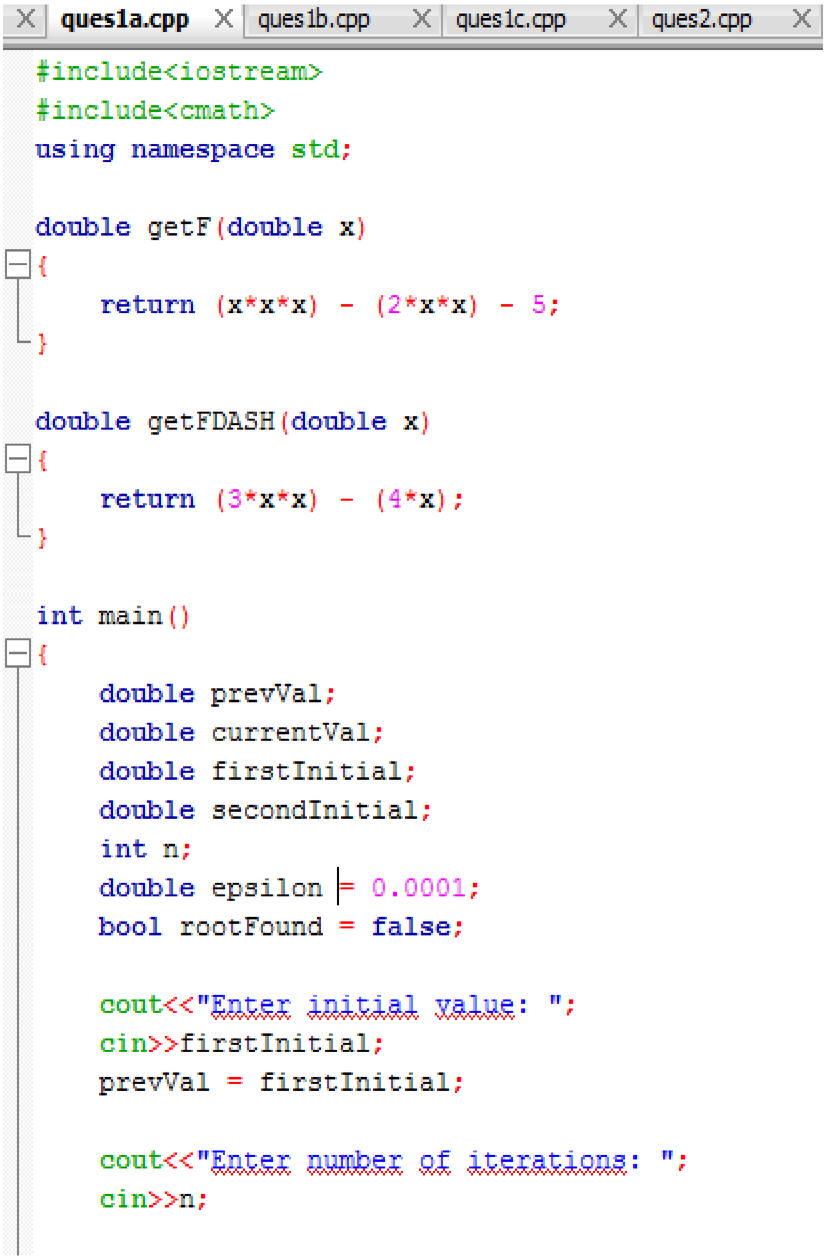
SCIENTIFIC COMPUTING

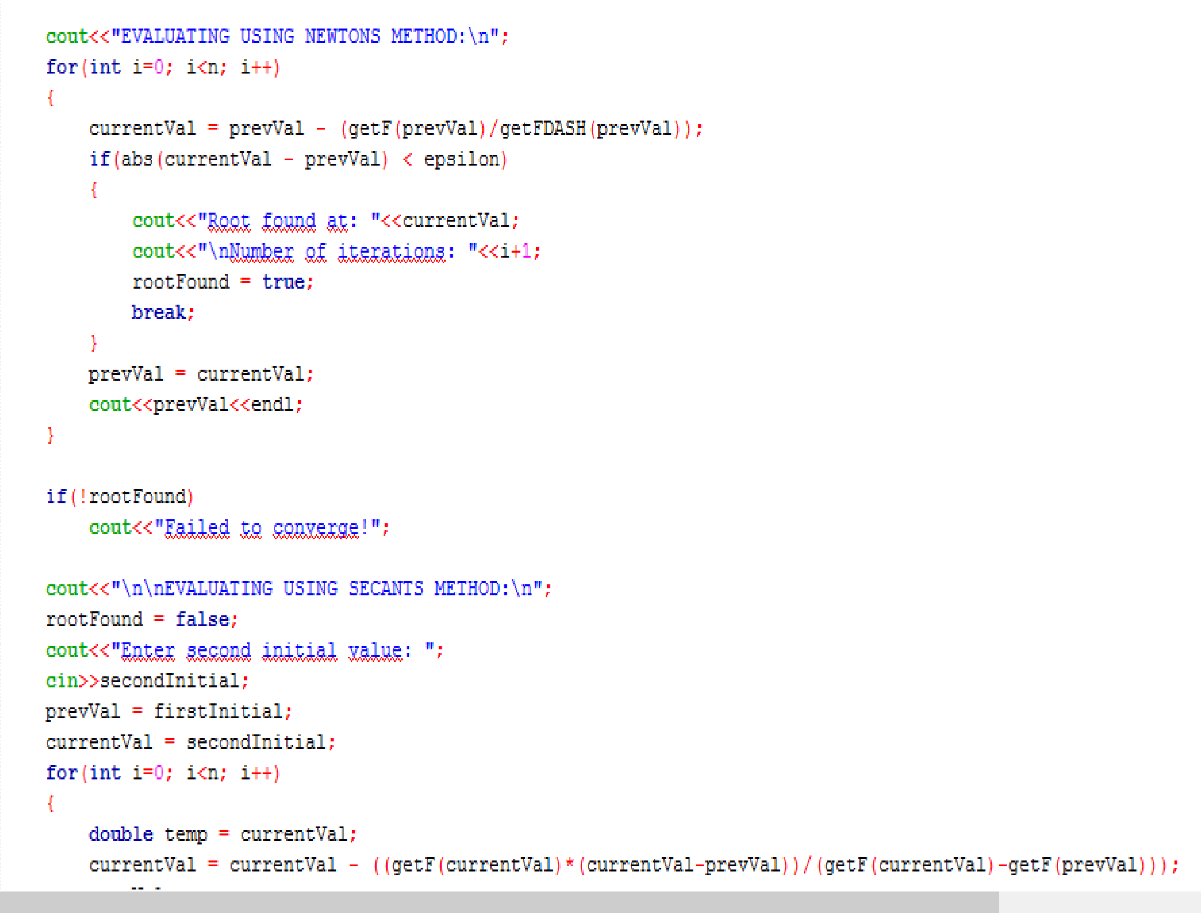
ASSIGNMENT 3

TUSHAR SIRCAR
130123038

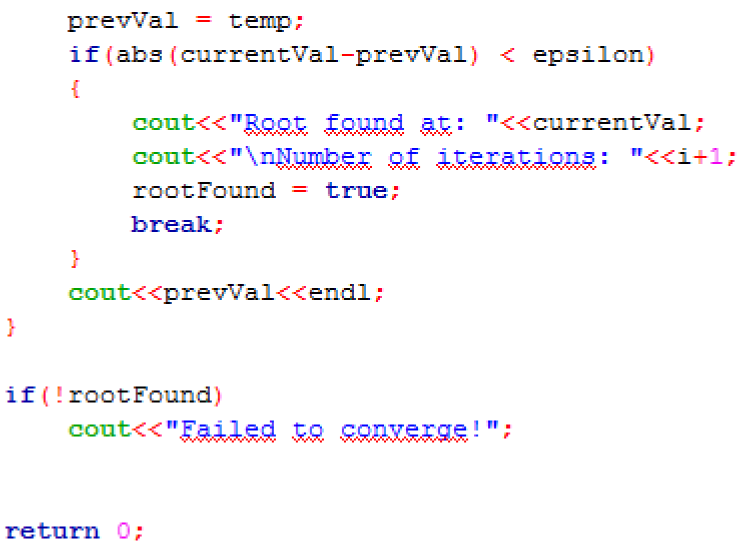
1. SOURCE CODE:



```
#include<iostream>
#include<cmath>
using namespace std;

double getF(double x)
{
    return (x*x*x) - (2*x*x) - 5;
}

double getFDASH(double x)
{
    return (3*x*x) - (4*x);
}

int main()
{
    double prevVal;
    double currentVal;
    double firstInitial;
    double secondInitial;
    int n;
    double epsilon = 0.0001;
    bool rootFound = false;

    cout<<"Enter initial value: ";
    cin>>firstInitial;
    prevVal = firstInitial;

    cout<<"Enter number of iterations: ";
    cin>>n;
```



```cpp
cout<<"EVALUATING USING NEWTONS METHOD:\n";
for(int i=0; i<n; i++)
{
    currentVal = prevVal - (getF(prevVal)/getFDASH(prevVal));
    if(abs(currentVal - prevVal) < epsilon)
    {
        cout<<"Root found at: "<<currentVal;
        cout<<"\nNumber of iterations: "<<i+1;
        rootFound = true;
        break;
    }
    prevVal = currentVal;
    cout<<prevVal<<endl;
}

if(!rootFound)
    cout<<"Failed to converge!";

cout<<"\n\nEVALUATING USING SECANTS METHOD:\n";
rootFound = false;
cout<<"Enter second initial value: ";
cin>>secondInitial;
prevVal = firstInitial;
currentVal = secondInitial;
for(int i=0; i<n; i++)
{
    double temp = currentVal;
    currentVal = currentVal - ((getF(currentVal)*(currentVal-prevVal))/(getF(currentVal)-getF(prevVal)));
```



```cpp
    prevVal = temp;
    if(abs(currentVal-prevVal) < epsilon)
    {
        cout<<"Root found at: "<<currentVal;
        cout<<"\nNumber of iterations: "<<i+1;
        rootFound = true;
        break;
    }
    cout<<prevVal<<endl;
}

if(!rootFound)
    cout<<"Failed to converge!";


return 0;
```

