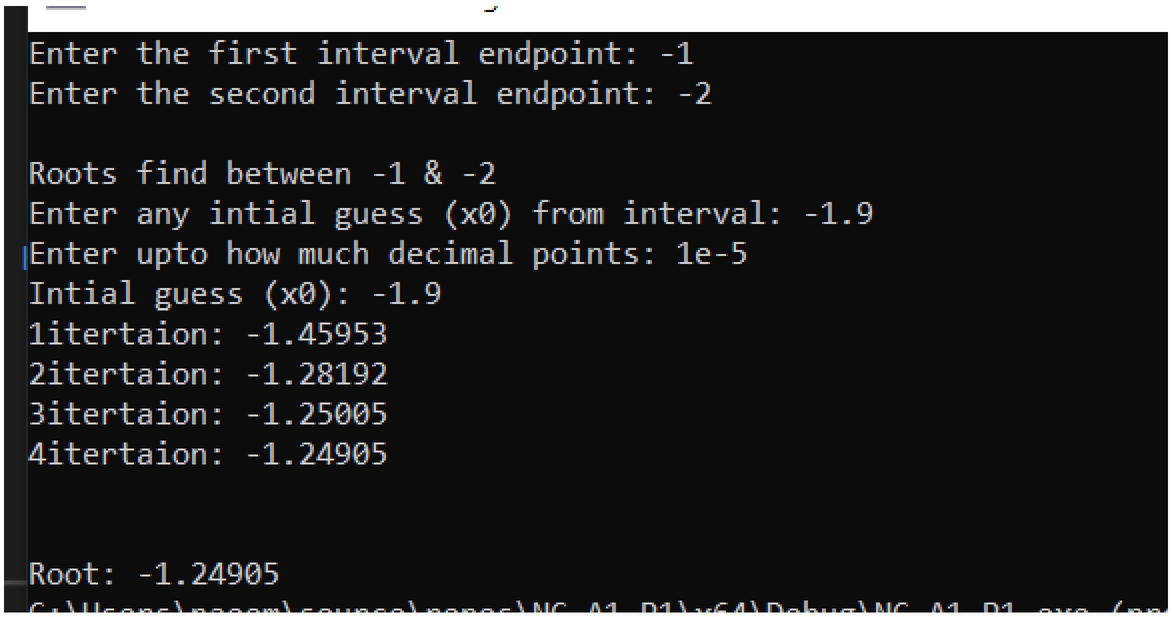
M.NAEEM F228816 A-01 CS-4D

Q1

NEWTON'S RAPSON

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

//defining or set f(x)
double func(double x) {
    return (sin(x) - 1 - pow(x, 3));
}

//Defining or setting f_prime(x) w.r.t 'x'
double fun_prime(double x)
{
    return (cos(x) - 3 * pow(x, 2));
}

double newton_raphson( double initial_guess, double tolerance)
{
    int count = 1;
        double x0 = initial_guess;
        double x1 = x0 - func(x0) / fun_prime(x0);
        cout << "Intial guess (x0): " << x0 << endl;
        while (abs(x1 - x0) > tolerance)
        {
            x0 = x1;
            x1 = x0 - func(x0) / fun_prime(x0);
            cout << count << "itertaion: "  << x0 << endl;
            count++;
        }
        cout << "\n\nRoot: " << x1;

        return x1;
}

    bool root_in_interval(double x1, double x2) {
        double y1 = func(x1);
        double y2 = func(x2);



