Name: Jahirul Islam ID : C193083

Semester : 7th Section : 7BM

Course Code : CSE 4746

Assignment 3

1

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

|  |  |  |
| --- | --- | --- |
| #define | fast | ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0); |
| #define | ll | long long |
| #define | ull | unsigned long long |
| #define | ld | long double |

int main()

{

fast; int n;

cin >> n;

int x[n+5],y,del\_y[n+5][n+5]; for (int i = 0; i < n; ++i)

{

cin >> x[i];

}

for (int i = 0; i < n; ++i)

{

cin >> del\_y[0][i];

}

long double X,h,first\_derivative,second\_derivative; cin >> X;

for (int i = 1; i < n; ++i)

{

for (int j = 0; j < n-i; ++j)

{

del\_y[i][j] = del\_y[i-1][j+1] - del\_y[i-1][j];

}

}

h = x[1] - x[0];

long double cal=0,d; for (int i = 1; i < n; ++i)

{

if(i%2==0)

{

cal-=(del\_y[i][0]/i);

}

else

{

cal+=(del\_y[i][0]/i);

}

}

first\_derivative = cal / h;

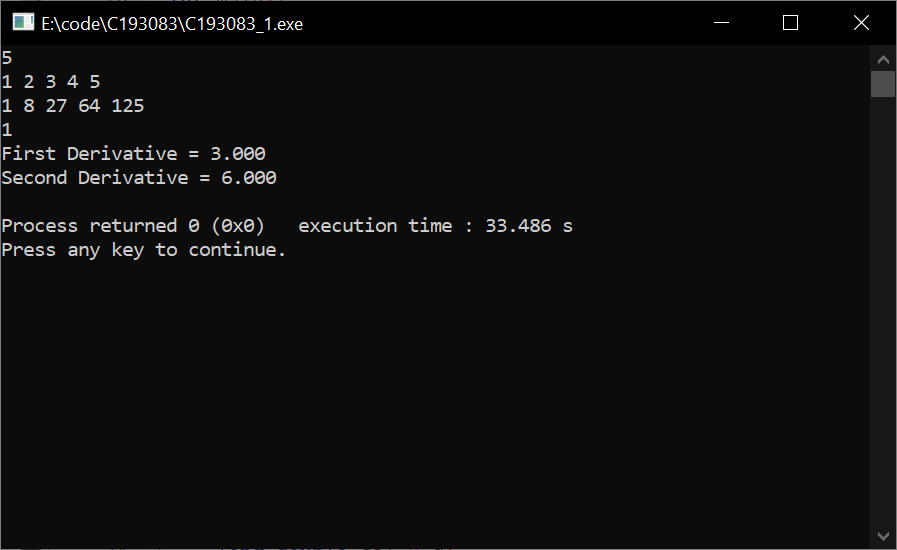
cal = del\_y[2][0]-del\_y[3][0]+(11\*del\_y[4][0]/12); second\_derivative = cal / (h\*h);

cout << "First Derivative = " << setprecision(3) << fixed << first\_derivative << "\n";

cout << "Second Derivative = " << setprecision(3) << fixed

<< second\_derivative << "\n";

return 0;

}

2

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

|  |  |  |
| --- | --- | --- |
| #define | fast | ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0); |
| #define | ll | long long |
| #define | ull | unsigned long long |
| #define | ld | long double |

int main()

{

fast; int n;

cin >> n;

int x[n+5],y,del\_y[n+5][n+5]; for (int i = 0; i < n; ++i)

{

cin >> x[i];

}

for (int i = 0; i < n; ++i)

{

cin >> del\_y[0][i];

}

for (int i = 1; i < n; ++i)

{

for (int j = 0; j < n-i; ++j)

{

del\_y[i][j] = del\_y[i-1][j+1] - del\_y[i-1][j];

}

}

long double X,h,first\_derivative,second\_derivative,u; cin >> X;

h = x[1] - x[0];

u = (X-x[0])/h;

long double cal=0,d;

cal = del\_y[1][0] + (2 \* u - 1) \* (del\_y[2][0] / 2) + (3 \* u \* u - 6

\* u + 2) \* (del\_y[3][0] / 6) + (4 \* u \* u \* u - 18 \* u \* u + 22 \* u - 6) \* (del\_y[4][0] / 24);

first\_derivative = cal / h;

cal = del\_y[2][0] + (6 \* u - 6) \* (del\_y[3][0] / 6) + (12 \* u \* u - 36 \* u + 22) \* (del\_y[4][0] / 24);

second\_derivative = cal / (h\*h);

