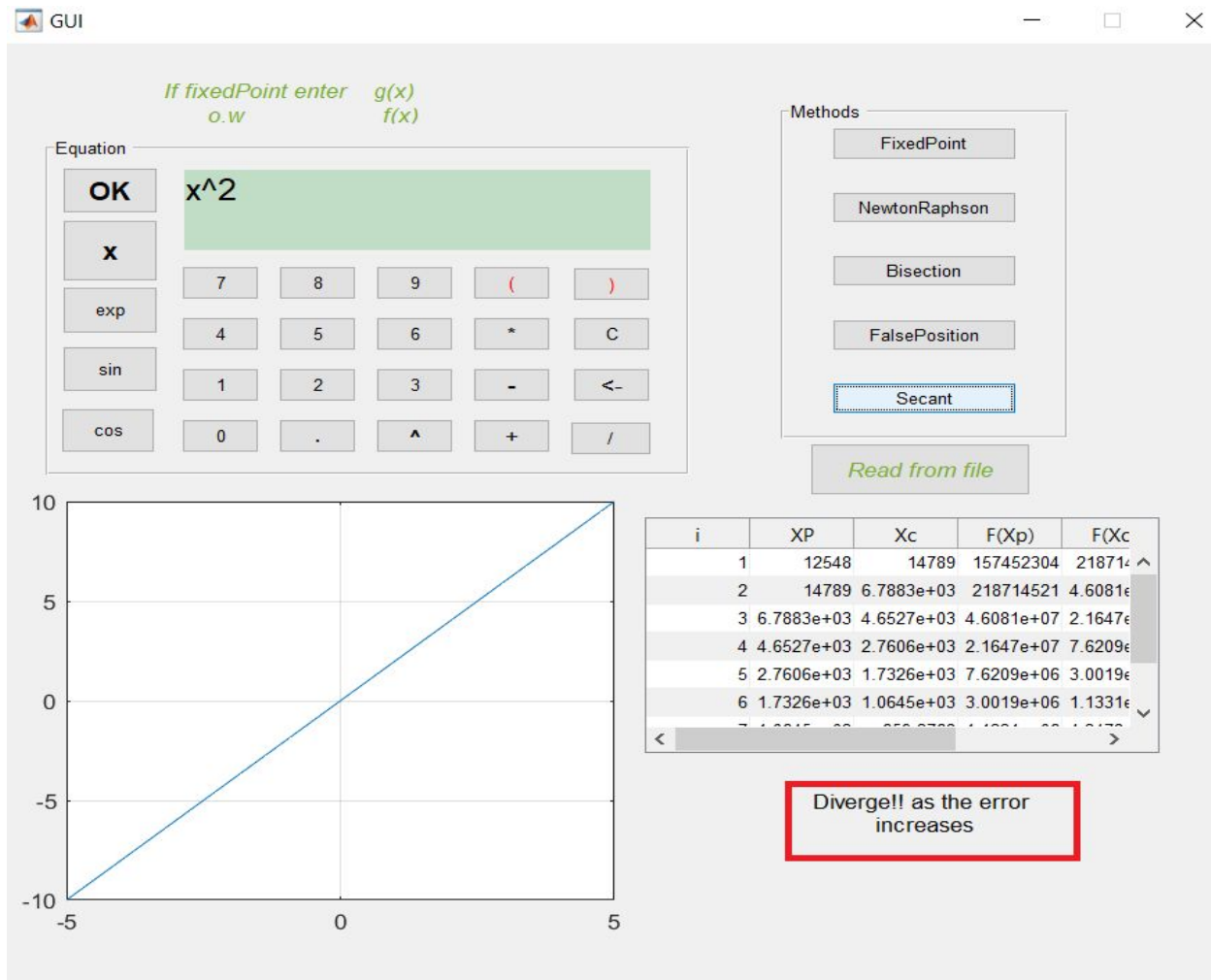


## II. example two

In case of this equation:  $x^2$

$x_p=12548$  ,  $x_c=14789$  , Maximum number of iterations=100 , precision: 0.00001 .

We know that the root of the previous equation is zero, however because of bad initial points the method diverges!