        if ((y1 < 0 && y2 > 0) || (y1 > 0 && y2 < 0))
        {
            cout << "\nRoots find between " << x1 << " & " << x2;
            return true;
        }
        else
        {
            cout << "\nSorry! Roots does not lies between these interval \n Try
again.";
```

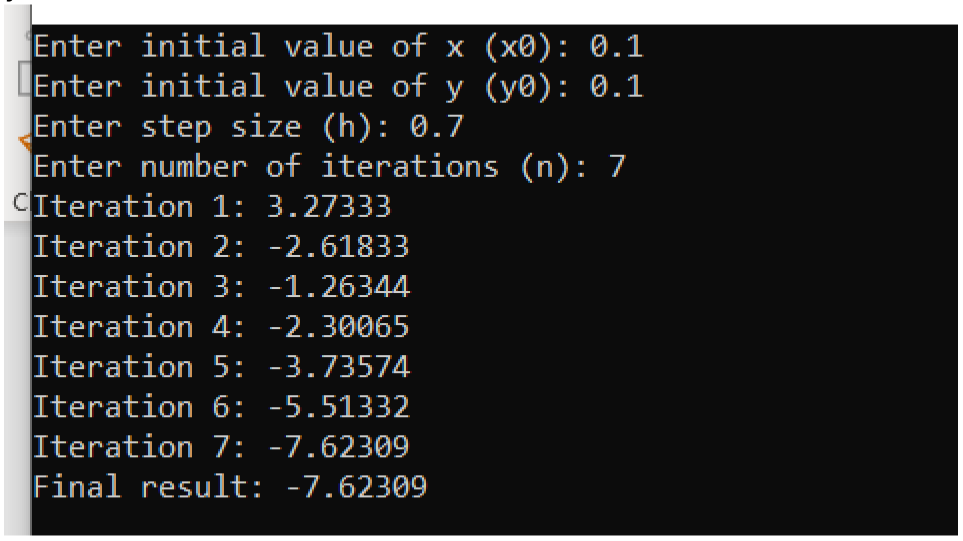
```cpp
            return false;
        }
    }


int main()
{
    double x1, x2,ig,tol;
    while (true)
    {
        cout << "Enter the first interval endpoint: ";
        cin >> x1;
        cout << "Enter the second interval endpoint: ";
        cin >> x2;
        if (root_in_interval(x1, x2))
        {
            break;
        }
    }

    cout << "\nEnter any intial guess (x0) from interval: ";
    cin >> ig;
    cout << "Enter upto how much decimal points: ";
    cin >> tol;
    newton_raphson(ig, tol);


    return 0;
}
```



```
Enter the first interval endpoint: -1
Enter the second interval endpoint: -2

Roots find between -1 & -2
Enter any intial guess (x0) from interval: -1.9
Enter upto how much decimal points: 1e-5
Intial guess (x0): -1.9
1itertaion: -1.45953
2itertaion: -1.28192
3itertaion: -1.25005
4itertaion: -1.24905


