Team Members	Academic ID
Omar Muhammed El said Metmowh	19016082
Ahmed Rabea Salam Ali	19015229
Marwan Mahmoud Ibrahim Muhammed	19016621
Ussif Ashraf Ussif	19016910
Ibrahim Tarek Ibrahim Abdelaal	19015167

Numerical computing project

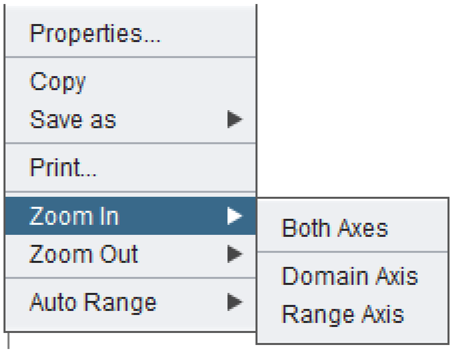
Calculating the root of equation with the methods (Bisection, False-Position, Fixed point, Newton-Raphson, Secant Method) with JAVA

Data structures

User defined data structure Point (x, y) which accepts x and y as floats or integers.
ArrayList: to store given equations
EQN: which is arraylist of equation members.
equation members: in which each coefficient is attached to its variable
for example: 2x+3y = 5
EQN --> [(2, x), (3, y), (5, null)]
equation members --> (2, x), (3, y), (5, null)

User manual

Zooming → user can zoom in by selecting the desired area using the mouse left click and zoom out by the same technique but in reverse direction, or simply right click on the desired place and select zoom in or zoom out (both axes, x axis, y axis) from the menu



Saving→ user should right click on the desired chart to be saved and select save as.

Coloring→ **Blue color indicates Xu**
Red color indicates Xl
Yellow color indicates final answer(root)

Syntax→ e^x is written as exp(x)

Libraries

We used **JFreeChart** library to draw the graphs.

Design decisions

Fixed point → there is no way to always guarantee convergence, we add x to both sides of the equation and it converges in most of the cases but diverges with bad initials

• PSEUDO CODE FOR EACH METHOD

False position

```
if substitute in function with (xl, A, y, x, p) * substitute in function with
(xu, A, y, x, p) > 0
   return wrong assumption
 else if substitute in function with (xl, A, y, x, p) = 0
   return xl
 else if substitute in function with(xu, A, y, x, p) = 0
   return xu
while ea >= epsilon and i < iterations
       add point (xl, 0) to points
       add point (xu, 0) to points
       fxl = substitute in function with (xl, A, Y, x, P)
       fxu = substitute in function with (xu, A, y, x, p)
       xr = (xl * fxu - xu * fxl) / (fxu - fxl)
       fxr = substitute in function with (xr, A, y, x, p)
       if fxr == 0
           break
       else if fxr < 0
           xu = xr
       else
           xl = xr
       ea = absolute (((xr - xrold) / xr)) * 100
       xrold = xr
       i++
return root value
```

Bisection

```
if substitute in function with (xl, A, y, x, p) * substitute in function with
(xu, A, y, x, p) > 0
   return wrong assumption
 else if substitute in function with (xl, A, y, x, p) = 0
   return xl
 else if substitute in function with (xu, A, y, x, p) = 0
   return xu
while ea >= epsilon and i < iterations)
   add point (xl, 0) to points
   add point (xu, 0) to points
   xr = (xu + xl) / 2
   if substitute in function with (xr, A, y, x, p) = 0.0
       break
   else if substitute in function with (xr, A, y, x, p) * substitute in
function with (xl, A, y, x, p) < 0)
       xu = xr
   else
       xl = xr
   ea = absolute (((xr - xrold) / xr)) * 100;
   xrold = xr
   i++
return root value
```

FixedPoint

```
for i = 0 to A.length
   for j = 0 to A[0].length
       Ag[i][j] = A[i][j];

while ea >= epsilon and i < iterations
   xold = xc
   xc = substitute in function with (xc, Ag, y, xg, p)
   fxc = substitute in function with (xc, A, y, x, p)
   if fxc = 0
       break
   ea = Math.abs((xc[0] - xold) / xc[0]) * 100
   I++
return root value
```