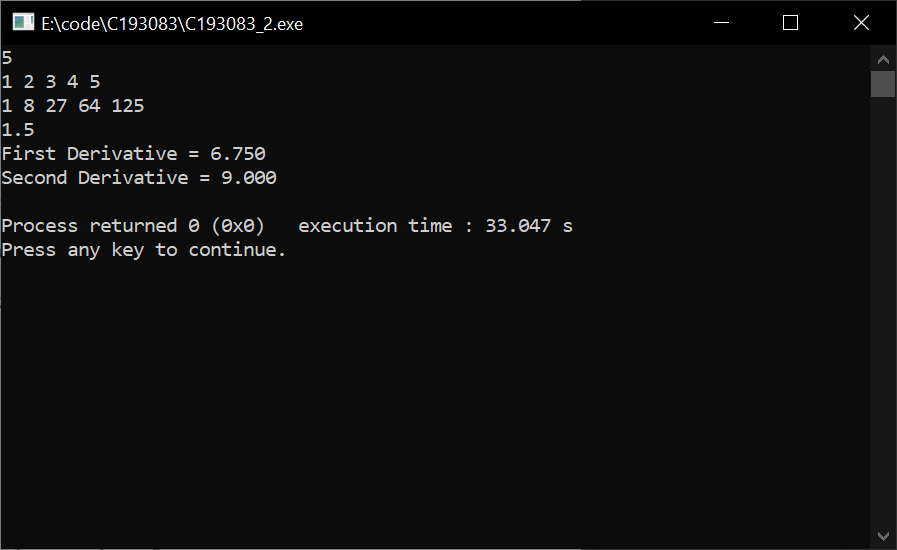
cout << "First Derivative = " << setprecision(3) << fixed << first\_derivative << "\n";

cout << "Second Derivative = " << setprecision(3) << fixed

<< second\_derivative << "\n";

return 0;

}



3

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

typedef long long ll;

typedef unsigned long long ull; typedef long double ld; #define EPS 1e-6

#define fraction() cout.unsetf(ios::floatfield); cout.precision(3); cout.setf(ios::fixed,ios::floatfield);

int main()

{

fraction(); int n = 3; ld a[n][n]; ld b[n];

ld x[n];

ld x\_new[n]; int i, j, k;

ld sum, error;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cin >> a[i][j];

}

cin >> b[i];

}

for (i = 0; i < n; i++)

{

x[i] = 0.0;

}

for (k = 1; k <= 10; k++)

{

int key = 0; error = 0.0;

for (i = 0; i < n; i++)

{

sum = b[i]; key = 0;

for (j = 0; j < n; j++)

{

if (i != j)

{

sum -= a[i][j] \* x[j];

}

}

x\_new[i] = sum / a[i][i];

error = fabs(x\_new[i] - x[i]); if (error > EPS)

{

key = 1;

}

x[i] = x\_new[i];

}

if (key == 0)

{

break;

}

}

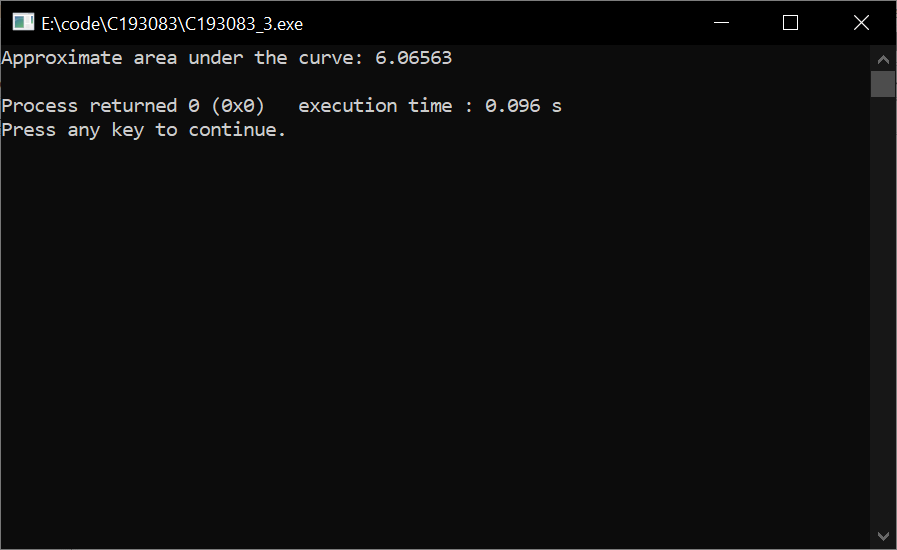
for (i = 0; i < n; i++)