Root: -1.24905
C:\Users\naeem\source\repos\NC_A1_P1\x64\Debug\NC_A1_P1.exe (pro
```

EULER'S METHOD

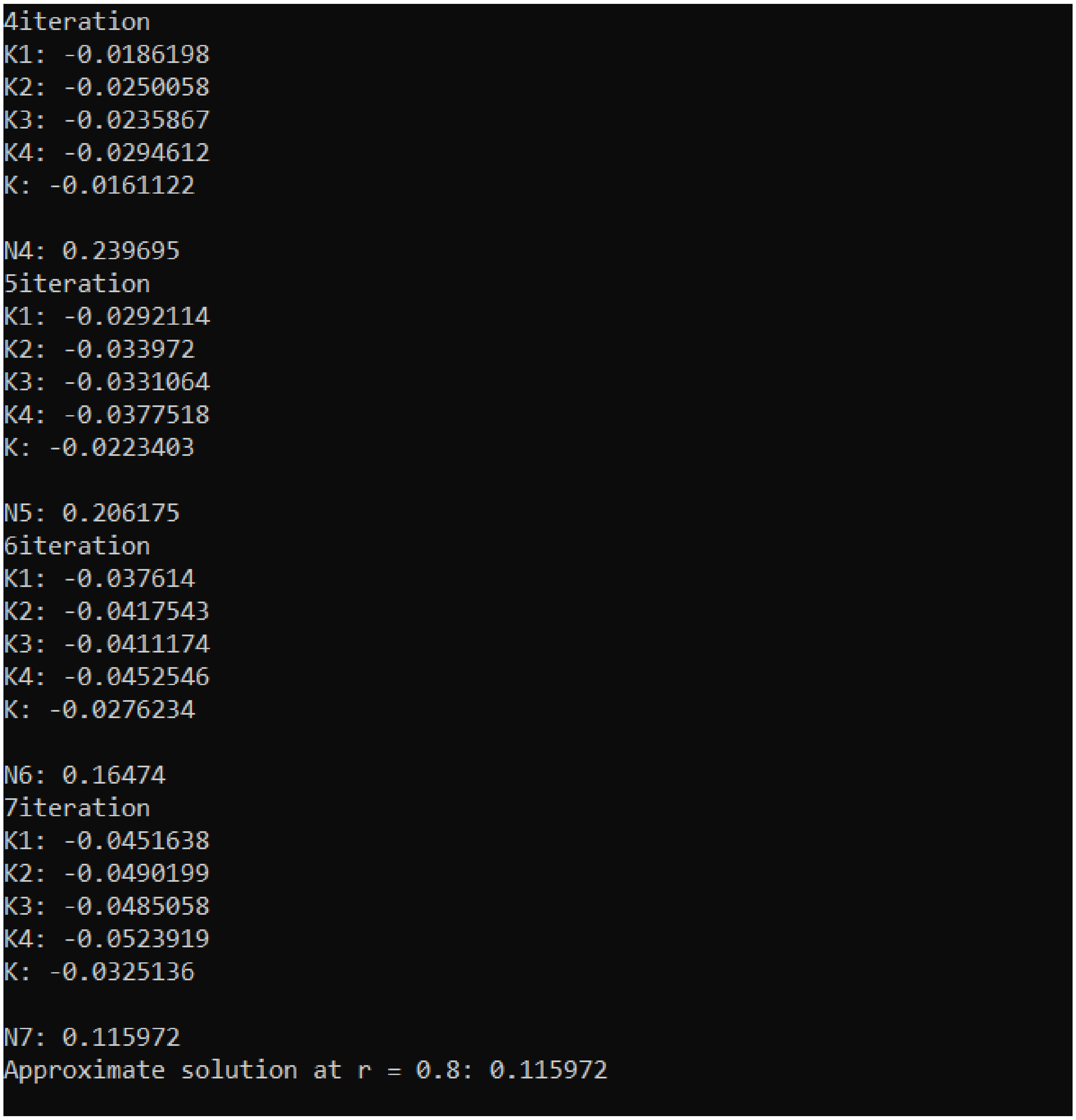
```
#include<iostream>
#include<cmath>
using namespace std;

// Defining or setting f(r, N)
double func(double r, double N)
{
    int  B = 1;
    if (r == 0)
    {
        cout << "Error: Division bN zero encountered." << endl;
        return NAN;
    }
    return ((-2 * N / r) + (2 / (3 * r)) - (4 *B* r / 3));
}

float eulersMethod(float r0, float N0, float h, int n)
{
    float r = r0;
    float N = N0;
    for (int i = 1; i <= n; i++)
    {
        N = N + h * func(r, N);
        r = r + h;
        cout << "Iteration " << i << ": " << N << endl;
    }
    return N;
}

int main()
{
    float r0, N0, h;
    int n;


    cout << "Enter initial value of r (r0): ";
    cin >> r0;
    cout << "Enter initial value of N (N0): ";
    cin >> N0;
    cout << "Enter step size (h): ";
    cin >> h;
    cout << "Enter number of iterations (n): ";
    cin >> n;


    if (h <= 0)
    {
        cout << "Error: Step size (h) must be positive." << endl;
        return 1;
    }


    if (n < 0)
    {
        cout << "Error: Number of iterations (n) must be non-negative." << endl;
        return 1;
    }


    float result = eulersMethod(r0, N0, h, n);
    cout << "Final result: " << result << endl;