Newton Raphson

```
if substitute in function with (xold, A, y, x, p) = 0
   return initial guess
while ea >= epsilon and i < iterations)
   fxold = substitute in function with (xold, A, y, x, p)
   fdxold = substitute in function with (xold, A, y, x, p)
   xnew = xold - (fxold / fdxold)

   if fxold = 0
       break

   ea = absolute ((xnew - xold) / xnew) * 100
   xold = xnew
   i++
return root value
```

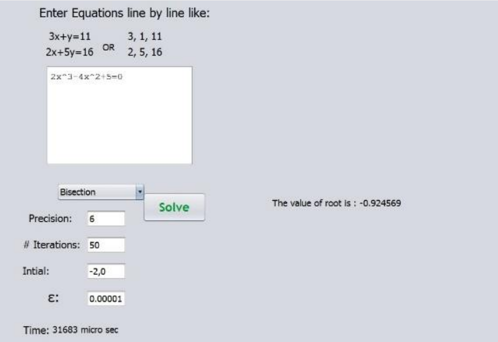
Secant

```
while ea >= epsilon and i < iterations
   fxp = substitute in function with(xp, A, y, x, p)
   fxc = substitute in function with(xc, A, y, x, p)
   xn = xc - (fxc * (xp - xc)) / (fxp - fxc)
   fxn = substitute in function with(xn, A, y, x, p)

   if fxn = 0
       break

   ea = absolute ((xn - xc) / xn) * 100
   xp = xc
   xc = xn
   i++
return root value
```