{

cout << "x" << i + 1 << " = " << x[i] << endl;

}

return 0;

}

4

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

double f(double x) { return exp(sin(x));

}

double simpson(double a, double b, int n) { double h = (b - a) / n;

double sum = 0.0;

for (int i = 0; i <= n; i++) { double x = a + i \* h;

if (i == 0 || i == n) { sum += f(x);

} else if (i % 2 == 1) { sum += 4 \* f(x);

} else {

sum += 2 \* f(x);

}

}

return (h / 3.0) \* sum;

}

int main() {

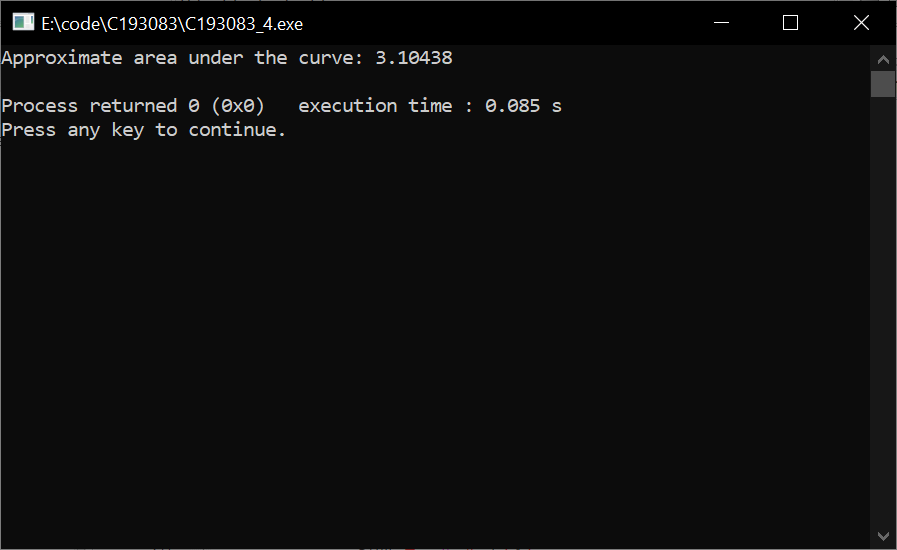
double a = 0.0; // lower limit of integration double b = M\_PI/2; // upper limit of integration int n = 10; // number of intervals

double area = simpson(a, b, n);

cout << "Approximate area under the curve: " << area << endl;

return 0;

}



5

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

typedef long long ll;

typedef unsigned long long ull; typedef long double ld;

#define EPS 1e-6

#define fraction() cout.unsetf(ios::floatfield); cout.precision(3); cout.setf(ios::fixed,ios::floatfield);

int main()

{

fraction(); int n = 3; ld a[n][n]; ld b[n];

ld x[n];

ld x\_new[n]; int i, j, k;

ld sum, error;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cin >> a[i][j];

}

cin >> b[i];

}

for (i = 0; i < n; i++)

{

x[i] = 0.0;

}

for (k = 1; k <= 9; k++)

{

for (i = 0; i < n; i++)

{

sum = b[i];

for (j = 0; j < n; j++)

{

if (i != j)

{

sum -= a[i][j] \* x[j];

}

}

x\_new[i] = sum / a[i][i];

}

error = 0.0;

for (i = 0; i < n; i++)

{

error += fabs(x\_new[i] - x[i]);

}

for (i = 0; i < n; i++)

{

x[i] = x\_new[i];

}

if (error < EPS)

{

break;

}

}

for (i = 0; i < n; i++)

