**Ministry of education and science of Republic of Kazakhstan**

**Kazakh National University named after al-Farabi**



**Faculcy**: “Mechanics and Mathematis”

**Department**: “Mathematical and Computer Modelling”

**Report-4,5**

**Done by:** Kakibay A

**Checked by:** Zhandaulet E.

**Almaty**

**2020**

1. **1D Burgers equation**

#include<iostream>

#include<fstream>

#include<cmath>

using namespace std;

int main() {

int n=101, iter = 0;

double dx = 1.0 / (n - 1), dt = 0.01, eps=0.00001, dif, Re=15.0;

double \*u0 = new double[n];

double \*alfa = new double[n];

double \*betta = new double[n];

double \*A = new double[n];

double \*B= new double[n];

double \*D = new double[n];

double \*u = new double[n];

double \*C=new double [n];

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

u0[i] = 0.0;

do { u0[0]=2.0;

u0[n-1]=0.0;

u[0] = 2.0;

u[n-1]=0.0;

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

A[i] = u0[i] / dx - 1.0 / (Re \* dx \* dx);

B[i] = 1 / dt + 2.0 / (Re \* dx \* dx)- u0[i] / dx;

C[i] = -1.0 / (Re \* dx \* dx);

}

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

D[i] = u0[i] / dt;

alfa[1] = 0.0;

betta[1] = 2.0;

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

alfa[i + 1] = -A[i] / (B[i] + C[i] \* alfa[i]);

betta[i + 1] = (D[i] - C[i] \* betta[i]) / (B[i] + C[i] \* alfa[i]);

}

for (int i = n - 2; i >= 0; i--){

u[i] = alfa[i + 1] \* u[i + 1] + betta[i + 1];

}

dif = 0.0;

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

if ( dif < abs(u[i] - u0[i]))

dif = abs(u[i] - u0[i]);

} if (iter % 40 == 0) {

cout << "difference " << iter << " = " << dif << endl;

}

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

u0[i] = u[i];

iter++;

}

while (dif > eps);

//while (iter <1000);

fstream fout("task4.dat", ios::out);

fout << "VARIABLES=\"X\",\"U\"" << endl;

fout << "ZONE I= "<< n<< ",F=POINT" << endl;

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

fout << i \* dx << "\t" << u[i] << endl;

}

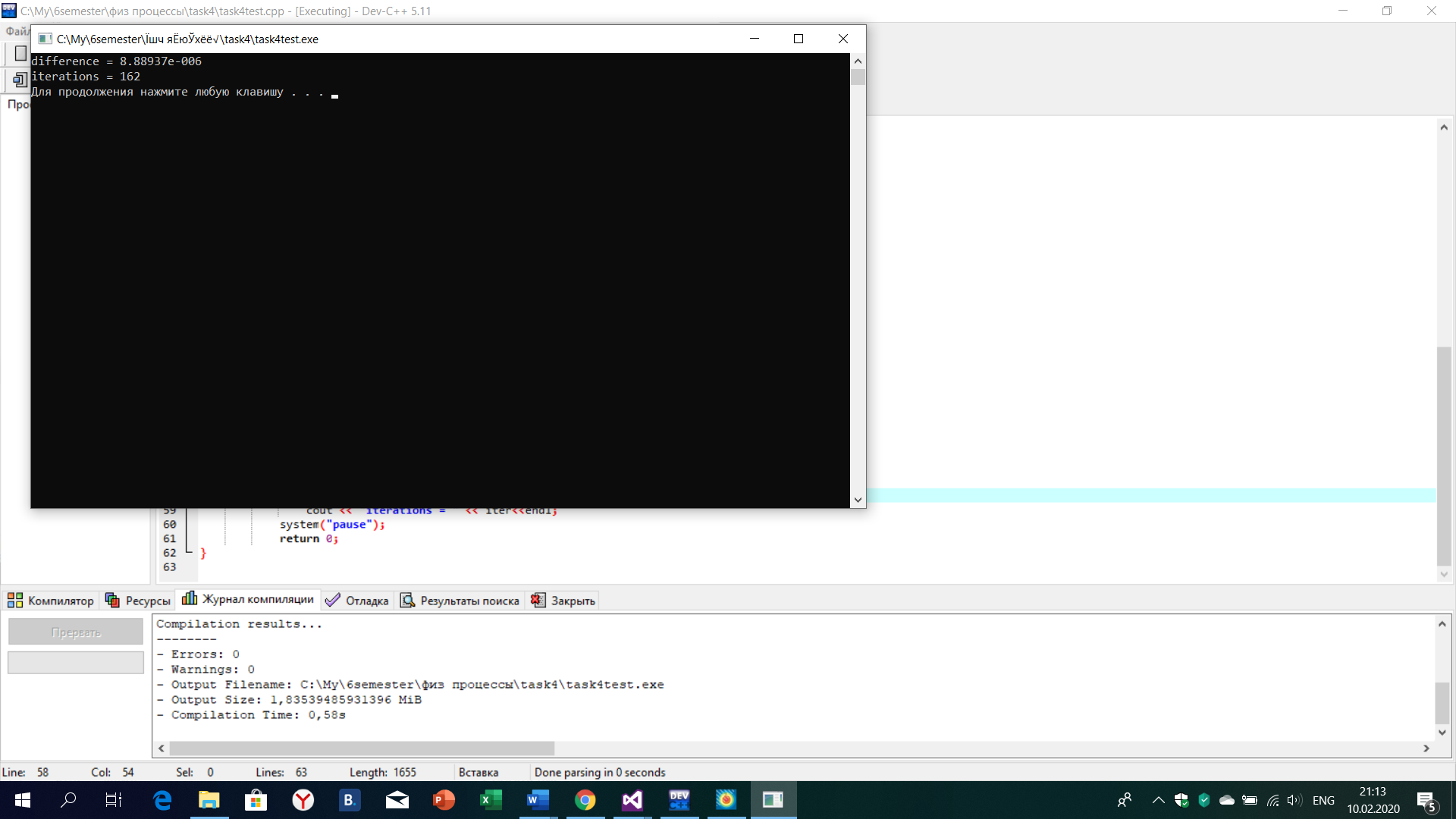
cout << "difference = " << dif<<endl;