(a) NEWTONS METHOD

Initial Value: 2.5

Root Found: 2.69065

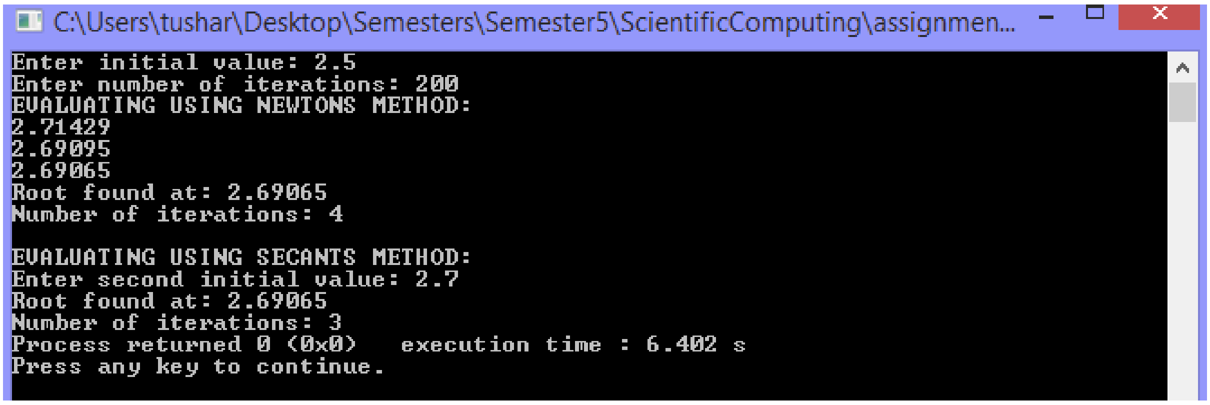
Number Of Iterations: 4

SECANT METHOD

Initial Values: 2.5, 2.7

Root Found: 2.69065

Number Of Iterations: 3


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 2.5
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
2.71429
2.69095
2.69065
Root found at: 2.69065
Number of iterations: 4

EVALUATING USING SECANTS METHOD:
Enter second initial value: 2.7
Root found at: 2.69065
Number of iterations: 3
Process returned 0 (0x0)   execution time : 6.402 s
Press any key to continue.
```

(b) NEWTONS METHOD

Initial Value: 0

Root Found: -0.470064

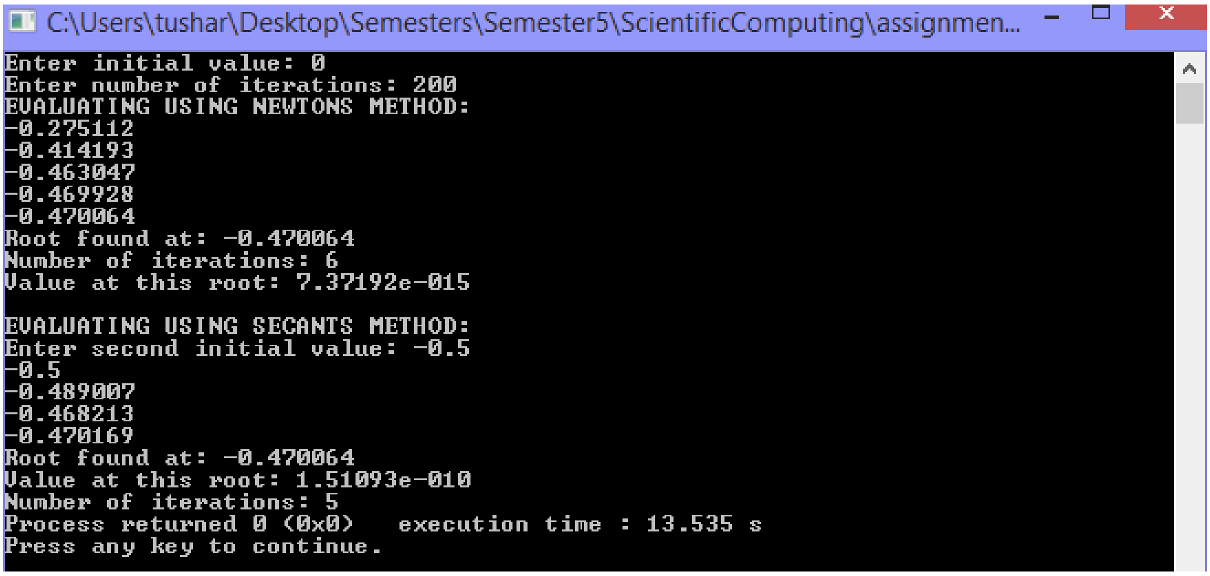
Number Of Iterations: 6

SECANT METHOD

Initial Values: 0, -0.5

Root Found: -0.470064

Number Of Iterations: 5


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 0
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
-0.275112
-0.414193
-0.463047
-0.469928
-0.470064
Root found at: -0.470064
Number of iterations: 6
Value at this root: 7.37192e-015

EVALUATING USING SECANTS METHOD:
Enter second initial value: -0.5
-0.5
-0.489007
-0.468213
-0.470169
Root found at: -0.470064
Value at this root: 1.51093e-010
Number of iterations: 5
Process returned 0 (0x0)   execution time : 13.535 s
Press any key to continue.
```

NEWTONS METHOD

Initial Value: -100

Root Found: -2.64561

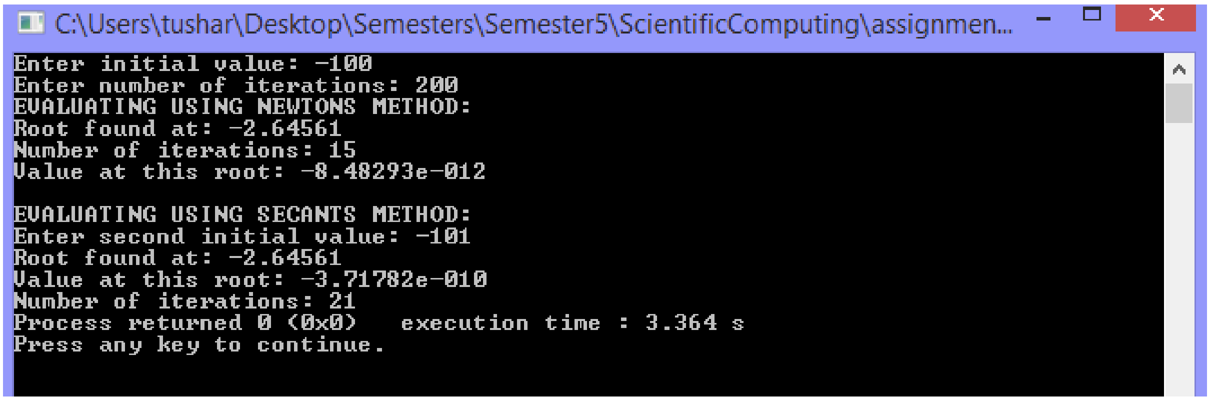
Number Of Iterations: 15

SECANT METHOD

Initial Values: -100, -101

Root Found: -2.64561

Number Of Iterations: 21


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: -100
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
Root found at: -2.64561
Number of iterations: 15
Value at this root: -8.48293e-012