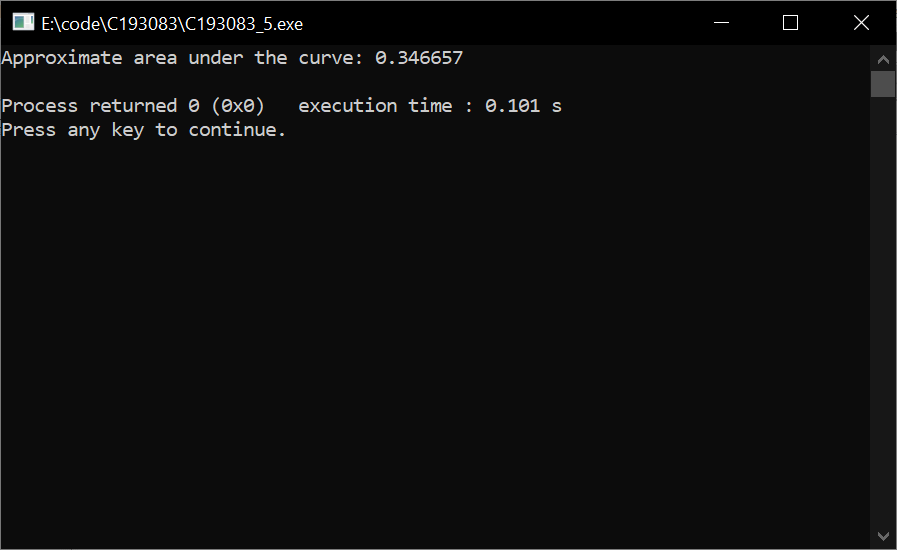
{

cout << "x" << i + 1 << " = " << x[i] << endl;

}

return 0;

}



6

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

int main() {

int mat[3][3], i, j; int det = 0;

cout << "Enter the elements of the matrix:\n"; for(i=0; i<3; i++) {

for(j=0; j<3; j++) {

cin >> mat[i][j];

}

}

// calculate the determinant

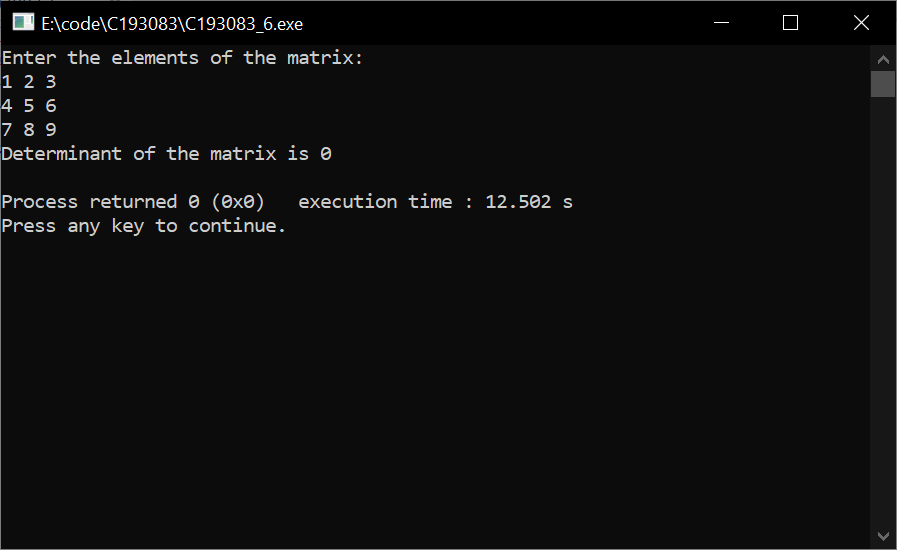
det = mat[0][0]\*(mat[1][1]\*mat[2][2] - mat[1][2]\*mat[2][1]) -

mat[0][1]\*(mat[1][0]\*mat[2][2] - mat[1][2]\*mat[2][0]) +

mat[0][2]\*(mat[1][0]\*mat[2][1] - mat[1][1]\*mat[2][0]); cout << "Determinant of the matrix is " << det << endl;

return 0;

}



7

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

int main()

{

float a[3][3] = {{1,1,1},{1,2,3},{1,3,4}};

float b[3][1] = {{1},{6},{6}};

float inv[3][3], det = 0;

// Finding the determinant of the matrix for(int i=0; i<3; i++)

{

det = det + (a[0][i]\*(a[1][(i+1)%3]\*a[2][(i+2)%3] -

a[1][(i+2)%3]\*a[2][(i+1)%3]));

}

// Finding the inverse of the matrix for(int i=0; i<3; i++)

{

for(int j=0; j<3; j++)

