**Introduction:**

**Ramanujan Method:** It is a method of iterative procedure to determine the smallest root of the equation. SrinivasaRamanujan first described it.

Let the equation

a1x+a2x2+a1x3+……………..=1

or,1-(a1x+a2x2+a1x3+……………..)=0

to find the smallest root of *f(x)=0*,we consider *f(x)*in the form

*f(x)*=1-(a1x+a2x2+a1x3+……………..)

and then write

[1-(a1x+a2x2+a3x3+……………..)]-1=b1+b2x+b3x3…………

=>1+(a1x+a2x2+a3x3+…..)+(a1x+a2x2+a3x3+…..)+……….=b1+b2x+b3x3…….. (i)

To find the value of bi, we equate coeffients of like power of x on both sides of eq. (i),we then obtain

b1=1

b2=a1=a1b1 since b1=1

b3=a2+a12=a2b1+a1b2 since b2=a1

…………..

…………..

bn=a1bn-1+a2bn+……….+an-1b1

The ratio bi-1/bi called the convergent,approachin the limit, smallest root of the f(x)=0.

When (bi-1/bi) -(bi/bi+1)<10-4 the root will be found..

**Source Code:**

#include<bits/stdc++.h>

using namespace std;

double A[100];

double B[100];

double DIV[100];

doubleFind\_B(int x)

{

inti;

double sum=0.0;

for(i=1;i<x;i++)

{

sum=sum+1.0\*(A[i]\*B[x-i]);

}

return sum;

}

voiddivisn()

{

for(inti=0;i<35;i++)

{

DIV[i]=B[i]/(1.0\*B[i+1]);

}

}

int main()

{

doublea,b,c,d;

cout<<"Enter co-effeciant"<<endl;

cout<<"\nco-effient of x3=";

cin>>c;;

cout<<"\nco-effient of x2=";

cin>>b;

cout<<"\nco-effient of x=";

cin>>a;

cout<<"\nco-efficint of constant=";

cin>>d;

A[1]=a/(-d);

A[2]=b/(-d);

A[3]=c/(-d);

B[1]=1;

for(inti=2;i<35;i++)

{

B[i]=Find\_B(i);

}

divisn();

inti=1;

double h=DIV[i]-DIV[i+1];

while(fabs(h)>0.0001)

{

cout<<endl;

i++;

divisn();

h=DIV[i]-DIV[i+1];

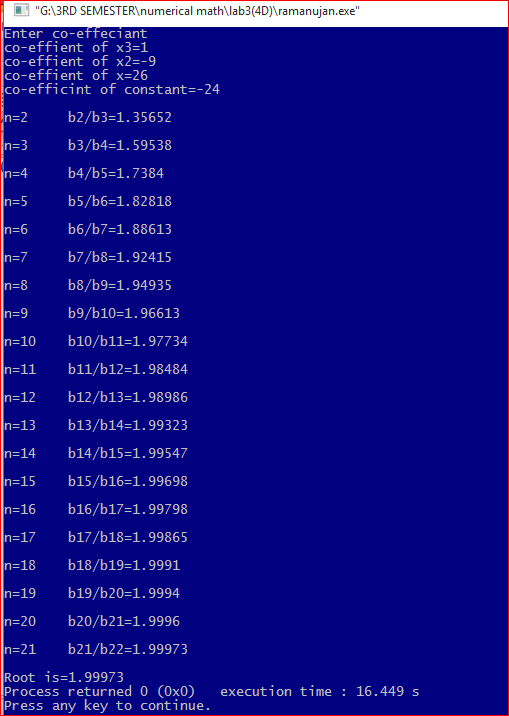
cout<<"n="<<i<<"\t"<<"b"<<i<<"/b"<<i+1<<"="<<DIV[i]<<endl;

}

cout<<"\nRoot is="<<DIV[i]<<endl;

}

**Input/Output:**



**Discussion:**

This method is used to find smallest root of the equation. In the above equation the root of the equation is 2,3 ,4 and its can be seen that the successive convergent approach the value 2.