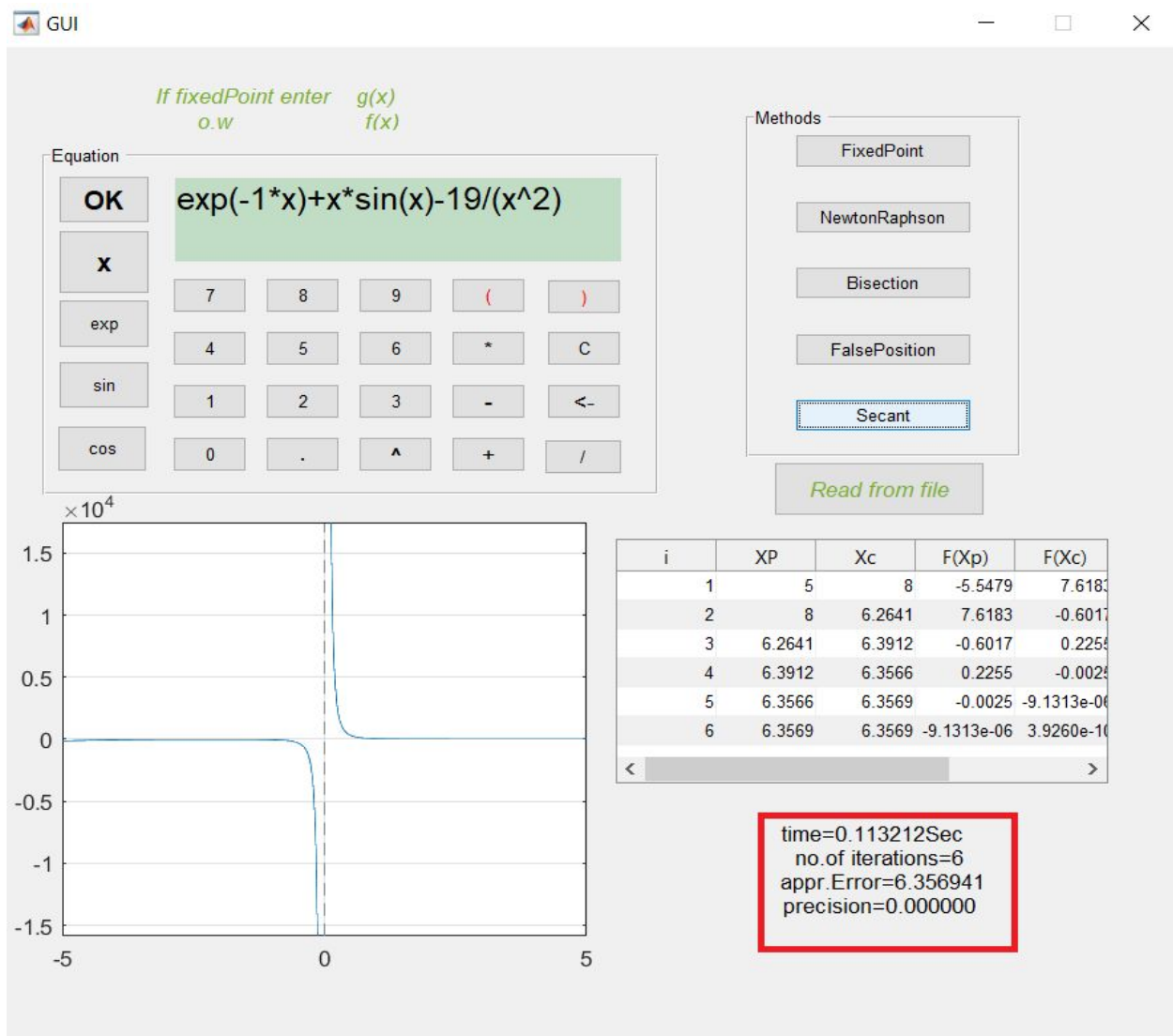


### III. example three

In case of this equation:  $e^{-x} + x \sin(x) - (x^2)^{19}$

$x_p=5$  ,  $x_c=8$  , Maximum number of iterations=100 , precision: 0.00001 .

Although the equation is quite complex the method managed to converge after only **6** iterations taking only **0.11321** sec as there are not many calculations inside this method.



## conclusion

- It converges faster than a linear rate, so that it is more rapidly convergent than the bisection and false position methods.
- It may diverge
- It's simpler than some methods as it does not require use of the derivative of the function and requires only one function evaluation per iteration compared with Newton's method which requires two.