{

inv[i][j] = ((a[(j+1)%3][(i+1)%3] \* a[(j+2)%3][(i+2)%3]) -

(a[(j+1)%3][(i+2)%3]\*a[(j+2)%3][(i+1)%3]))/det;

}

}

// Multiplying inverse of matrix with the constant matrix float x[3][1];

for(int i=0; i<3; i++)

{

for(int j=0; j<1; j++)

{

x[i][j] = 0;

for(int k=0; k<3; k++)

{

x[i][j] += inv[i][k] \* b[k][j];

}

}

}

// Displaying the solution

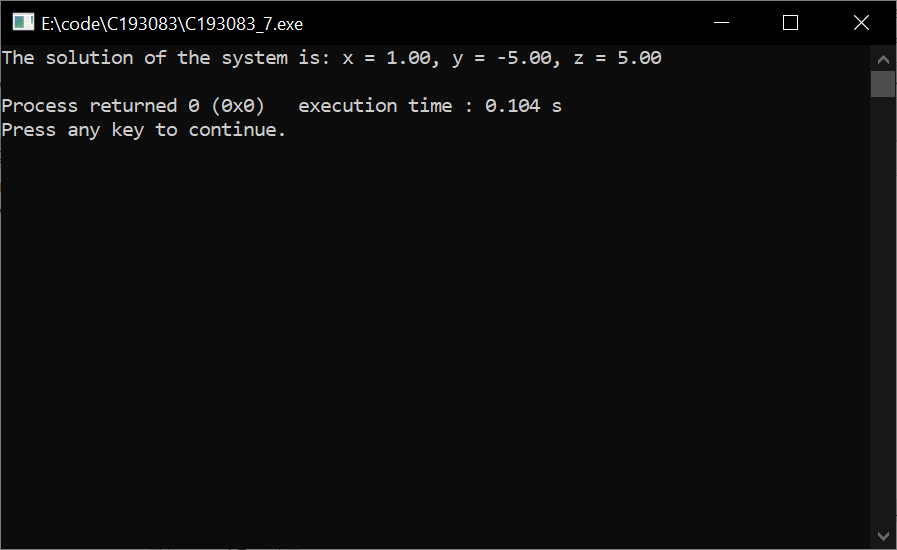
cout << "The solution of the system is: ";

cout << "x = " << fixed << setprecision(2) << x[0][0] << ", y

= " << x[1][0] << ", z = " << x[2][0] << endl;

return 0;

}



8

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

typedef long long ll;

typedef unsigned long long ull; typedef long double ld;

#define EPS 1e-9

#define fraction() cout.unsetf(ios::floatfield); cout.precision(10); cout.setf(ios::fixed,ios::floatfield);

const int mx = 10; int n, m;

ld det(int mat[10][10])

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < m; j++)

{

cout << mat[i][j] << " ";

}

cout << endl;

}

cout << endl; int det = 0.0;

for (int i = 0; i < 3; i++)

{

det = det + (mat[0][i] \* (mat[1][(i + 1) % 3] \* mat[2][(i + 2)

% 3] - mat[1][(i + 2) % 3] \* mat[2][(i + 1) % 3]));

}

return det;

}

int main()

{

fraction();

int i, j, k, x, y, z, cnt = 0; string s;

int A[mx][mx];

int B[mx][mx];

int a[mx][mx];

int b[mx][mx];

int c[mx][mx]; cin >> n >> m;

for (i = 0; i < n; i++)

{

for (j = 0; j < m; j++)

{

cin >> A[i][j]; a[i][j] = A[i][j];

b[i][j] = A[i][j];

c[i][j] = A[i][j];

}

cin >> B[0][i];

}

ld d = det(A);

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

if (j == 0)

{

a[i][j] = B[0][i];

}

if (j == 1)

{

b[i][j] = B[0][i];

}

if (j == 2)

{

c[i][j] = B[0][i];