    return 0;
}
```



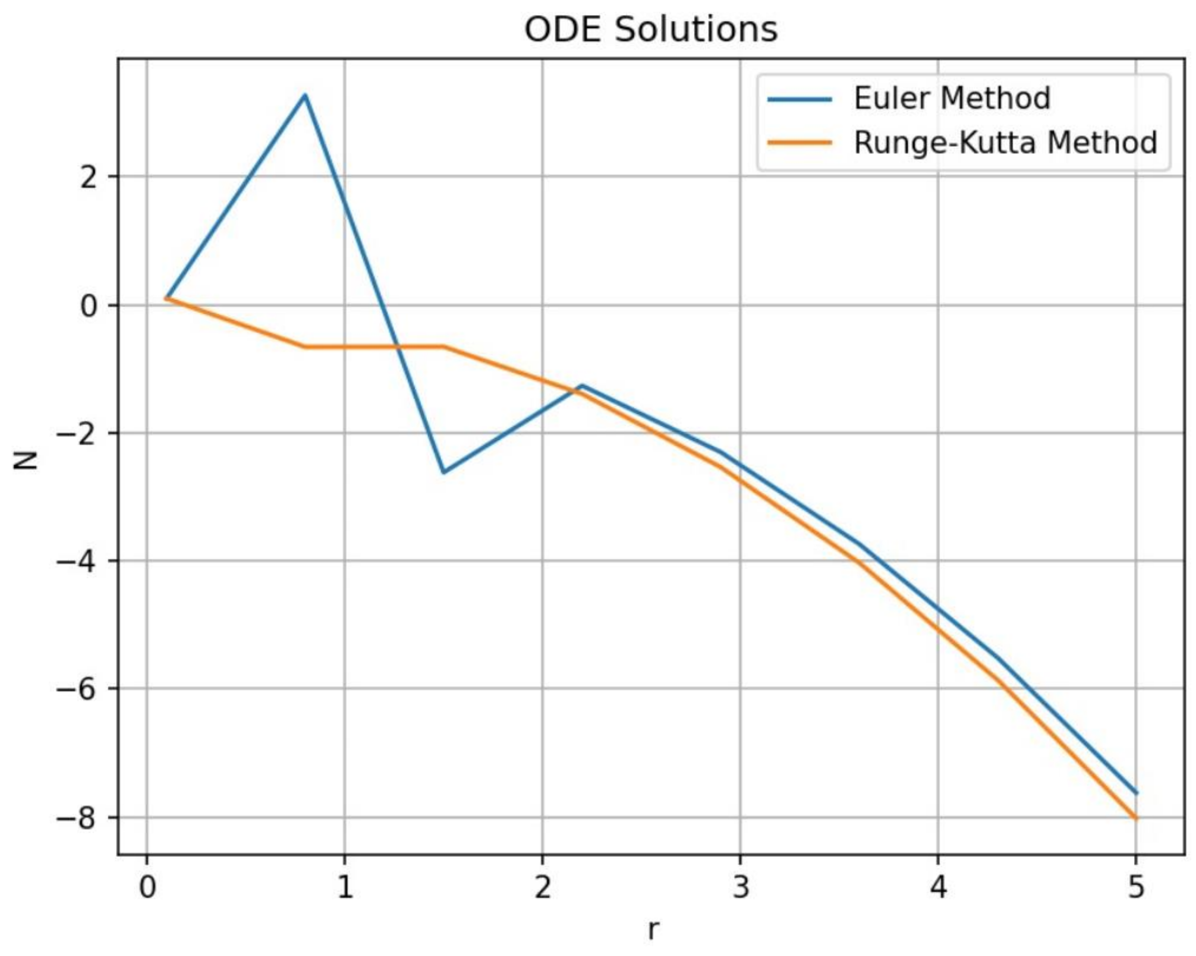
```
Enter initial value of x (x0): 0.1
Enter initial value of y (y0): 0.1
Enter step size (h): 0.7
Enter number of iterations (n): 7
Iteration 1: 3.27333
Iteration 2: -2.61833
Iteration 3: -1.26344
Iteration 4: -2.30065
Iteration 5: -3.73574
Iteration 6: -5.51332
Iteration 7: -7.62309
Final result: -7.62309
```

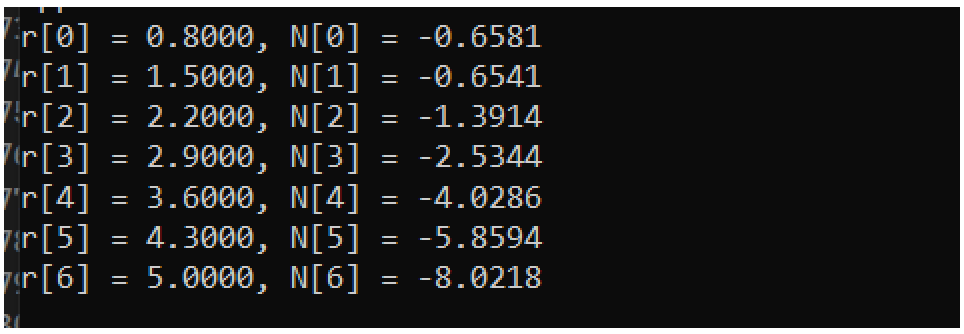
RK-METHOD

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

// Defining or setting f(r, N)
double func(double r, double N)
{
    int  B = 1;
    if (r == 0)
    {
        cout << "Error: Division bN zero encountered." << endl;
        return NAN;
    }
    return ((-2 * N / r) + (2 / (3 * r)) - (4 *B* r / 3));
}

double RK(double h, int n, double N0,double r0, double r_values[], double
N_values[])
{
    double N = N0;
    double r = r0;
    int count = 1;

    for (int i = 0; i < n; i++)
    {
        double K1 = h * func(r, N);
        double K2 = h * func(r + 0.5 * h, N + 0.5 * K1);
        double K3 = h * func(r + 0.5 * h, N + 0.5 * K2);
        double K4 = h * func(r + h, N + K3);
        cout << endl << count << "iteration" << endl;
        cout << "K1: " << K1 << endl;
        cout << "K2: " << K2 << endl;
        cout << "K3: " << K3 << endl;
        cout << "K4: " << K4 << endl;
        double k = (K1 + K2 + K3 + K4) / 6;
        cout << "K: " << k << endl;
        N =N+ (K1 + 2 * K2 + 2 * K3 + K4) / 6;
        cout << "\nN" << count <<": " << N;

        r =r+ h;
        count++;

        N_values[i] = N;
        r_values[i] = r;

    }
    return N;
}
int main() {
    double r0, h, N0;