## Problematic functions

- Functions that have a tangent at some point that is parallel to the x-axis it may have difficulty in its calculations.
- Any function with bad initial guesses can converge very slowly or badly it may diverge!

### Suggestions:

1. try to guess good initial values if the interval containing its root is quite easy to determine.
2. pick up two points from different intervals and compare the function behavior for each one.

# General algorithm

**Bisection** method can be used as general algorithm

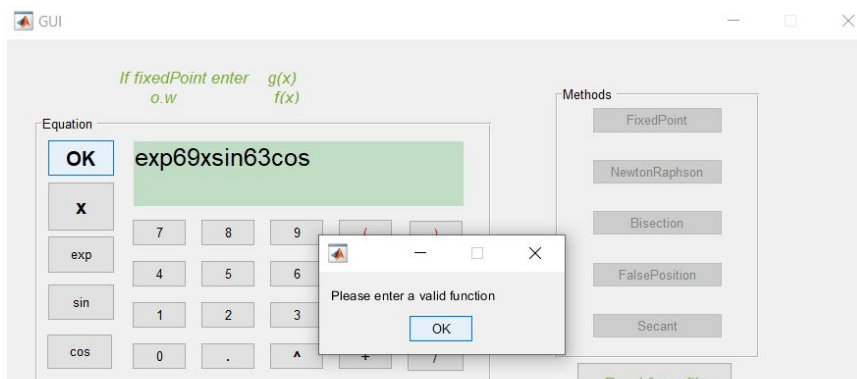
## I. Reason for this decisions

Bisection method is a bracketing method ,so it always converges.

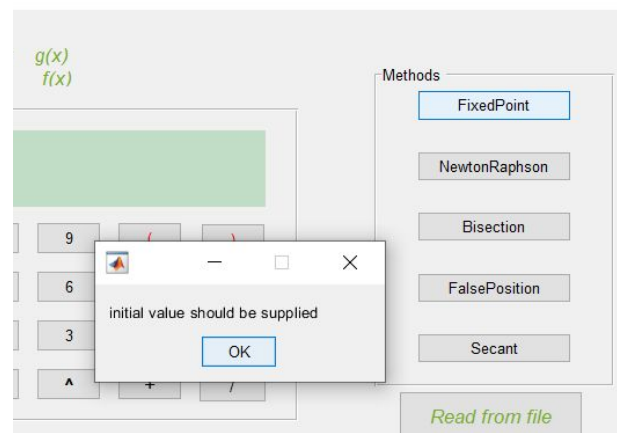
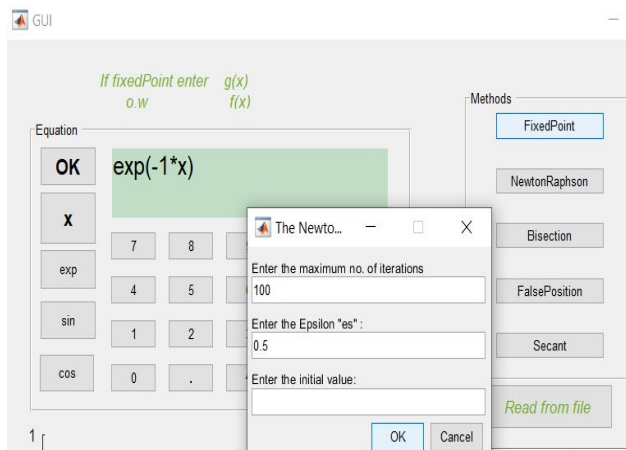
In most cases , bisection method is faster than false position method.

