**Introduction:**

**Iteration Method:** This is a root finding method which requires one starting value and the initial values need not necessarily bracket the root like bisection or false position method. This method is called fixed point Iteration. In [numerical analysis](https://en.wikipedia.org/wiki/Numerical_analysis), **fixed-point iteration** is a method of computing [fixed points](https://en.wikipedia.org/wiki/Fixed_point_%28mathematics%29) of [iterated functions](https://en.wikipedia.org/wiki/Iterated_function). It is a open method.

**The Method:**

To describe this method for finding the roots of the equation

*f (x)=*0

we rewrite this equation in the form

x=P(x);

There are many ways of doing this .For example ,the equation

X3+x2-1=0

can be expressed in different forms

1. x=(1+x)-1/2
2. x=(1-x3)1/2
3. x=(1-x2)1/3

Let i) is the required function means P(x)= (1+x)-1/2……(1)

Let x0 be the approximate root of equation (1). Then substituting in (1) ,we get the first approximate as

X1=P(x0)

Successive substitution X2=P(x1), X3=P(x2)…… Xn=P(xn-1).

When | Xn+1- Xn|<10-4 the root will be found.

It is also called Fixed point Iteration method.

**Newton-Raphson Method:**

It is a method for finding successive better approximations to the roots of a real valued function. The another name of this method is Method Of Tangent.

**Method:**

Let x0 be an approximate root of f(x)= 0 and let x1=x0+h be the correct root so that f(x0)=0. Expanding f(x0+h) by Taylor’s series, we obtain

f(x0)+hf’(x0)+h2/2!f’’(x0)+………..=0.

Neglecting the 2nd and 3rd order derivative,

f(x0)+hf’(x0)=0.

* h= - f(x0)/f’(x0)

so, x1=x0- f(x0)/f’(x0)

xn+1= xn – f(xn)/f’(xn)

when, (xn+1 – xn) < 10-4 the root will be found.

**The Source Code:**

#include<bits/stdc++.h>

using namespace std;

double f(double x)

{

double y=pow(1+x,-0.5);

return y;

}

double func(double x)

{

double y=pow(x,3)+pow(x,2)-1;

return y;

}

double deriv(double x)

{

double y=3\*pow(x,2)+2\*x;

return y;

}

void newtn(double x)

{

int i=0;

double h=func(x)/deriv(x);

while(fabs(h)>0.0001)

{

i++;

h=func(x)/deriv(x);

cout<<"\nn="<<i<<" "<<"Xn="<<x<<" ";

x=x-h;

cout<<"Xn+1 ="<<x<<endl;

}

cout<<"\nRoot is ="<<x<<endl;

}

void Iteration(double x)

{

double x1;

int i=0;

x1=f(x);

while(fabs(x-x1)>=0.0001)

{

i++;

cout<<"\nn="<<i<<" "<<"Xn="<<x<<" ";

x=x1;

x1=f(x);

cout<<"Xn+1="<<x1<<endl;

}

cout<<"\nRoot is = "<<x1<<endl;

}

int main()

{

double x;

int i=0,option;

cout<<"\*press '1'for Iteration method"<<"\n\*press '2' for newton Raphson method"<<"\n\*press '0' for exit"<<endl;

while(option)

{

cin>>option;

cout<<"\nEnter initial guesses\t"<<endl;

cin>>x;

switch(option)

{

case 1:

Iteration(x);

break;

case 2:

newtn(x);

break;

default :

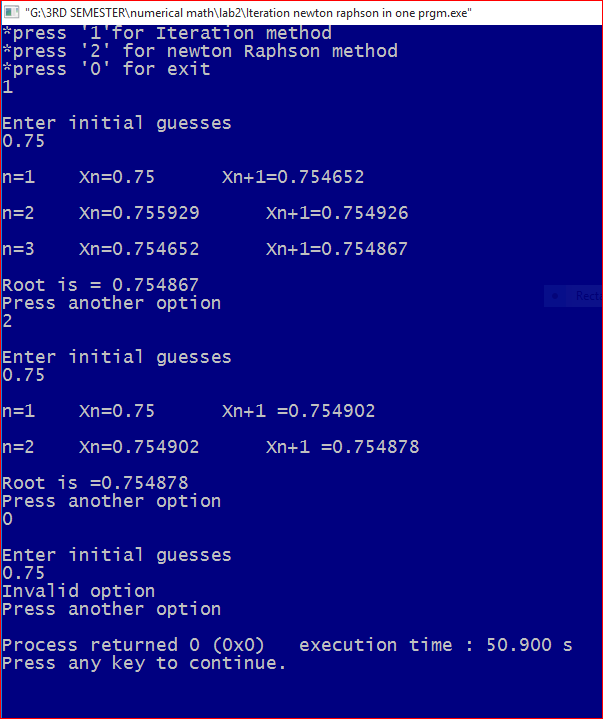
cout<<"Invalid option"<<endl;

}

cout<<"Press another option"<<endl;

}

}



**Discussion**: Newton-Raphson process has a second order or quadratic convergence. It can be used for solving both algebraic and transcendental equations and it can also be used when the roots are complex.Iteration method can be used to find the difference between two successive approximations to achieve a prescribed accuracy.