EVALUATING USING SECANTS METHOD:
Enter second initial value: -101
Root found at: -2.64561
Value at this root: -3.71782e-010
Number of iterations: 21
Process returned 0 (0x0)   execution time : 3.364 s
Press any key to continue.
```

(c) NEWTONS METHOD

Initial Value: 1

Root Found: 1.49819

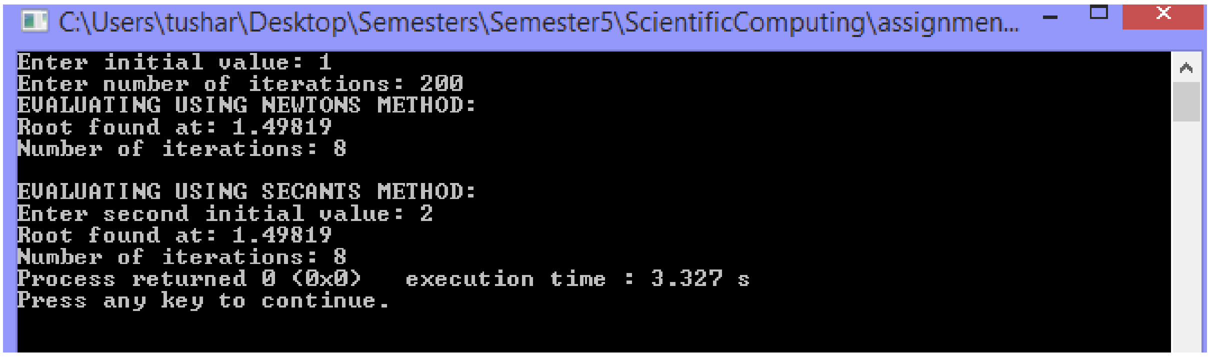
Number Of Iterations: 8

SECANT METHOD

Initial Values: 1, 2

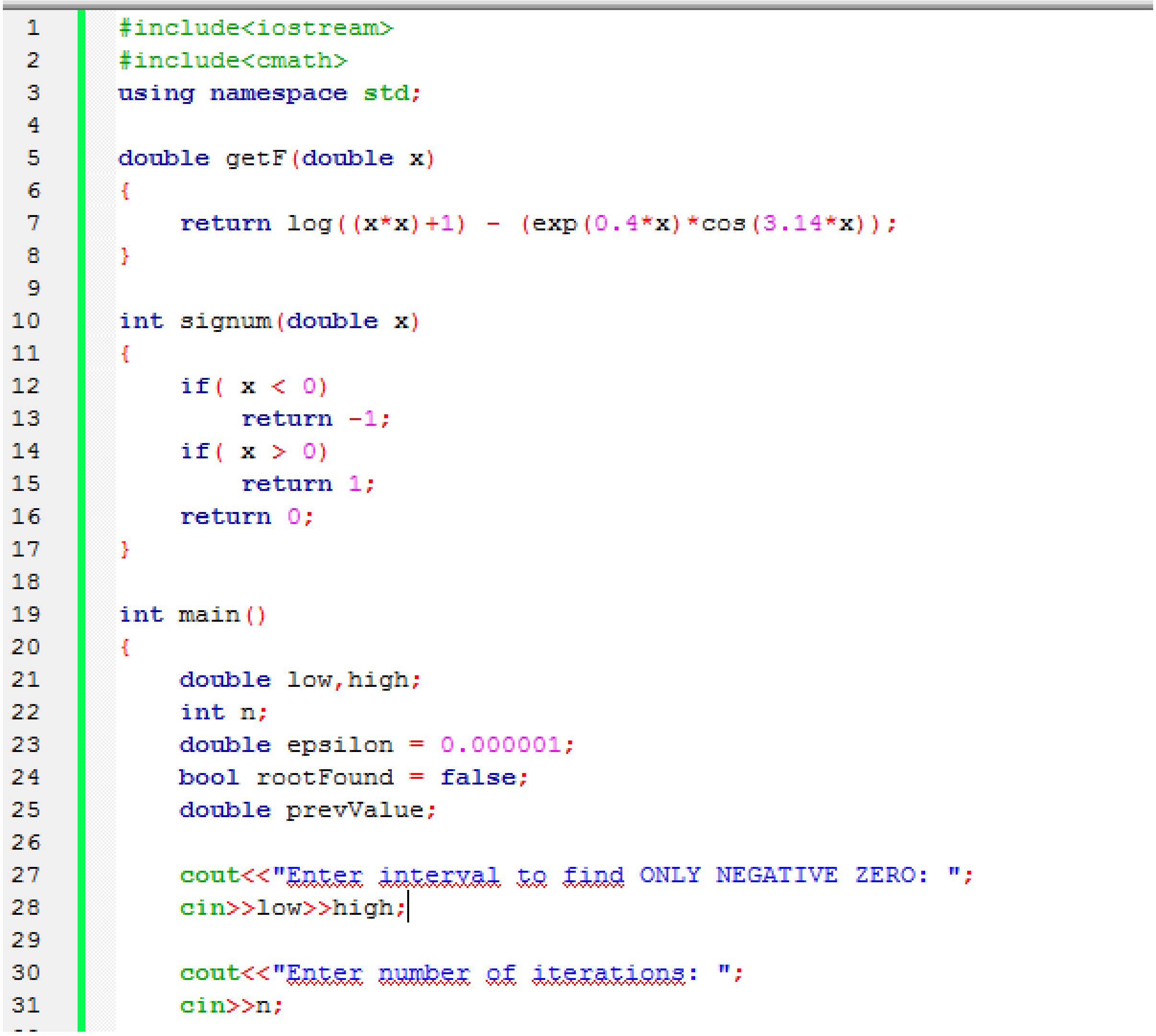
Root Found: 1.49819

Number Of Iterations: 8

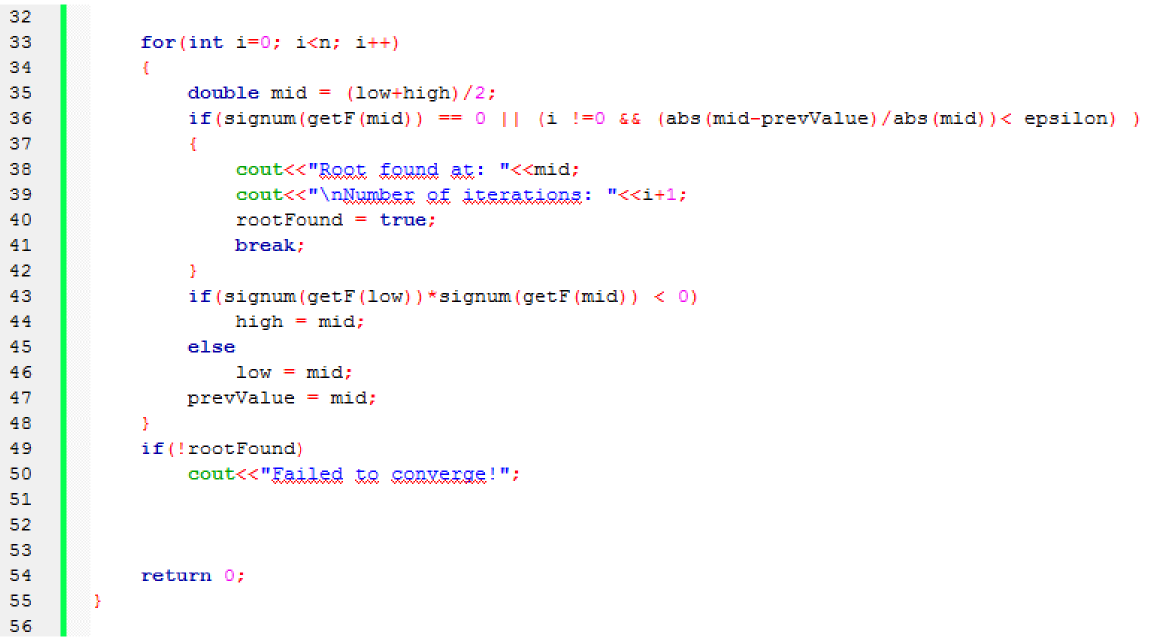

C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 1
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
Root found at: 1.49819
Number of iterations: 8

EVALUATING USING SECANTS METHOD:
Enter second initial value: 2
Root found at: 1.49819
Number of iterations: 8
Process returned 0 (0x0)   execution time : 3.327 s
Press any key to continue.
```

2. SOURCE CODE

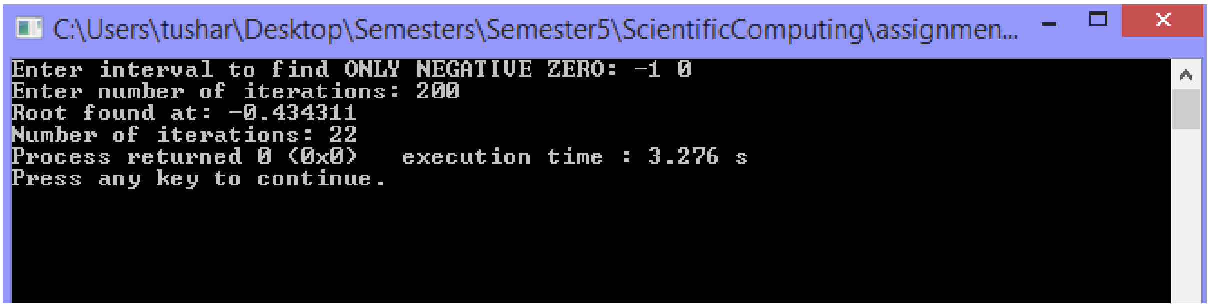


```cpp
#include<iostream>
#include<cmath>
using namespace std;

double getF(double x)
{
    return log((x*x)+1) - (exp(0.4*x)*cos(3.14*x));
}

int signum(double x)
{
    if( x < 0)
        return -1;
    if( x > 0)
        return 1;
    return 0;
}

int main()
{
    double low,high;
    int n;
    double epsilon = 0.000001;
    bool rootFound = false;
    double prevValue;

    cout<<"Enter interval to find ONLY NEGATIVE ZERO: ";
    cin>>low>>high;

    cout<<"Enter number of iterations: ";
    cin>>n;
```