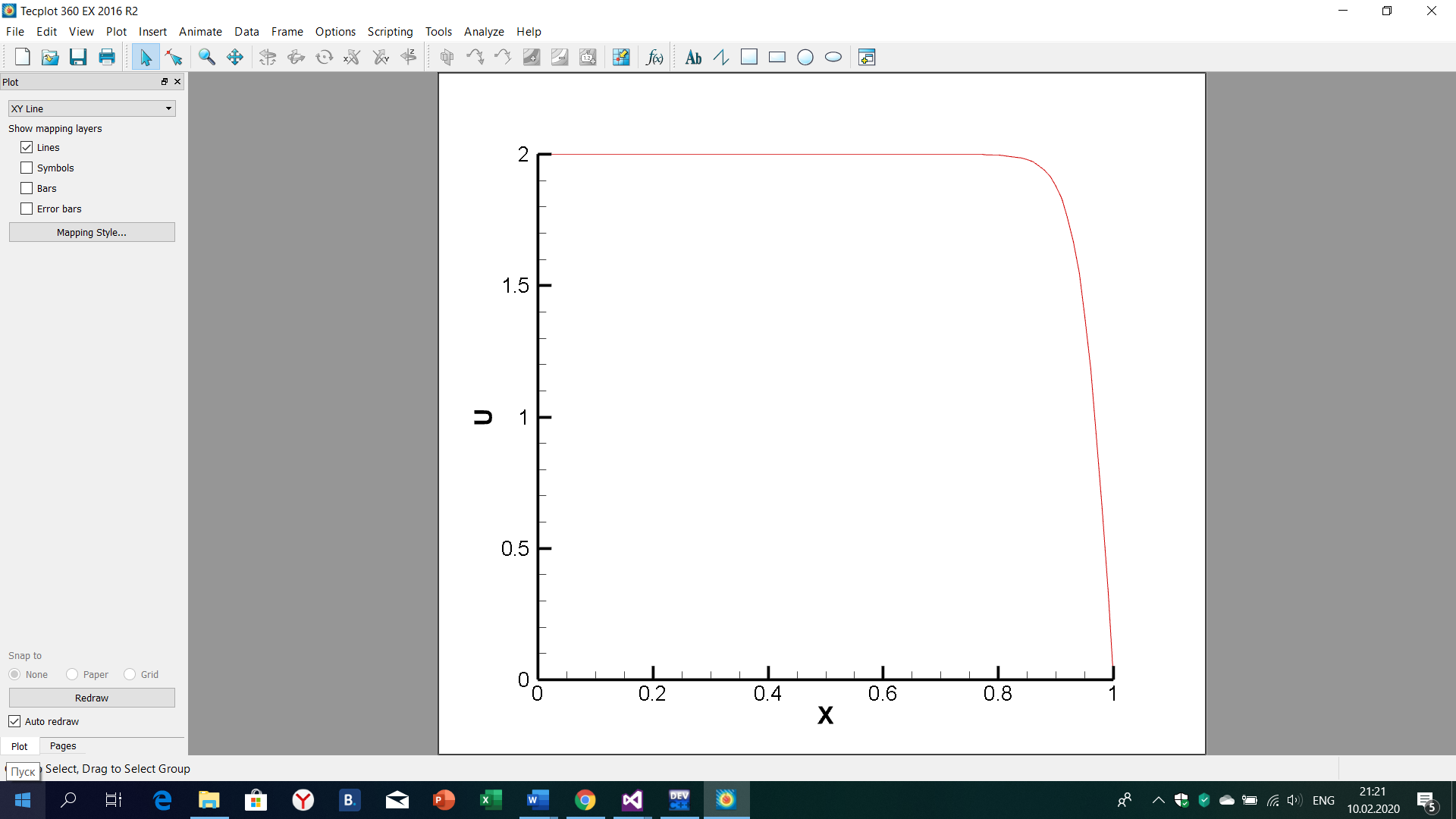
cout << "iterations = " << iter<<endl;