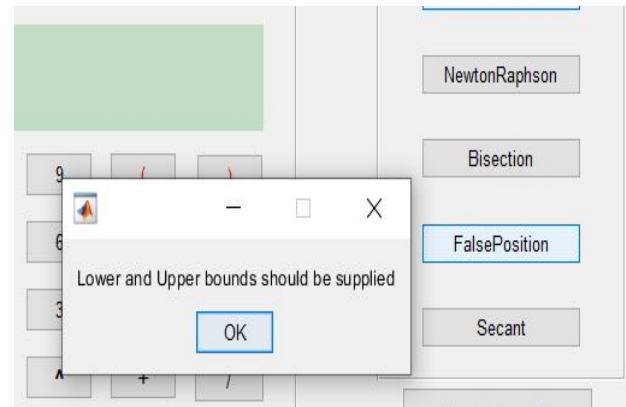
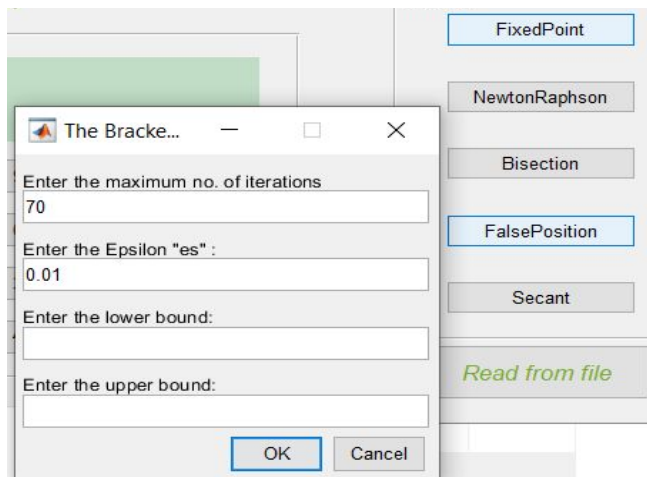
# Sample runs

## I. The user should enter a valid function if he doesn't, the buttons won't open.

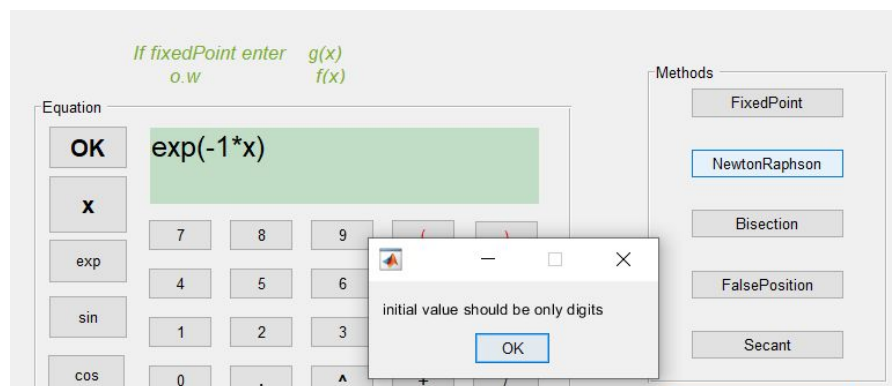
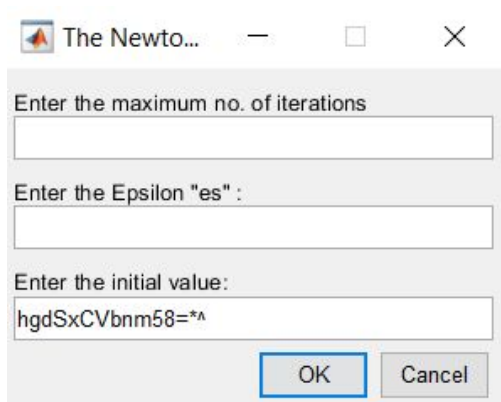


## II. The user should enter the values of x required in the window.





### III. Inputs should be digits.




# User Guide to read from file

- Input arguments can be read from a text file using the button (Read from file)



- Each argument should be in a separate line in the file.
- Users should consider the order of arguments in each method.
  1. In bisection method the order of arguments should be:
    - I. Equation.
    - II. Lower bound.
    - III. Maximum number of iterations.
    - IV. Absolute tolerance.
    - V. Upper bound.
  2. In False position method the order of arguments should be:
    - I. Equation.
    - II. Lower bound.
    - III. Maximum number of iterations.
    - IV. Absolute tolerance.
    - V. Upper bound.
  3. In secant method the order of arguments should be:
    - I. Equation.

- 
- II.  $X_p$ .
  - III. Maximum number of iterations.
  - IV. Absolute tolerance.
  - V.  $X_c$ .

4. In Newton Raphason method the order of arguments should be:

- I. Equation.
- II. Initial value.
- III. Maximum number of iterations.
- IV. Absolute tolerance.

5. In Fixed point method the order of arguments should be:

- I. Equation.
- II. Initial value.
- III. Maximum number of iterations.
- IV. Absolute tolerance.

- After choosing your file , you can select a method to solve .