    int n,x1,x2,precision;

    cout << "Enter the domain of f(x,y)";
    cin >> x1 >> x2;
    cout << "doamin is: " << x1 <<" & " << x2 << endl;
    cout << "Enter step size (h): ";
    cin >> h;
    cout << "Enter number of steps-max iteration you want (n): ";
    cin >> n;
    cout << "Enter initial value of y (N0): ";
    cin >> N0;
    cout << "Enter initial value of y (r0): ";
    cin >> r0;
    cout << "Enter the number of decimal points to display: ";
    cin >> precision;

    double* r_values = new double[n];
    double* N_values = new double[n];
    double result = RK(h, n, N0,r0, r_values, N_values);


    cout << fixed << setprecision(precision);
    cout << "\nApproximate solution at r = " << n * h + 0.1  << ": " << result <<
endl;

    for (int i = 0; i < n; ++i) {
        cout << "r[" << i << "] = " << r_values[i] << ", N[" << i << "] = " <<
N_values[i] << endl;
    }

    delete[] r_values;
    delete[] N_values;
    return 0;
}
```



```
4iteration
K1: -0.0186198
K2: -0.0250058
K3: -0.0235867
K4: -0.0294612
K: -0.0161122

N4: 0.239695
5iteration
K1: -0.0292114
K2: -0.033972
K3: -0.0331064
K4: -0.0377518
K: -0.0223403

N5: 0.206175
6iteration
K1: -0.037614
K2: -0.0417543
K3: -0.0411174
K4: -0.0452546
K: -0.0276234

N6: 0.16474
7iteration
K1: -0.0451638
K2: -0.0490199
K3: -0.0485058
K4: -0.0523919
K: -0.0325136

N7: 0.115972
Approximate solution at r = 0.8: 0.115972
```





```
r[0] = 0.8000, N[0] = -0.6581
r[1] = 1.5000, N[1] = -0.6541
r[2] = 2.2000, N[2] = -1.3914
r[3] = 2.9000, N[3] = -2.5344
r[4] = 3.6000, N[4] = -4.0286
r[5] = 4.3000, N[5] = -5.8594
r[6] = 5.0000, N[6] = -8.0218
```