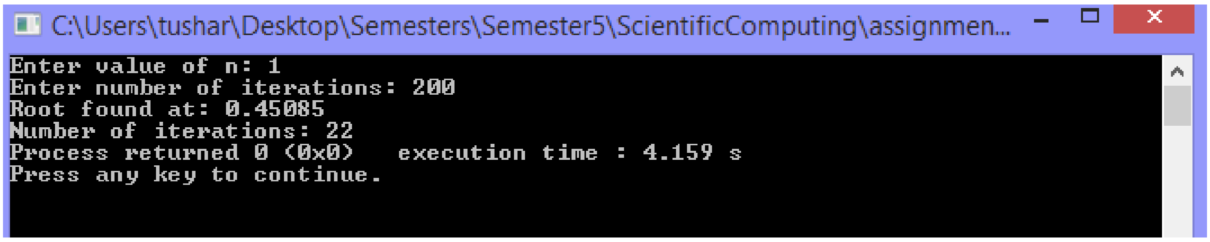
```cpp

    for(int i=0; i<n; i++)
    {
        double mid = (low+high)/2;
        if(signum(getF(mid)) == 0 || (i !=0 && (abs(mid-prevValue)/abs(mid))< epsilon) )
        {
            cout<<"Root found at: "<<mid;
            cout<<"\nNumber of iterations: "<<i+1;
            rootFound = true;
            break;
        }
        if(signum(getF(low))*signum(getF(mid)) < 0)
            high = mid;
        else
            low = mid;
        prevValue = mid;
    }
    if(!rootFound)
        cout<<"Failed to converge!";



    return 0;
}

```

Negative Root: -0.434311

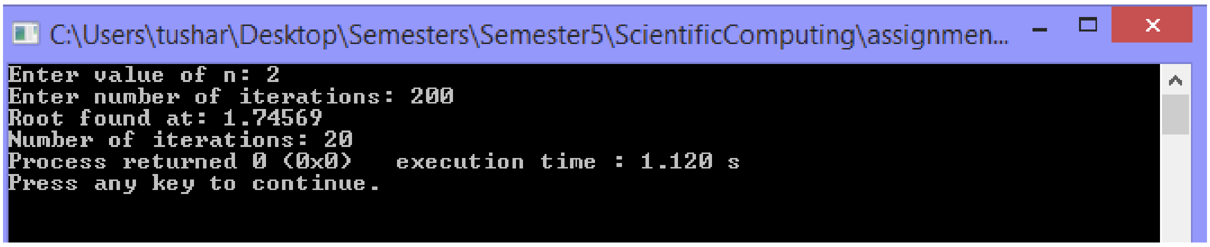

C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter interval to find ONLY NEGATIVE ZERO: -1 0
Enter number of iterations: 200
Root found at: -0.434311
Number of iterations: 22
Process returned 0 (0x0)   execution time : 3.276 s
Press any key to continue.
```

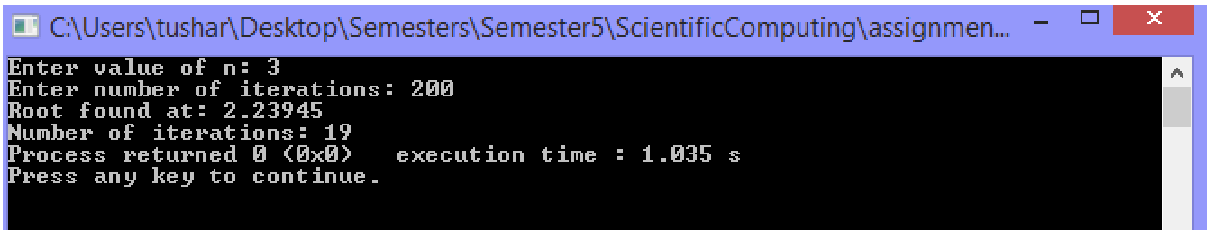
First 4 positive roots:


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

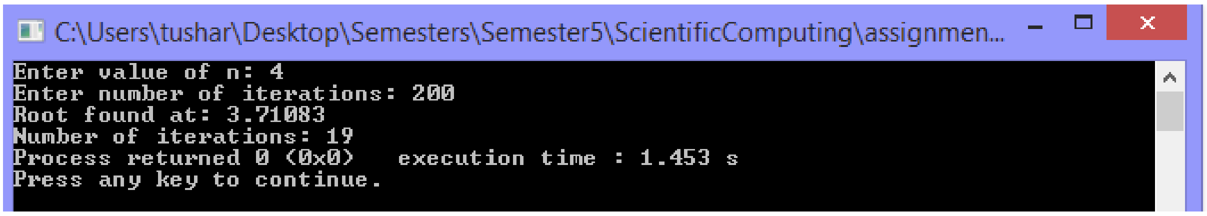
```
Enter value of n: 1
Enter number of iterations: 200
Root found at: 0.45085
Number of iterations: 22
Process returned 0 (0x0)   execution time : 4.159 s
Press any key to continue.
```


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter value of n: 2
Enter number of iterations: 200
Root found at: 1.74569
Number of iterations: 20
Process returned 0 (0x0)   execution time : 1.120 s
Press any key to continue.
```

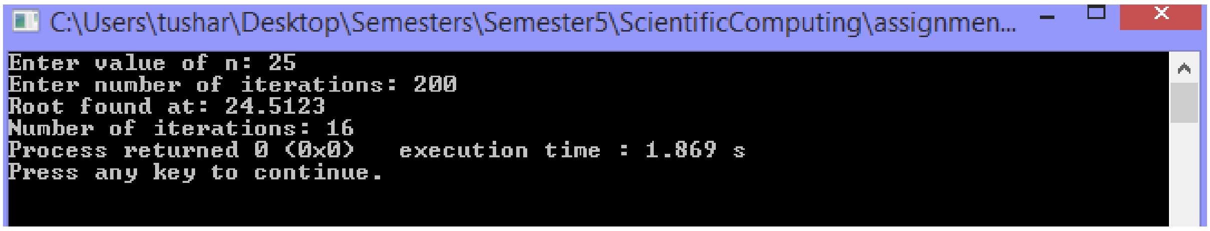

C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter value of n: 3
Enter number of iterations: 200
Root found at: 2.23945
Number of iterations: 19
Process returned 0 (0x0)   execution time : 1.035 s
Press any key to continue.
```

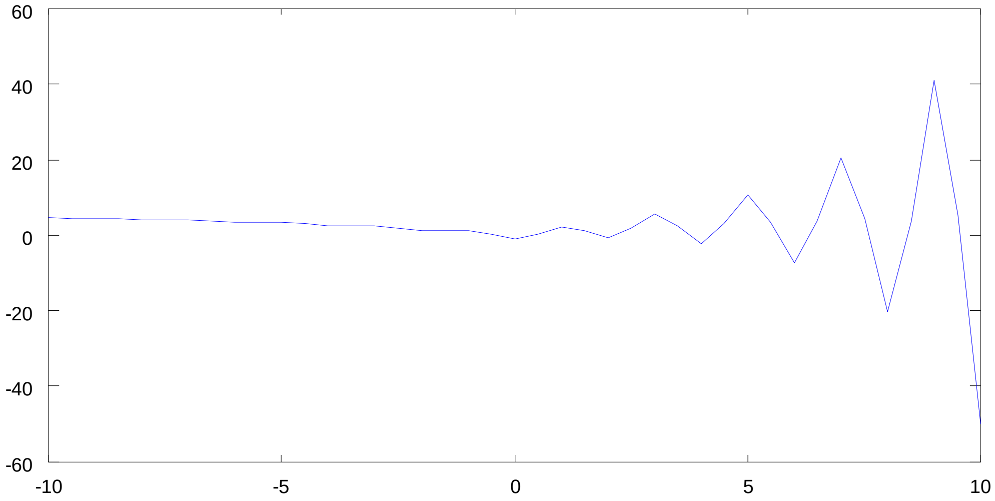

C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter value of n: 4
Enter number of iterations: 200
Root found at: 3.71083
Number of iterations: 19
Process returned 0 (0x0)   execution time : 1.453 s
Press any key to continue.
```

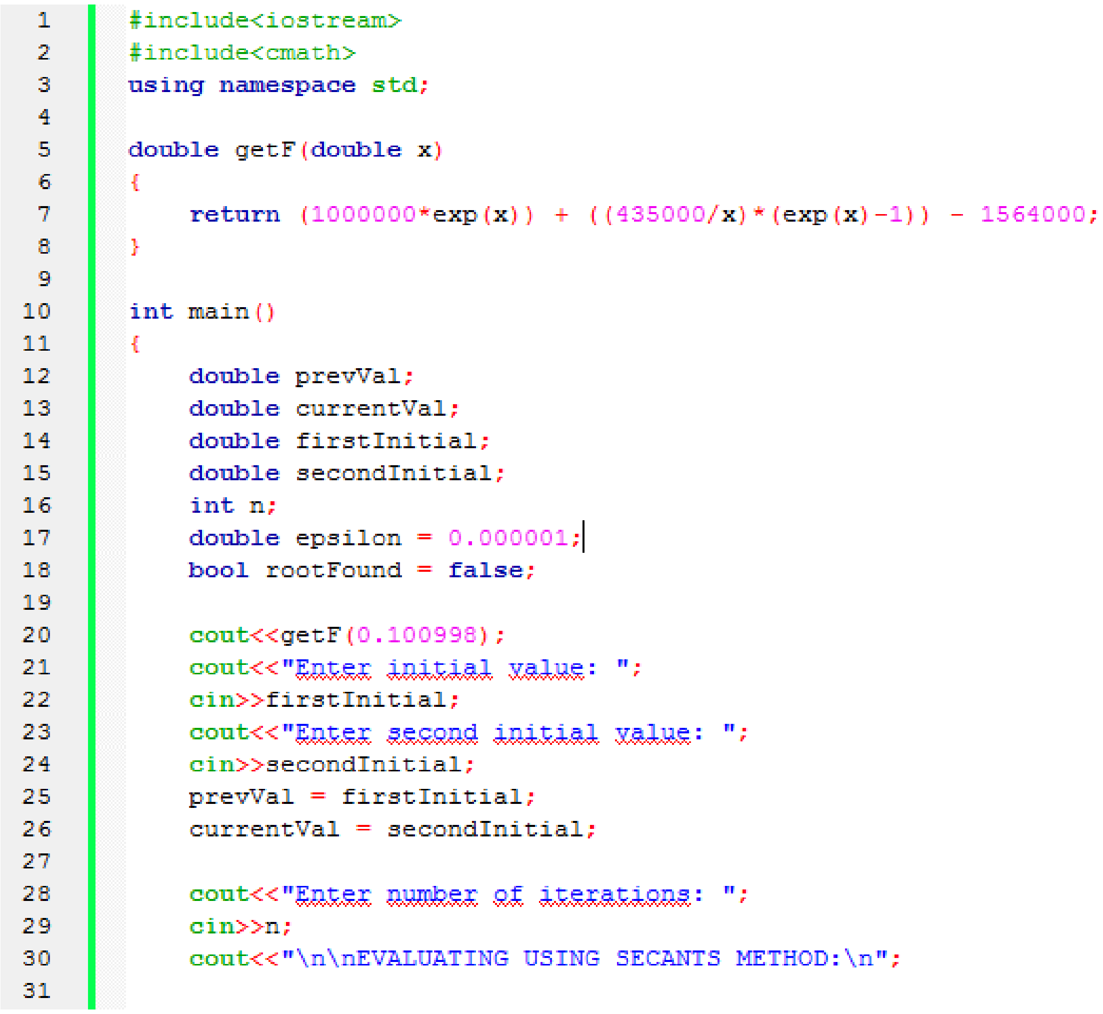
By observation of the graph, nth positive root lies between (n-1) and n. So choosing [n-1,n] as the starting interval for bisection method gives the nth positive root as the answer. Similarly for the 25th root,


