Bisection method:

Code:

```
#include<bits/stdc++.h>


using namespace std;


#define max_iteration 100


double equation(double x,double y, double z, double i){
    return ((i*i*x) + (y*i) + z);
}


void bisection(double a, double b,double x,double y, double z){
    double x0 = (a+b)/2.0;
    double fx0,fa,xold;
    int count = 0;
    while(true){
        cout<<count<<" : "<<x0<<endl;
        fx0 = equation(x,y,z,x0);
        fa = equation(x,y,z,a);
        if(fx0<1e-9 && fx0>(-1e-9)){
            cout<<"The root is "<<x0<<endl;
            break;
        }
        if((fx0 * fa) < 0){
            cout<<x0<<endl;
            xold = x0;
            x0 = (a+x0)/2.0;
            b = xold;
        }
        else{
            cout<<x0<<endl;
            xold = x0;
            x0 = (b+x0)/2.0;
            a = xold;
        }
        count++;
    }
}

int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout<<setprecision(9);
    bool flag = 0,found=0;
    double x,y,z;
    cin>>x>>y>>z;
    double a,b,fa,fb,fx;
    for(int i = 0;i<max_iteration;i++){
        fa = equation(x,y,z,i);
        fb = equation(x,y,z,i+1);
        if(fa<1e-9 && fa>(-1e-9)){
            cout<<"The root is "<<i<<endl;
            found=1;
            break;
        }
        if(fb<1e-9 && fb>(-1e-9)){
            cout<<"The root is "<<i+1<<endl;
            found=1;
            break;
        }
        if((fa*fb)<0){
            a = i;
            b = i+1;
            flag = 1;
            break;
        }
    }
    if(flag==0){
        for(int i = 0;i>(-max_iteration);i--){
        fa = equation(x,y,z,i);
        fb = equation(x,y,z,i-1);
        if(fa<1e-9 && fa>(-1e-9)){
            cout<<"The root is "<<i<<endl;
            found=1;
            break;
        }
        if(fb<1e-9 && fb>(-1e-9)){
            cout<<"The root is "<<i-1<<endl;
            found=1;
            break;
        }
        if((fa*fb)<0){
            a = i;
            b = i-1;
            break;
        }
    }
    }
    if(found==0) bisection(a,b,x,y,z);
}
```

False position method:

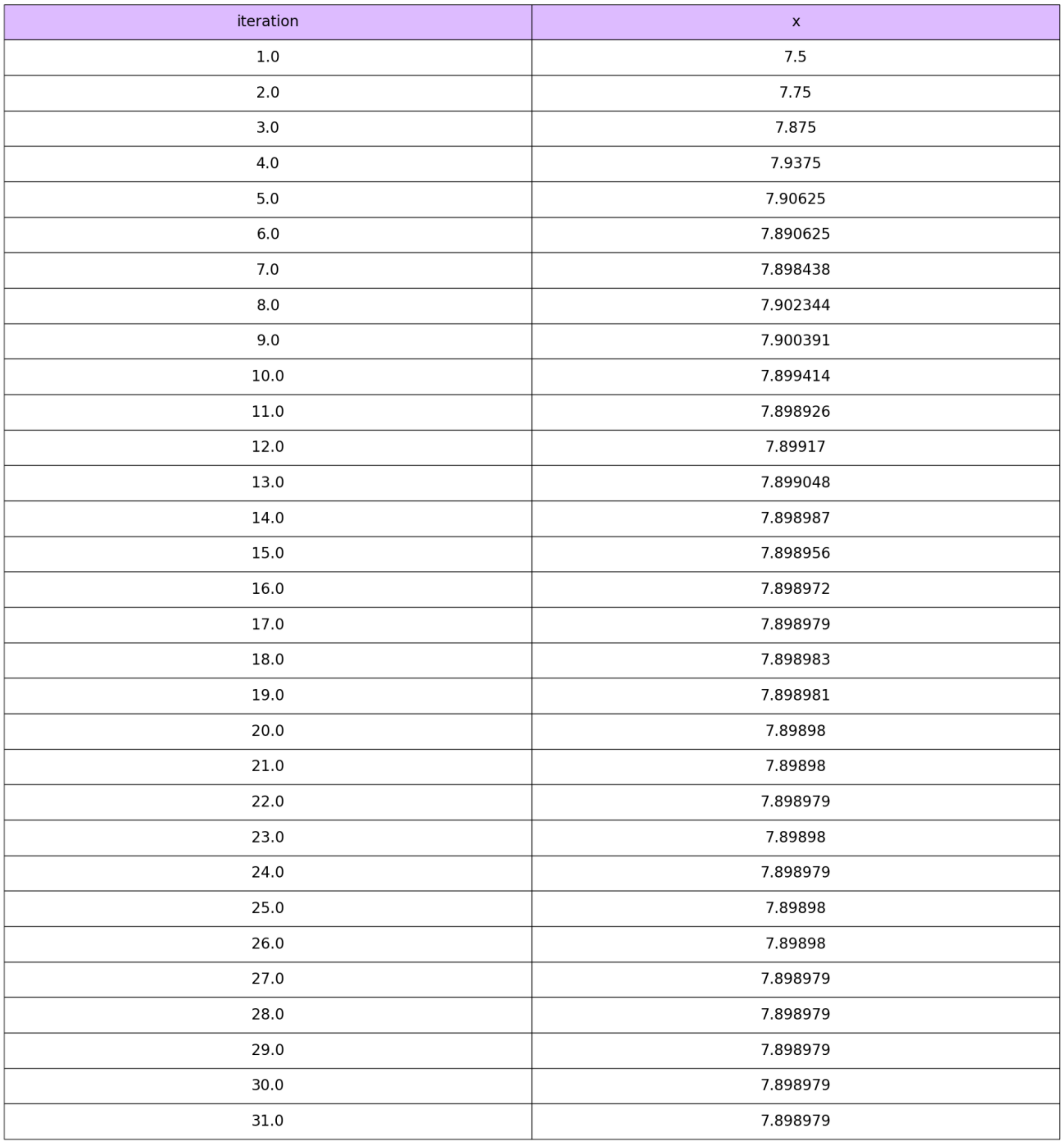
Code:

```
#include<bits/stdc++.h>


using namespace std;


#define max_iteration 100


double equation(double x,double y, double z, double i){
    return ((i*i*x) + (y*i) + z);
}


void falseposition(double a, double b,double x,double y, double z){
    double fa = equation(x,y,z,a), fb = equation(x,y,z,b);
    double x0 = ((a*fb)-(b*fa))/(fb-fa);
    double fx0,xold;
    int count = 0;
    while(true){
        cout<<count<<" : "<<x0<<endl;
        fx0 = equation(x,y,z,x0);
        fa = equation(x,y,z,a);
        if(fx0<1e-9 && fx0>(-1e-9)){
            cout<<"The root is "<<x0<<endl;
            break;
        }
        if((fx0 * fa) < 0){
            cout<<x0<<endl;
            xold = x0;
            x0 = ((a*fx0)-(x0*fa))/(fx0-fa);;
            b = xold;
        }
        else{
            cout<<x0<<endl;
            xold = x0;
            x0 = ((x0*fb)-(b*fx0))/(fb-fx0);
            a = xold;
        }
        count++;
    }
}
```

```
int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cout<<setprecision(9);
    bool flag = 0,found=0;
    double x,y,z;
    cin>>x>>y>>z;
    double a,b,fa,fb,fx;
    for(int i = 0;i<max_iteration;i++){
        fa = equation(x,y,z,i);
        fb = equation(x,y,z,i+1);
        if(fa<1e-9 && fa>(-1e-9)){
            cout<<"The root is "<<i<<endl;
            found=1;
            break;
        }
        if(fb<1e-9 && fb>(-1e-9)){
            cout<<"The root is "<<i+1<<endl;
            found=1;
            break;
        }
        if((fa*fb)<0){
            a = i;
            b = i+1;
            flag = 1;
            break;
        }
    }
    if(flag==0){
        for(int i = 0;i>(-max_iteration);i--){
        fa = equation(x,y,z,i);
        fb = equation(x,y,z,i-1);
        if(fa<1e-9 && fa>(-1e-9)){
            cout<<"The root is "<<i<<endl;
            found=1;
            break;
        }
        if(fb<1e-9 && fb>(-1e-9)){
            cout<<"The root is "<<i-1<<endl;
            found=1;
            break;
        }
        if((fa*fb)<0){
            a = i;
            b = i-1;
            break;
        }
    }
    }
    if(found==0) falseposition(a,b,x,y,z);
}
```

Comparison Table:

1. Bisection Method:

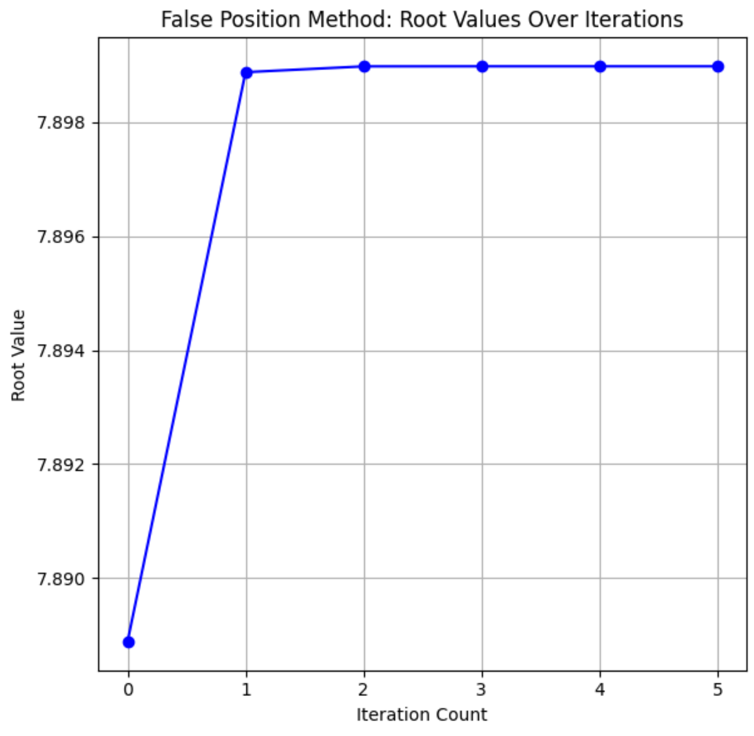


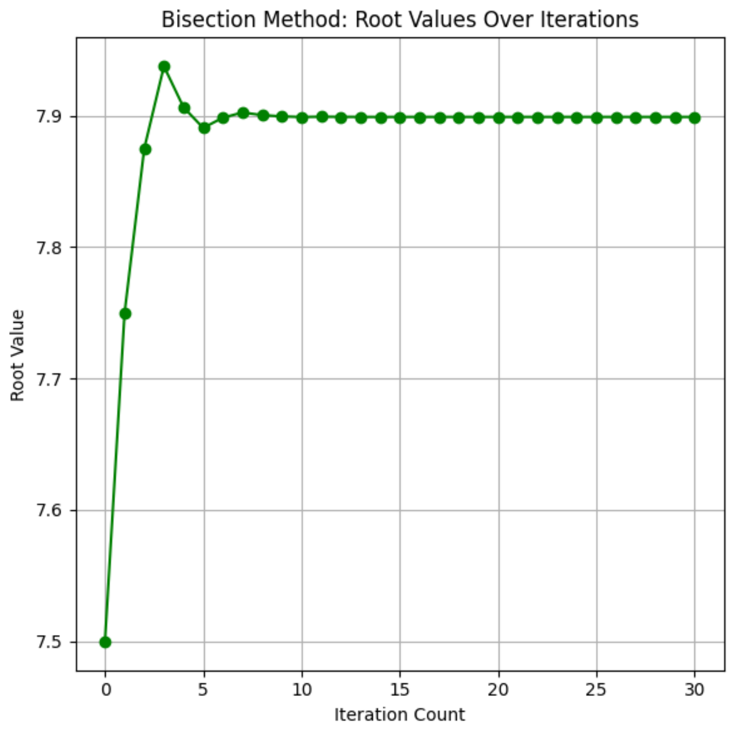
iteration	x
1.0	7.5
2.0	7.75
3.0	7.875
4.0	7.9375
5.0	7.90625
6.0	7.890625
7.0	7.898438
8.0	7.902344
9.0	7.900391
10.0	7.899414
11.0	7.898926
12.0	7.89917
13.0	7.899048
14.0	7.898987
15.0	7.898956
16.0	7.898972
17.0	7.898979
18.0	7.898983
19.0	7.898981
20.0	7.89898
21.0	7.89898
22.0	7.898979
23.0	7.89898
24.0	7.898979
25.0	7.89898
26.0	7.89898
27.0	7.898979
28.0	7.898979
29.0	7.898979
30.0	7.898979
31.0	7.898979

2. False position method:

iteration	x
1.0	7.888889
2.0	7.898876
3.0	7.898978
4.0	7.898979
5.0	7.898979
6.0	7.898979

Comparison Graph:


False Position Method: Root Values Over Iterations
Root Value
Iteration Count


Bisection Method: Root Values Over Iterations
Root Value
Iteration Count