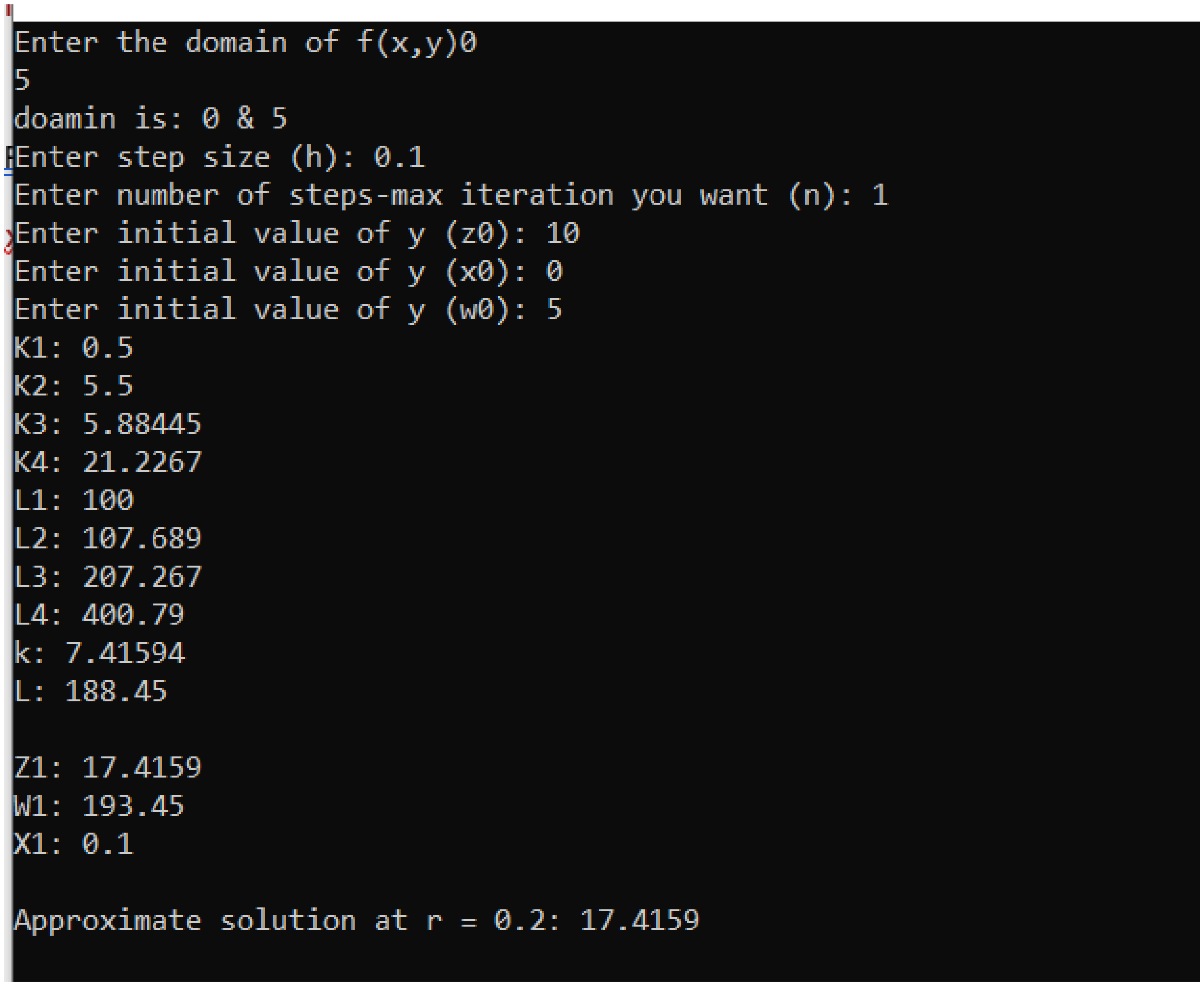
Q3

SYSTEM OF RK

```
#include <iostream>
#include <cmath>
#include <iomanip>
using namespace std;
// Defining or setting f(x,z,w)
double f1(double r, double N,double w)
{
    return (w);
}

double f2(double r, double z, double w)
{
    return (pow(z,3));
}


double RK(double h, int n, double z0, double x0, double w0)
{
    double Z = z0;
    double X = x0;
    double W = w0;
    int count = 1;

    for (int i = 0; i < n; i++)
    {
        double K1 = h * f1(X, Z, W);
        double L1 = h * f2(X, Z, W);
        double K2 = h * f1(X + 0.5 * h, Z + 0.5 * K1, W + 0.5 * L1);
        double L2 = h * f2(X + 0.5 * h, Z + 0.5 * K1, W + 0.5 * L1);
        double K3 = h * f1(X + 0.5 * h, Z + 0.5 * K2, W + 0.5 * L2);
        double L3 = h * f2(X + 0.5 * h, Z + 0.5 * K2, W + 0.5 * L2);
        double K4 = h * f1(X + h, Z + K3, W + L3);
        double L4 = h * f2(X + h, Z + K3, W + L3);
```

```cpp
        cout << "K1: " << K1 << endl;
        cout << "K2: " << K2 << endl;
        cout << "K3: " << K3 << endl;
        cout << "K4: " << K4 << endl;
        cout << "L1: " << L1 << endl;
        cout << "L2: " << L2 << endl;
        cout << "L3: " << L3 << endl;
        cout << "L4: " << L4 << endl;
        double k = (K1 + 2 * K2 + 2 * K3 + K4) / 6;
        double L = (L1 + 2 * L2 + 2 * L3 + L4) / 6;
        cout << "k: " << k << endl;
        cout << "L: " << L << endl;