}

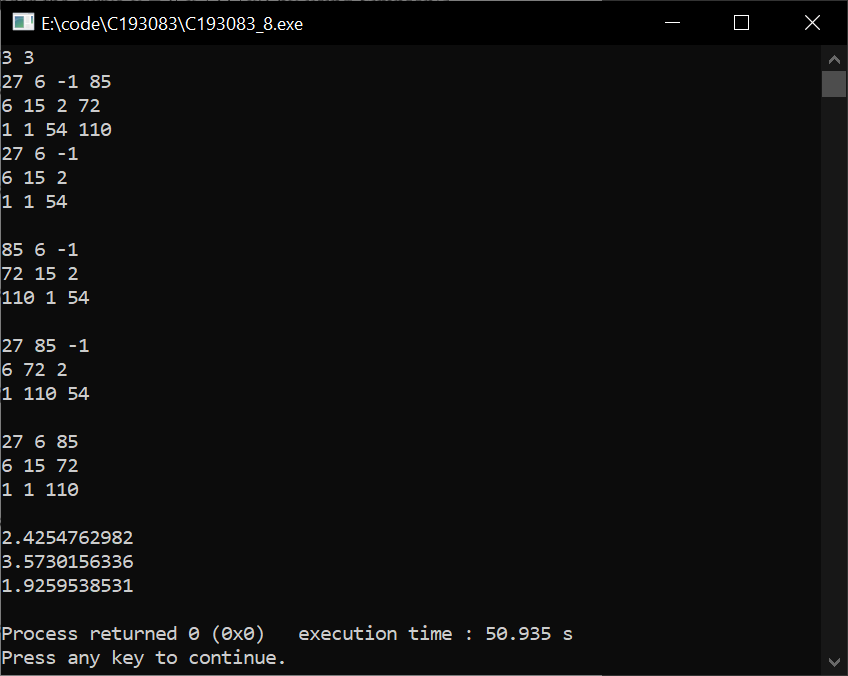
}

}

ld da = det(a); ld db = det(b); ld dc = det(c);

cout << da / d << endl; cout << db / d << endl; cout << dc / d << endl;

return 0;

}

10

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std; typedef long long ll;

typedef unsigned long long ull; typedef long double ld;

#define EPS 1e-9 int main()

{

int i, j, k , n, m, x, y, z, c = 0, cnt = 0; string s;

cin >> n; m = n + 1;

ld a[n + 10][m + 10]; for (i = 0; i < n; i++)

{

for (j = 0; j < m; j++)

{

cin >> a[i][j];

}

}

for (i = 0; i < n; i++)

{

for (j = i + 1; j < n; j++)

{

ld factor = a[j][i] / a[i][i]; for (k = i; k <= n; k++)

{

a[j][k] = a[j][k] - factor \* a[i][k];

}

}

}

for (i = n - 1; i >= 0; i--)

{

for (j = i - 1; j >= 0; j--)

{

ld factor = a[j][i] / a[i][i];

for (k = i; k <= n; k++)

{

a[j][k] = a[j][k] - factor \* a[i][k];

}

}

}

for (i = 0; i < n; i++)

{

double factor = a[i][i]; for (j = i; j <= n; j++)

{

a[i][j] = a[i][j] / factor;

}

}

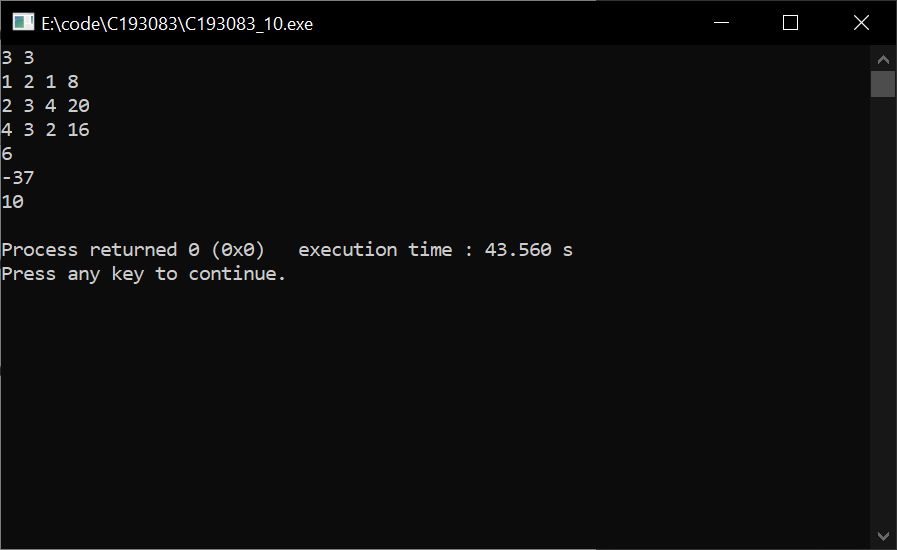
for (i = 0; i < n; i++)