- # Test cases


Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 OR 2, 5, 16

2x^3-4x^2+5=0

Bisection

Precision: 6

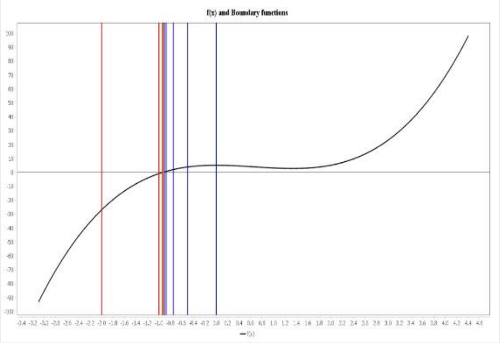
Iterations: 50

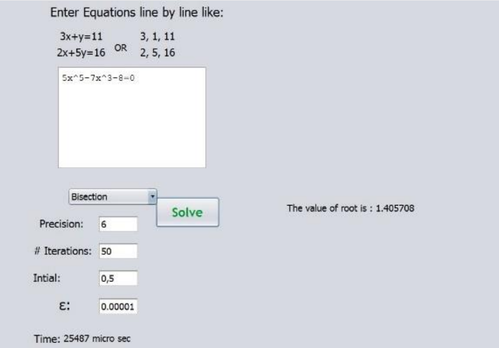
Intial: -2,0

ε: 0.00001

The value of root is : -0.924569

Time: 31683 micro sec




Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 OR 2, 5, 16

5x^5-7x^3-8=0

Bisection

Precision: 6

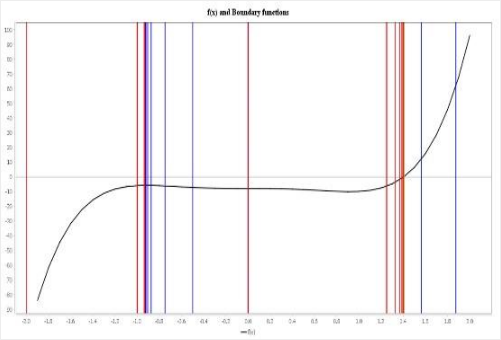
Iterations: 50

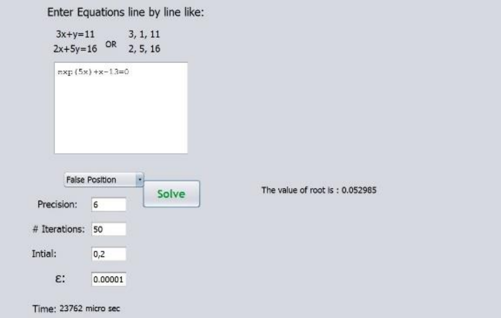
Intial: 0,5

ε: 0.00001

The value of root is : 1.405708

Time: 25487 micro sec




Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 OR 2, 5, 16

exp(5x)+x-13=0

False Position

Precision: 6

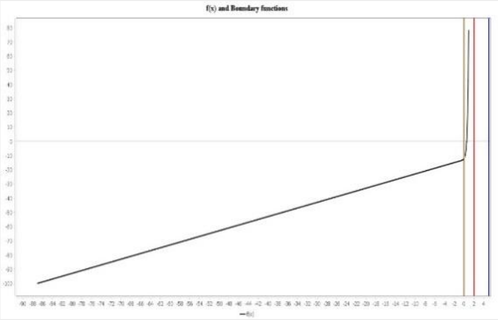
Iterations: 50

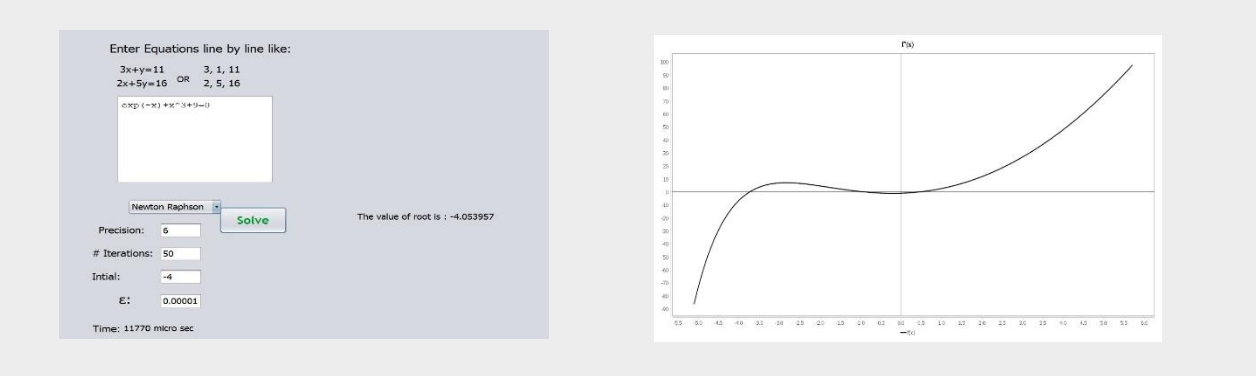
Intial: 0,2

ε: 0.00001

The value of root is : 0.052985

Time: 23762 micro sec





Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 2, 5, 16

exp(-x)+x^3+9=0

Newton Raphson

Precision: 6