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

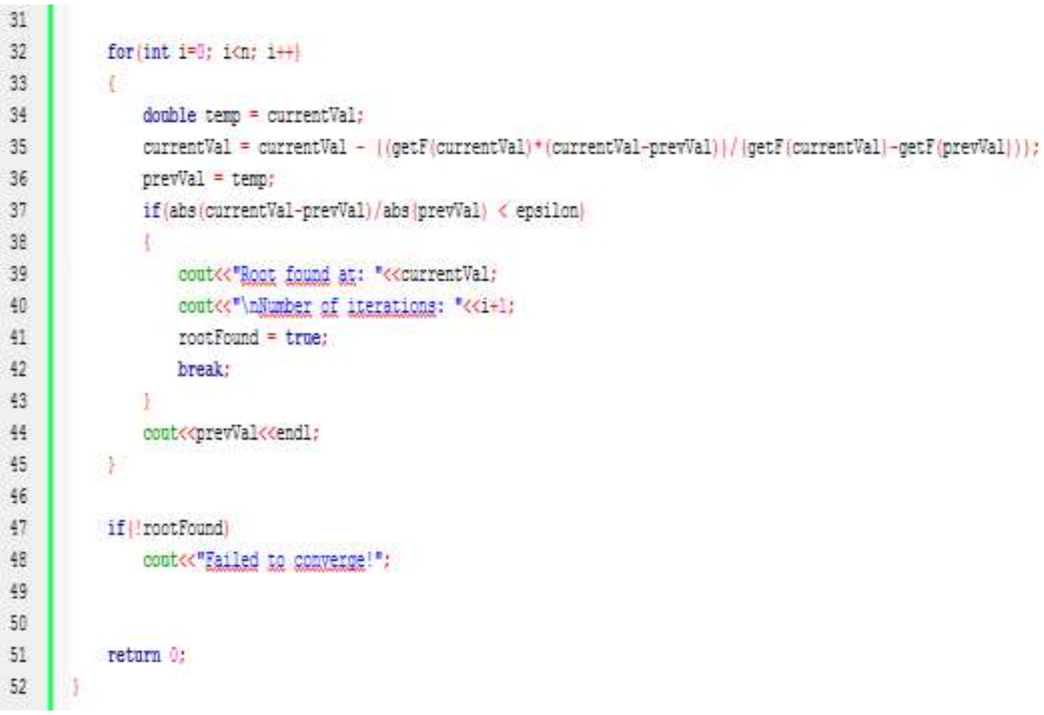
```
Enter value of n: 25
Enter number of iterations: 200
Root found at: 24.5123
Number of iterations: 16
Process returned 0 (0x0)   execution time : 1.869 s
Press any key to continue.
```



3. SOURCE CODE



```
#include<iostream>
#include<cmath>
using namespace std;

double getF(double x)
{
    return (1000000*exp(x)) + ((435000/x)*(exp(x)-1)) - 1564000;
}

int main()
{
    double prevVal;
    double currentVal;
    double firstInitial;
    double secondInitial;
    int n;
    double epsilon = 0.000001;
    bool rootFound = false;

    cout<<getF(0.100998);
    cout<<"Enter initial value: ";
    cin>>firstInitial;
    cout<<"Enter second initial value: ";
    cin>>secondInitial;
    prevVal = firstInitial;
    currentVal = secondInitial;

    cout<<"Enter number of iterations: ";
    cin>>n;
    cout<<"\n\nEVALUATING USING SECANTS METHOD:\n";

```



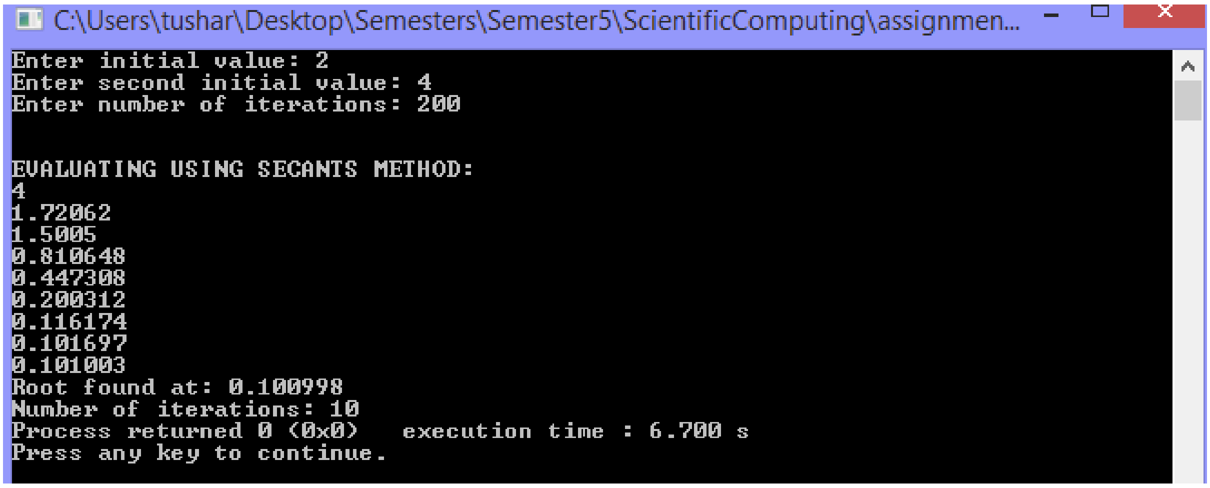
```
    for(int i=0; i<n; i++)
    {
        double temp = currentVal;
        currentVal = currentVal - ((getF(currentVal)*(currentVal-prevVal))/(getF(currentVal)-getF(prevVal)));
        prevVal = temp;
        if(abs(currentVal-prevVal)/abs(prevVal) < epsilon)
        {
            cout<<"Root found at: "<<currentVal;
            cout<<"\nNumber of iterations: "<<i+1;
            rootFound = true;
            break;
        }
        cout<<prevVal<<endl;
    }

    if(!rootFound)
        cout<<"Failed to converge!";


    return 0;
}
```