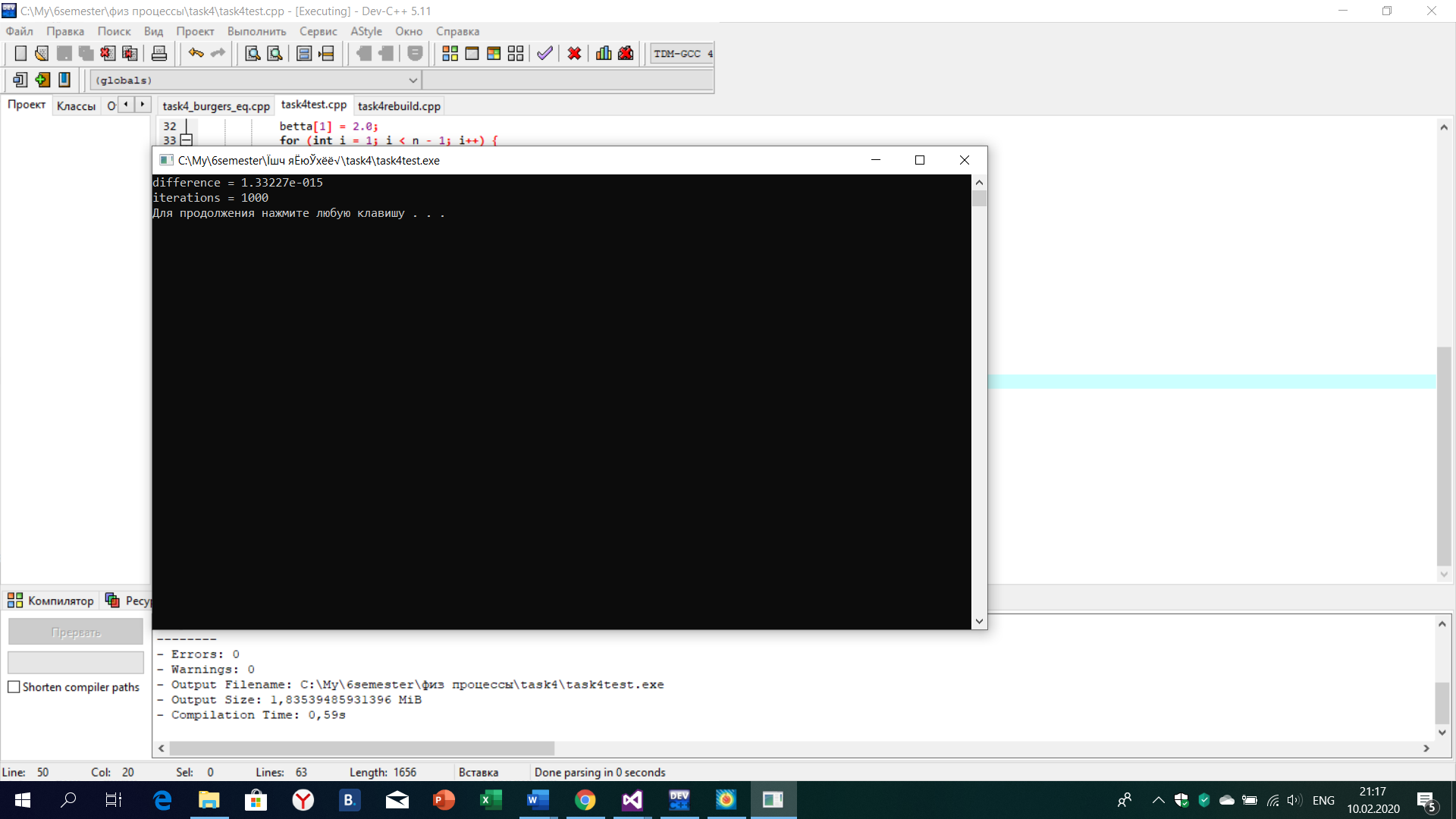
system("pause");