{

cout << a[i][n] << endl;

}

return 0;

}

11

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

typedef long long ll;

typedef unsigned long long ull; typedef long double ld;

#define EPS 1e-6

#define fraction() cout.unsetf(ios::floatfield); cout.precision(3); cout.setf(ios::fixed,ios::floatfield);

int main()

{

fraction(); int n = 3; ld a[n][n]; ld b[n];

ld x[n];

ld x\_new[n]; int i, j, k;

ld sum, error;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cin >> a[i][j];

}

cin >> b[i];

}

for (i = 0; i < n; i++)

{

x[i] = 0.0;

}

for (k = 1; k <= 9; k++)

{

for (i = 0; i < n; i++)

{

sum = b[i];

for (j = 0; j < n; j++)

{

if (i != j)

{

sum -= a[i][j] \* x[j];

}

}

x\_new[i] = sum / a[i][i];

}

error = 0.0;

for (i = 0; i < n; i++)

{

error += fabs(x\_new[i] - x[i]);

}

for (i = 0; i < n; i++)

{

x[i] = x\_new[i];

}

if (error < EPS)

{

break;

}

}

for (i = 0; i < n; i++)

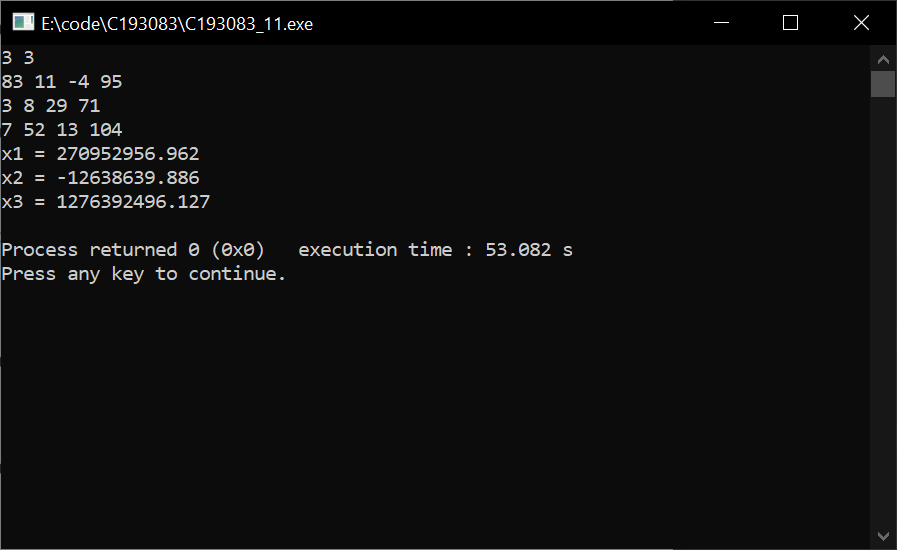
{

cout << "x" << i + 1 << " = " << x[i] << endl;

}

return 0;

}



12

========================

//Jahirul Islam

//Lab3 #include<bits/stdc++.h> using namespace std;

typedef long long ll;

typedef unsigned long long ull; typedef long double ld; #define EPS 1e-6

#define fraction() cout.unsetf(ios::floatfield); cout.precision(3); cout.setf(ios::fixed,ios::floatfield);

int main()

{

fraction(); int n = 3; ld a[n][n]; ld b[n];

ld x[n];

ld x\_new[n]; int i, j, k;

ld sum, error;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cin >> a[i][j];

}

cin >> b[i];

}

for (i = 0; i < n; i++)

{

x[i] = 0.0;

}

for (k = 1; k <= 10; k++)

{

int key = 0; error = 0.0;

for (i = 0; i < n; i++)

{

sum = b[i]; key = 0;

for (j = 0; j < n; j++)

{

if (i != j)

{

sum -= a[i][j] \* x[j];

}

}

x\_new[i] = sum / a[i][i];

error = fabs(x\_new[i] - x[i]); if (error > EPS)

{

key = 1;

}

x[i] = x\_new[i];

}

if (key == 0)

{

break;

}

}

for (i = 0; i < n; i++)

{

cout << "x" << i + 1 << " = " << x[i] << endl;

}

return 0;

}