SECANT METHOD

Initial Values: 2, 4

Root Found: 0.100998

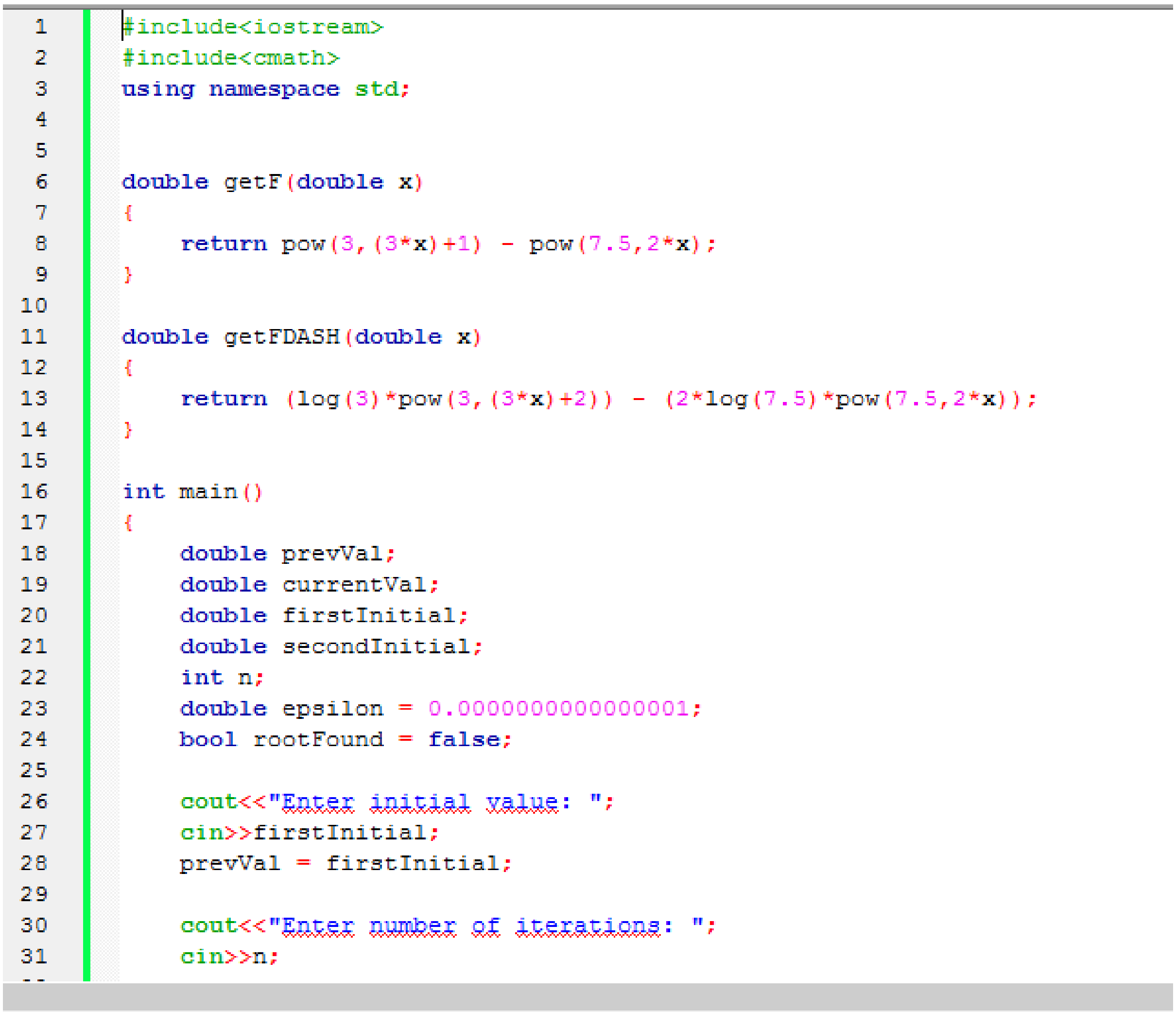
Number Of Iterations: 10



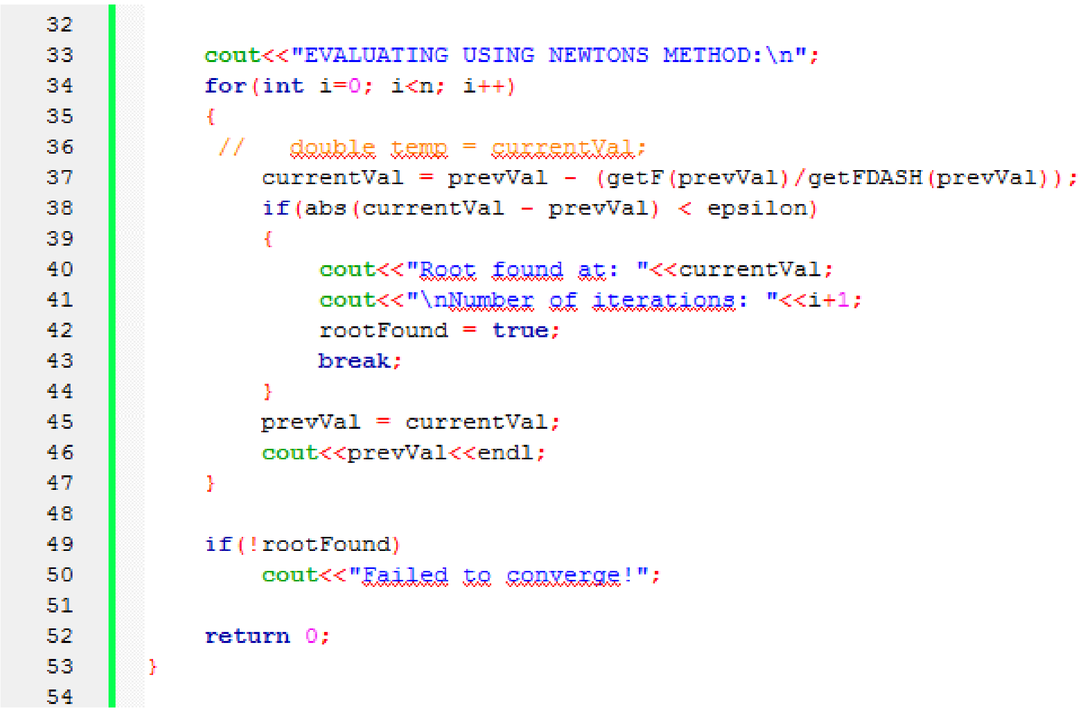
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 2
Enter second initial value: 4
Enter number of iterations: 200


EVALUATING USING SECANTS METHOD:
4
1.72062
1.5005
0.810648
0.447308
0.200312
0.116174
0.101697
0.101003
Root found at: 0.100998
Number of iterations: 10
Process returned 0 (0x0)   execution time : 6.700 s
Press any key to continue.
```

4. SOURCE CODE

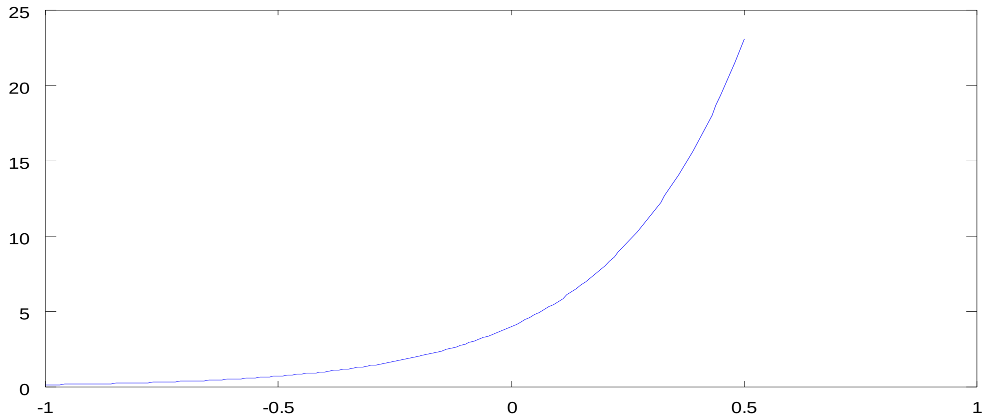


```cpp
#include<iostream>
#include<cmath>
using namespace std;


double getF(double x)
{
    return pow(3,(3*x)+1) - pow(7.5,2*x);
}

double getFDASH(double x)
{
    return (log(3)*pow(3,(3*x)+2)) - (2*log(7.5)*pow(7.5,2*x));
}

int main()
{
    double prevVal;
    double currentVal;
    double firstInitial;
    double secondInitial;
    int n;
    double epsilon = 0.0000000000000001;
    bool rootFound = false;

    cout<<"Enter initial value: ";
    cin>>firstInitial;
    prevVal = firstInitial;

    cout<<"Enter number of iterations: ";
    cin>>n;
```



```cpp

    cout<<"EVALUATING USING NEWTONS METHOD:\n";
    for(int i=0; i<n; i++)
    {
     //   double temp = currentVal;
        currentVal = prevVal - (getF(prevVal)/getFDASH(prevVal));
        if(abs(currentVal - prevVal) < epsilon)
        {
            cout<<"Root found at: "<<currentVal;
            cout<<"\nNumber of iterations: "<<i+1;
            rootFound = true;
            break;
        }
        prevVal = currentVal;
        cout<<prevVal<<endl;
    }

    if(!rootFound)
        cout<<"Failed to converge!";