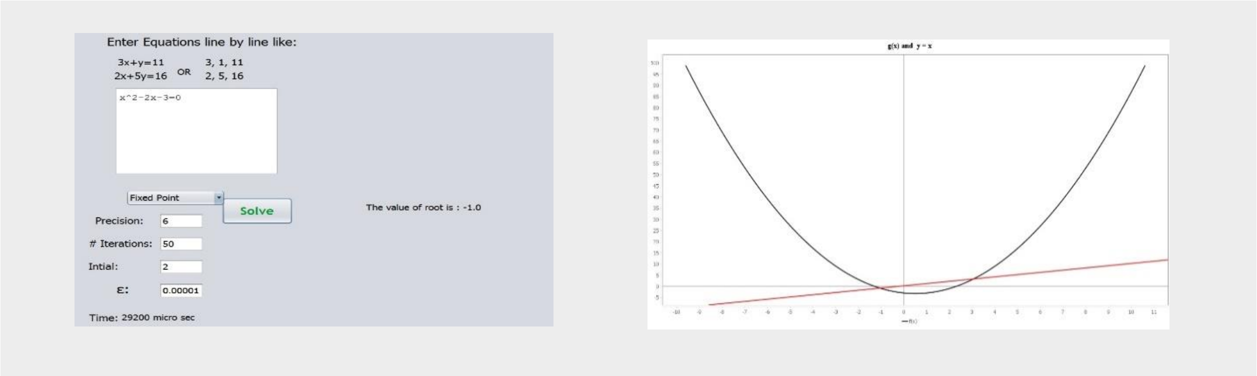
Iterations: 50

Intial: -4

ε: 0.00001

The value of root is : -4.053957

Time: 11770 micro sec



Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 2, 5, 16

x^2-2x-3=0

Fixed Point

Precision: 6

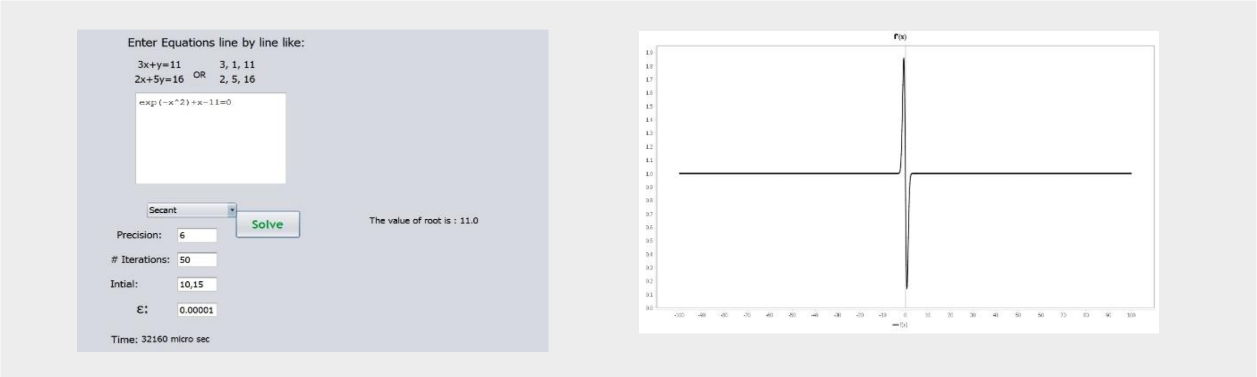
Iterations: 50

Intial: 2

ε: 0.00001

The value of root is : -1.0

Time: 29200 micro sec



Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 2, 5, 16

exp(-x^2)+x-11=0

Secant

Precision: 6

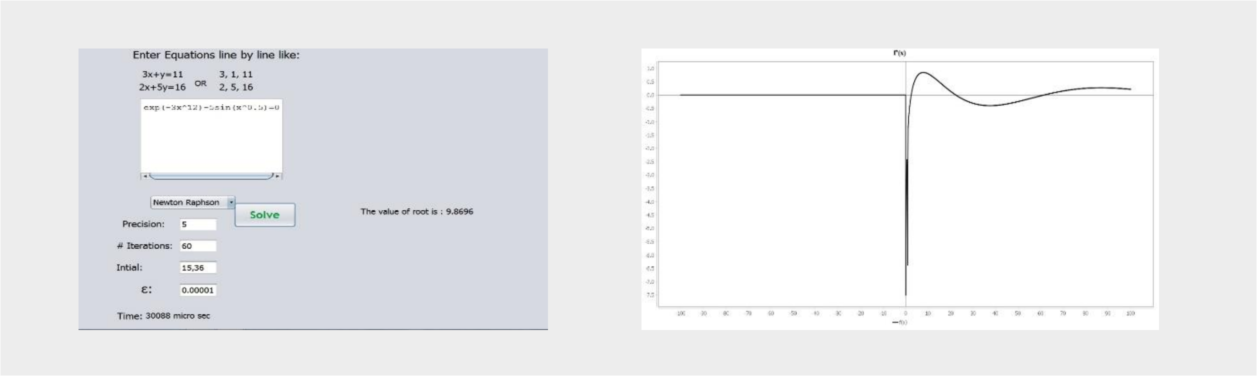
Iterations: 50

Intial: 10,15

ε: 0.00001

The value of root is : 11.0

Time: 32160 micro sec



Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 2, 5, 16

exp(-3x^12)-5sin(x^0.5)=0

Newton Raphson

Precision: 5

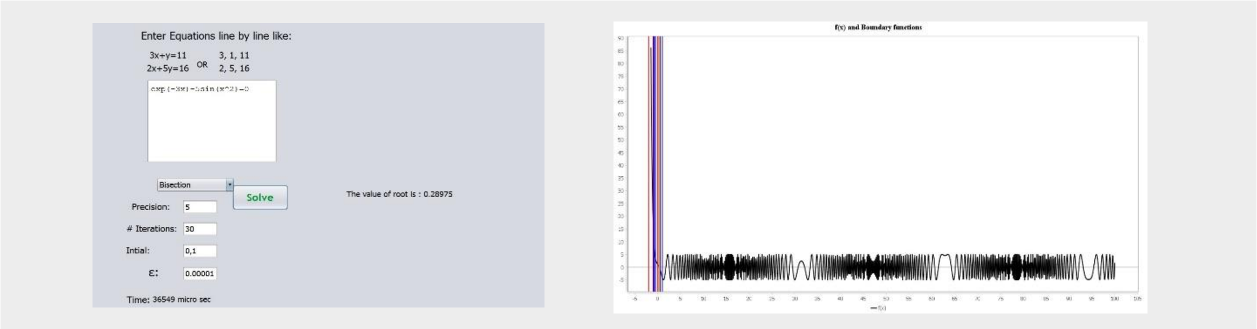
Iterations: 60

Intial: 15,36

ε: 0.00001

The value of root is : 9.8696

Time: 30088 micro sec



Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 OR 2, 5, 16

exp(-3x)-5sin(x^2)=0

Bisection

Precision: 5

