Problem Name:

Finding real root of non-linear equation by using Newton Raphson Method.

Apparatus:

- Computing Device (e.g.; Computer)
- Programming Environment (MATLAB R2016b)

Method:

Open Method are a type of graphical method where a formula is used to project the current approximate to the real root in an iterative fashion. These methods might converge or diverge depending on the initial guess. Newton-Raphson method fall under the category of Open methods.

The Newton-Raphson method is an iterative numerical technique used to find successively better approximations to the roots of a real-valued function. Named after Sir Isaac Newton and Joseph Raphson. The method is particularly effective for finding the roots of equations and functions.

Algorithm:

1. Start

2. Define function as f(x)

3. Define first derivative of f(x) as g(x)

4. Input initial guess (x0), tolerable error (e)

   and maximum iteration (N)

5. Initialize iteration counter i = 1

6. If g(x0) = 0 then print "Mathematical Error"
   and goto (12) otherwise goto (7)

7. Calcualte x1 = x0 - f(x0) / g(x0)

8. Increment iteration counter i = i + 1

9. If i >= N then print "Not Convergent"
   and goto (12) otherwise goto (10)

10. If |f(x1)| > e then set x0 = x1
    and goto (6) otherwise goto (11)

11. Print root as x1

12. Stop

Code:

```
% Clearing Screen
Clc

% Setting x as symbolic variable
syms x;

% Input Section
y = input('Enter non-linear equations: ');
a = input('Enter initial guess: ');
e = input('Tolerable error: ');
N = input('Enter maximum number of steps: ');
% Initializing step counter
step = 1;

% Finding derivate of given function
g = diff(y,x);

% Finding Functional Value
fa = eval(subs(y,x,a));

while abs(fa)> e
   fa = eval(subs(y,x,a));
   ga = eval(subs(g,x,a));
```

```
    if ga == 0
        disp('Division by zero.');
        break;
    end
    b = a - fa/ga;
    fprintf('step=%d\ta=%f\tf(a)=%f\n',step,a,fa);
    a = b;

    if step>N
       disp('Not convergent');
       break;
    end
    step = step + 1;
end

fprintf('Root is %f\n', a);
```

Input:

Enter non-linear equations: x^3-x-1

Enter initial guess: -2

Tolerable Error: 0.0001

Enter maximum number of steps: 5

Output:

NewtonRaphson_1039.m

/MATLAB Drive/MATLAB Reports/NewtonRaphson_1039.m

```
% Clearing Screen
clc

% Setting x as symbolic variable
syms x;

% Input Section
y = input('Enter non-linear equations: ');
a = input('Enter initial guess: ');
e = input('Tolerable error: ');
N = input('Enter maximum number of steps: ');
% Initializing step counter
step = 1;

% Finding derivate of given function
g = diff(y,x);

% Finding Functional Value
fa = eval(subs(y,x,a));

while abs(fa)> e
    fa = eval(subs(y,x,a));
    ga = eval(subs(g,x,a));
    if ga == 0
        disp('Division by zero.');
        break;
    end

    b = a - fa/ga;
    fprintf('step=%d\ta=%f\tf(a)=%f\n',step,a,fa);
    a = b;

    if step>N
       disp('Not convergent');
       break;
    end
    step = step + 1;
end

fprintf('Root is %f\n', a);
```

Command Window

```
Enter non-linear equations:
x^3-x-1
Enter initial guess:
-2
Tolerable error:
0.0001
Enter maximum number of steps:
5
step=1  a=-2.000000     f(a)=-7.000000
step=2  a=-1.363636     f(a)=-2.172051
step=3  a=-0.889235     f(a)=-0.813918
step=4  a=-0.296095     f(a)=-0.729864
step=5  a=-1.286435     f(a)=-1.842507
step=6  a=-0.821713     f(a)=-0.733118
Not convergent
Root is -0.106920
>>
```

Discussion:

In this report, we implemented the Newton-Raphson method as a powerful numerical tool for finding real roots in nonlinear equations. Leveraging this method allowed us to iteratively refine our initial guesses, bringing us closer to accurate solutions. While implementing the Newton-Raphson method, we experienced the method's rapid convergence under suitable conditions. The iterative refinement of initial approximations showcased the method's efficiency, especially when our initial guesses were in proximity to the actual roots. However, we also noted the sensitivity of the method to the choice of initial guesses and the behavior of the functions involved.

While navigating the implementation process, a notable challenge surfaced with the utilization of the MATLAB online version. Faced with the intricacies of bringing the output into a comprehensible format, we encountered a unique set of challenges that added a layer of complexity to an otherwise handy tool.

Despite these challenges, the overall experience highlighted the practicality of using MATLAB. In navigating through the intricacies, we not only overcame hurdles in output management but also gained valuable insights into the implementation nuances of the Newton-Raphson method.