    return 0;
}

```

Graph of given function:

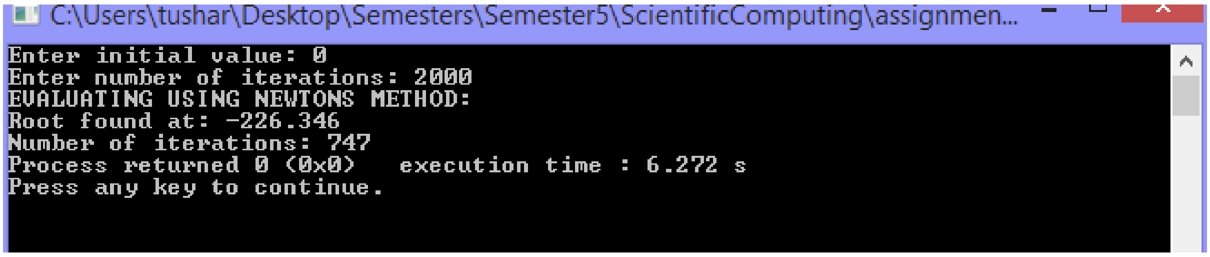


NEWTONS METHOD

Initial Value: 0

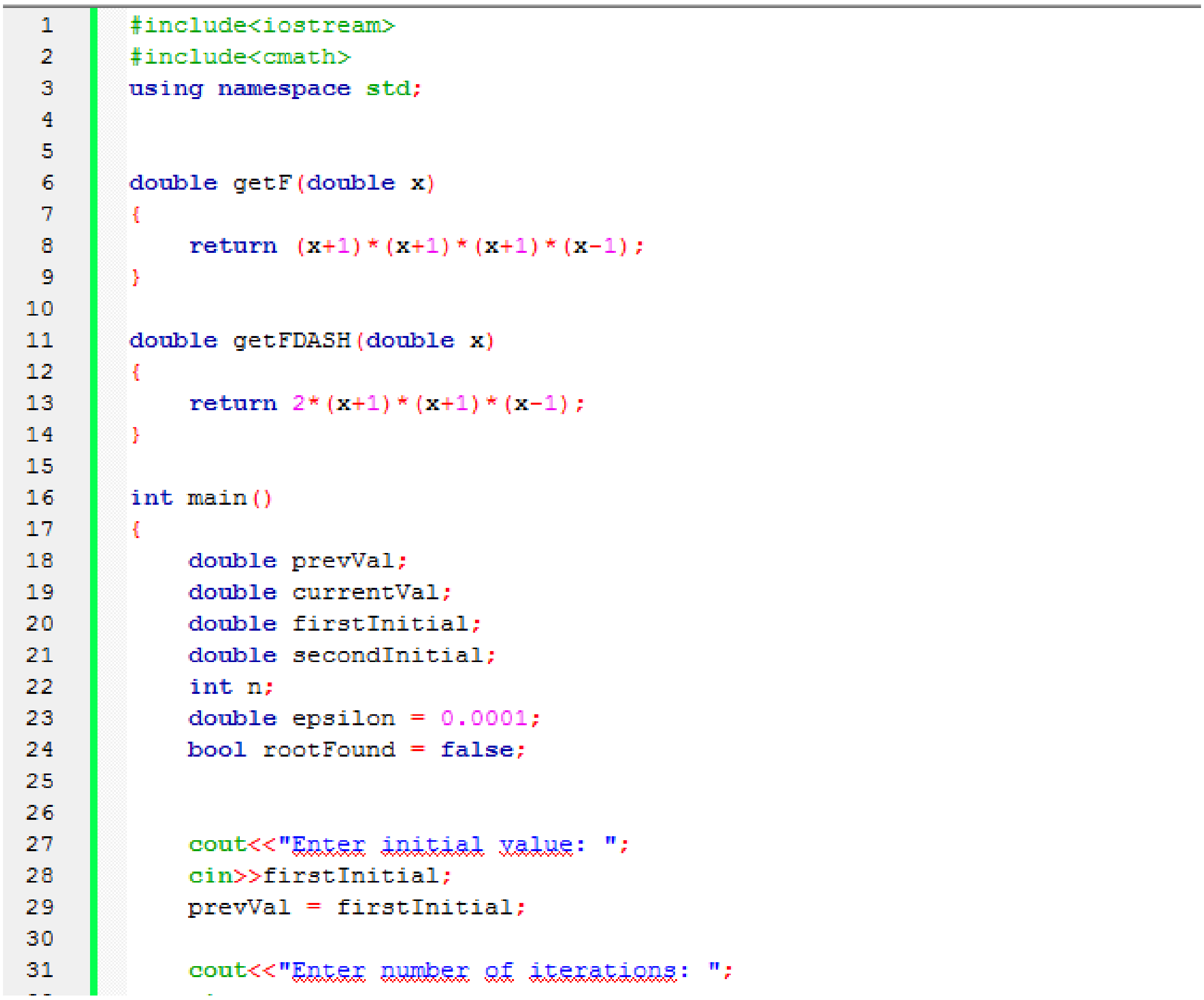
Root Found: -226.346

Number Of Iterations: 747

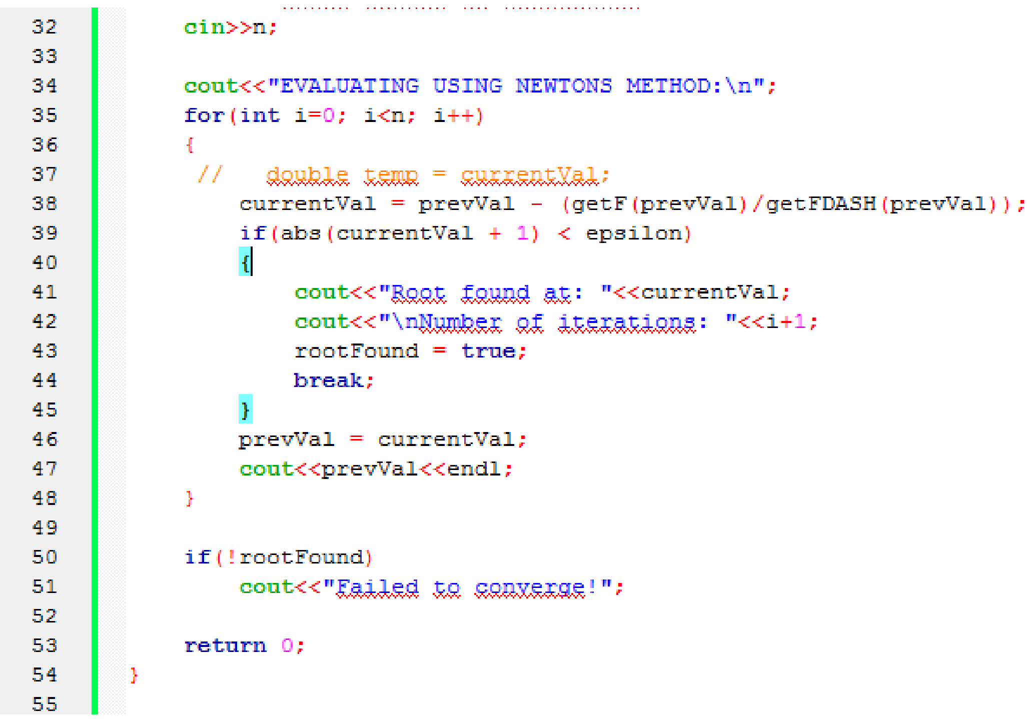

C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 0
Enter number of iterations: 2000
EVALUATING USING NEWTONS METHOD:
Root found at: -226.346
Number of iterations: 747
Process returned 0 (0x0)   execution time : 6.272 s
Press any key to continue.
```

5. SOURCE CODE



```cpp
#include<iostream>
#include<cmath>
using namespace std;


double getF(double x)
{
    return (x+1)*(x+1)*(x+1)*(x-1);
}

double getFDASH(double x)
{
    return 2*(x+1)*(x+1)*(x-1);
}

int main()
{
    double prevVal;
    double currentVal;
    double firstInitial;
    double secondInitial;
    int n;
    double epsilon = 0.0001;
    bool rootFound = false;


    cout<<"Enter initial value: ";
    cin>>firstInitial;
    prevVal = firstInitial;

    cout<<"Enter number of iterations: ";
```



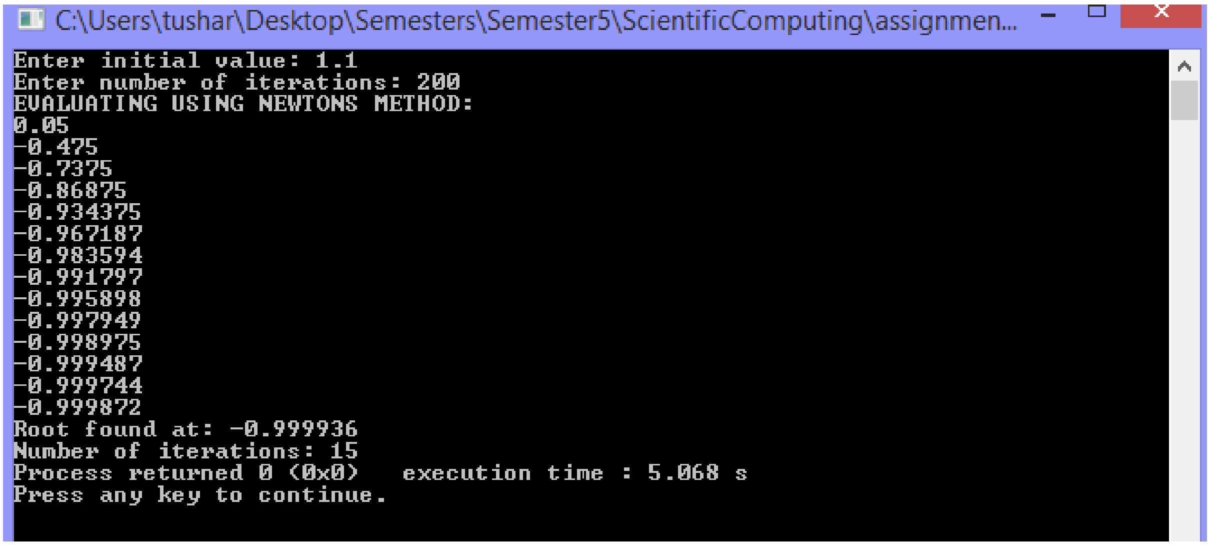
```cpp
    cin>>n;

    cout<<"EVALUATING USING NEWTONS METHOD:\n";
    for(int i=0; i<n; i++)
    {
     //   double temp = currentVal;
        currentVal = prevVal - (getF(prevVal)/getFDASH(prevVal));
        if(abs(currentVal + 1) < epsilon)
        {
            cout<<"Root found at: "<<currentVal;
            cout<<"\nNumber of iterations: "<<i+1;
            rootFound = true;
            break;
        }
        prevVal = currentVal;
        cout<<prevVal<<endl;
    }

    if(!rootFound)
        cout<<"Failed to converge!";

    return 0;
}
```