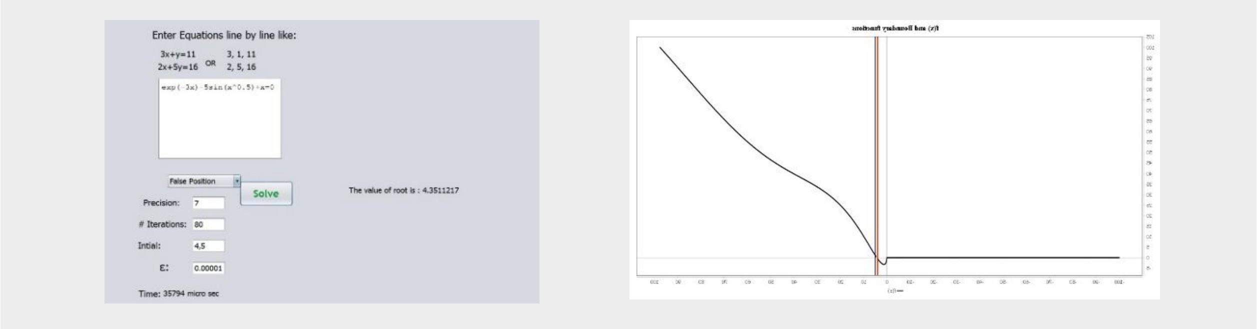
Iterations: 30

Intial: 0,1

ε: 0.00001

The value of root is : 0.28975

Time: 36549 micro sec



Enter Equations line by line like:

3x+y=11 OR 3, 1, 11
2x+5y=16 OR 2, 5, 16

exp(-3x)-5sin(x^0.5)+x=0

False Position

Precision: 7

Iterations: 80

Intial: 4,5

ε: 0.00001

The value of root is : 4.3511217

Time: 35794 micro sec

• Comparison between different methods

we will compare between methods with **2** equations to get the roots

first equation: “exp(-3x)-5*sin(x^0.5) +x=0”
second equation: “2*x^3-4*x^2+5=0”

- HINT: we try a method 10 times and take the average off the time in microseconds to be more accurate because the time effects with the memory state and the processor.

**number of iterations:50 and precision: 6.

Methods	*Average Time(μs) first equations*	*Average Time(μs) second equations*
False-Position Method	32897	22248
Bisection Method	34583	26689
Fixed point Method	31787	21334
Newton-Raphson Method	26590	20236
Secant Method	22414	23365