        Z += k;
        W += L;
        X += h;
        cout << "\nZ" << count << ": " << Z;
        cout << "\nW" << count << ": " << W;
        cout << "\nX" << count << ": " << X << endl;
        count++;
    }
    return Z;
}

int main() {
    double x0, h, z0,w0;
    int n,x1,x2,percision;


    cout << "Enter the domain of f(x,y)";
    cin >> x1 >> x2;
    cout << "doamin is: " << x1 <<" & " << x2 << endl;
    cout << "Enter step size (h): ";
    cin >> h;
    cout << "Enter number of steps-max iteration you want (n): ";
    cin >> n;
    cout << "Enter initial value of y (z0): ";
    cin >> z0;
    cout << "Enter initial value of y (x0): ";
    cin >> x0;

    cout << "Enter initial value of y (w0): ";
    cin >> w0;
    cout << "Enter the number of decimal points to display: ";
    cin >> percision;

    double result = RK(h, n, z0,x0,w0);
    cout << fixed << setprecision(percision);
    cout << "\nApproximate solution at r = " << n * h + 0.1  << ": " << result <<
endl;

    return 0;
```

```
}
```



```
Enter the domain of f(x,y)0
5
doamin is: 0 & 5
Enter step size (h): 0.1
Enter number of steps-max iteration you want (n): 1
Enter initial value of y (z0): 10
Enter initial value of y (x0): 0
Enter initial value of y (w0): 5
K1: 0.5
K2: 5.5
K3: 5.88445
K4: 21.2267
L1: 100
L2: 107.689
L3: 207.267
L4: 400.79
k: 7.41594
L: 188.45

Z1: 17.4159
W1: 193.45
X1: 0.1

Approximate solution at r = 0.2: 17.4159
```