NEWTON METHOD

Initial Value: 1.1

Root Found: -0.999936

Number Of Iterations: 15



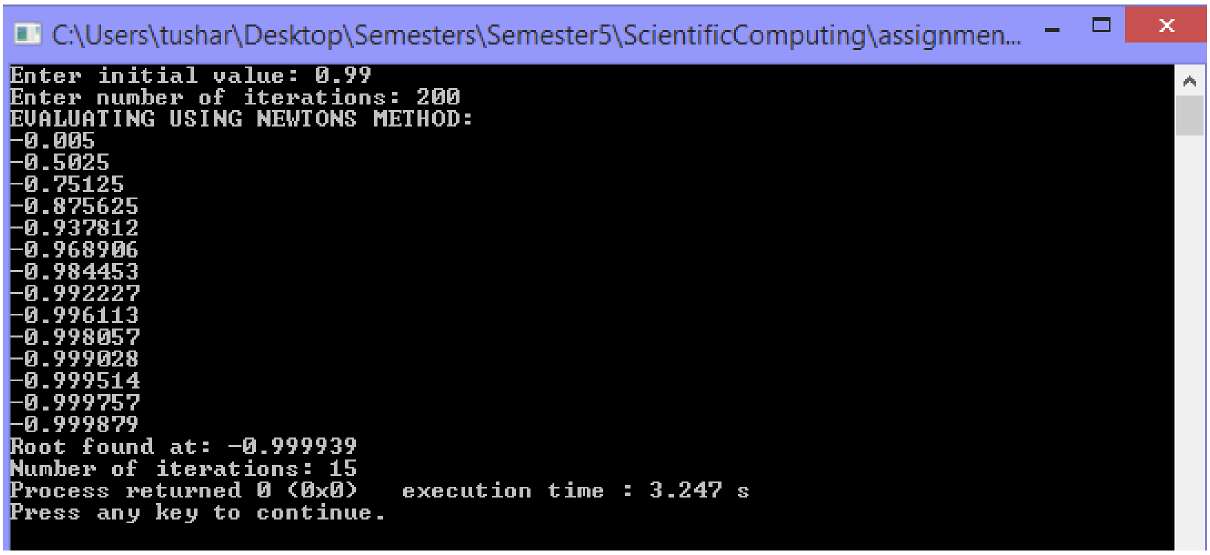
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 1.1
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
0.05
-0.475
-0.7375
-0.86875
-0.934375
-0.967187
-0.983594
-0.991797
-0.995898
-0.997949
-0.998975
-0.999487
-0.999744
-0.999872
Root found at: -0.999936
Number of iterations: 15
Process returned 0 (0x0)   execution time : 5.068 s
Press any key to continue.
```

Initial Value: 0.99

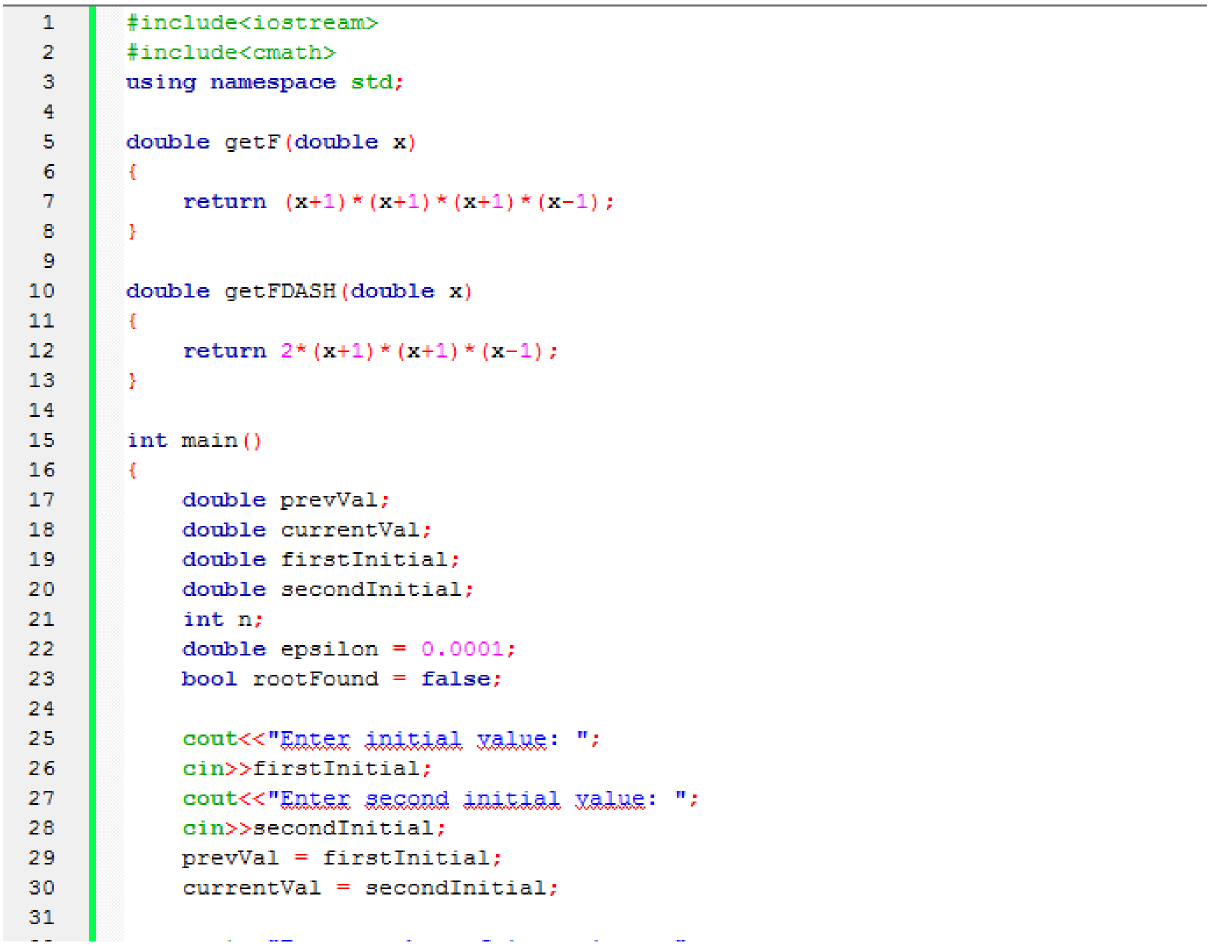
Root Found: -0.999939

Number Of Iterations: 15



C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 0.99
Enter number of iterations: 200
EVALUATING USING NEWTONS METHOD:
-0.005
-0.5025
-0.75125
-0.875625
-0.937812
-0.968906
-0.984453
-0.992227
-0.996113
-0.998057
-0.999028
-0.999514
-0.999757
-0.999879
Root found at: -0.999939
Number of iterations: 15
Process returned 0 (0x0)   execution time : 3.247 s
Press any key to continue.
```



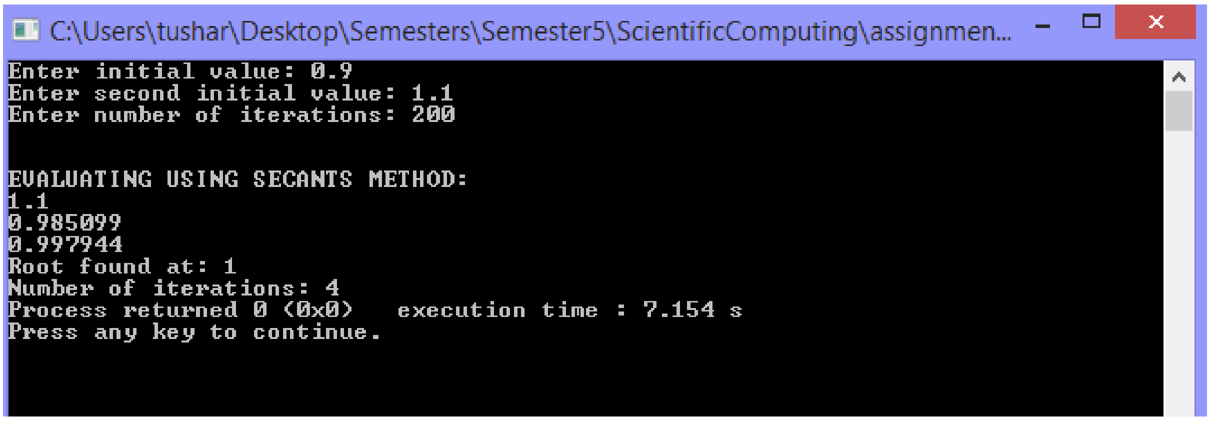
```
#include<iostream>
#include<cmath>
using namespace std;

double getF(double x)
{
    return (x+1)*(x+1)*(x+1)*(x-1);
}

double getFDASH(double x)
{
    return 2*(x+1)*(x+1)*(x-1);
}

int main()
{
    double prevVal;
    double currentVal;
    double firstInitial;
    double secondInitial;
    int n;
    double epsilon = 0.0001;
    bool rootFound = false;

    cout<<"Enter initial value: ";
    cin>>firstInitial;
    cout<<"Enter second initial value: ";
    cin>>secondInitial;
    prevVal = firstInitial;
    currentVal = secondInitial;

```

SECANT METHOD

Initial Values: 0.9, 1.1

Root Found: 1

Number Of Iterations: 4



C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...

```
Enter initial value: 0.9
Enter second initial value: 1.1
Enter number of iterations: 200


EVALUATING USING SECANTS METHOD:
1.1
0.985099
0.997944
Root found at: 1
Number of iterations: 4
Process returned 0 (0x0)   execution time : 7.154 s
Press any key to continue.
```