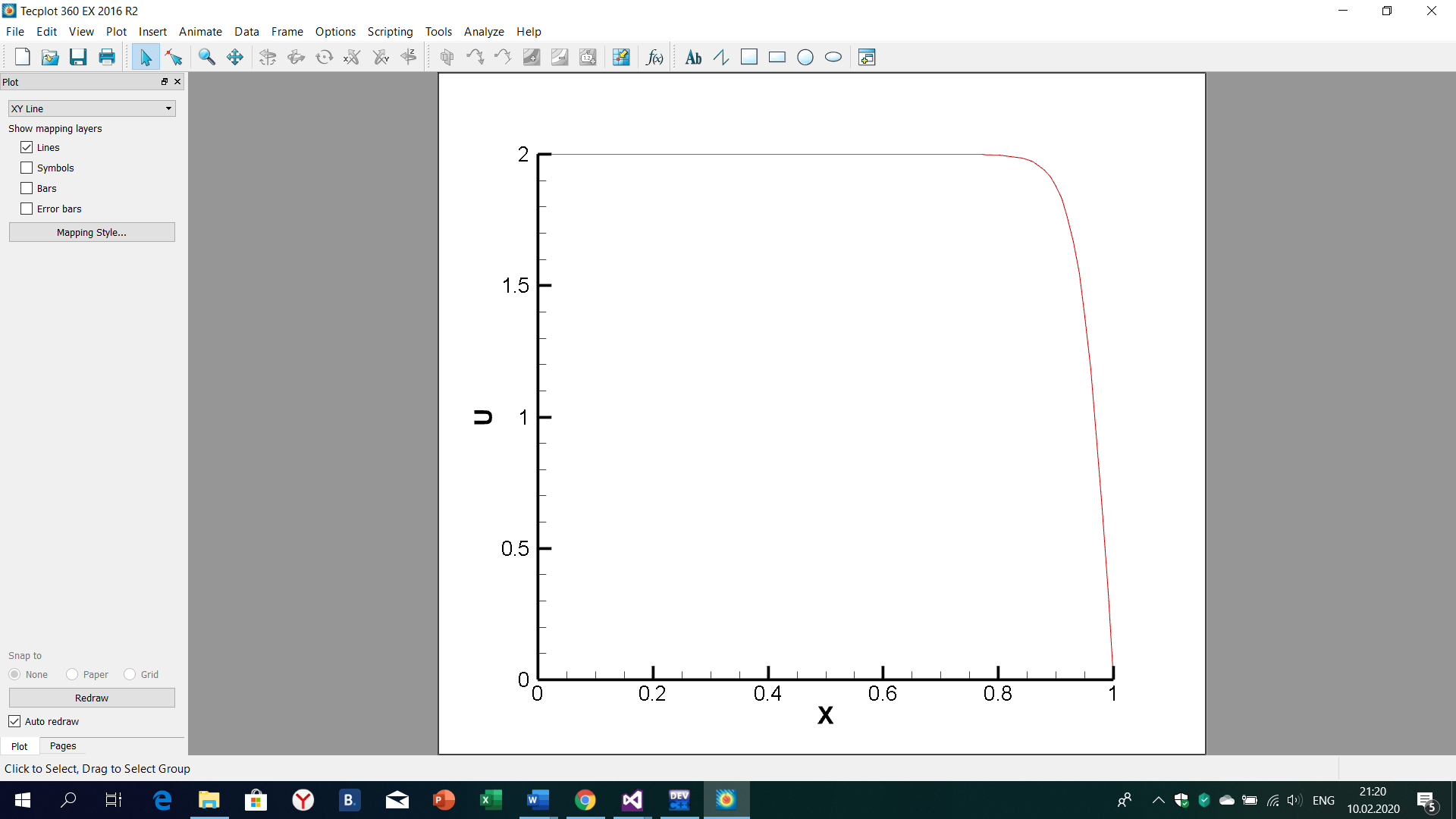
return 0;

}









if (iter % 40 == 0) {

fstream fout("out111.dat", ios::out);

fout << "VARIABLES=\"X\",\"U\"" << endl;

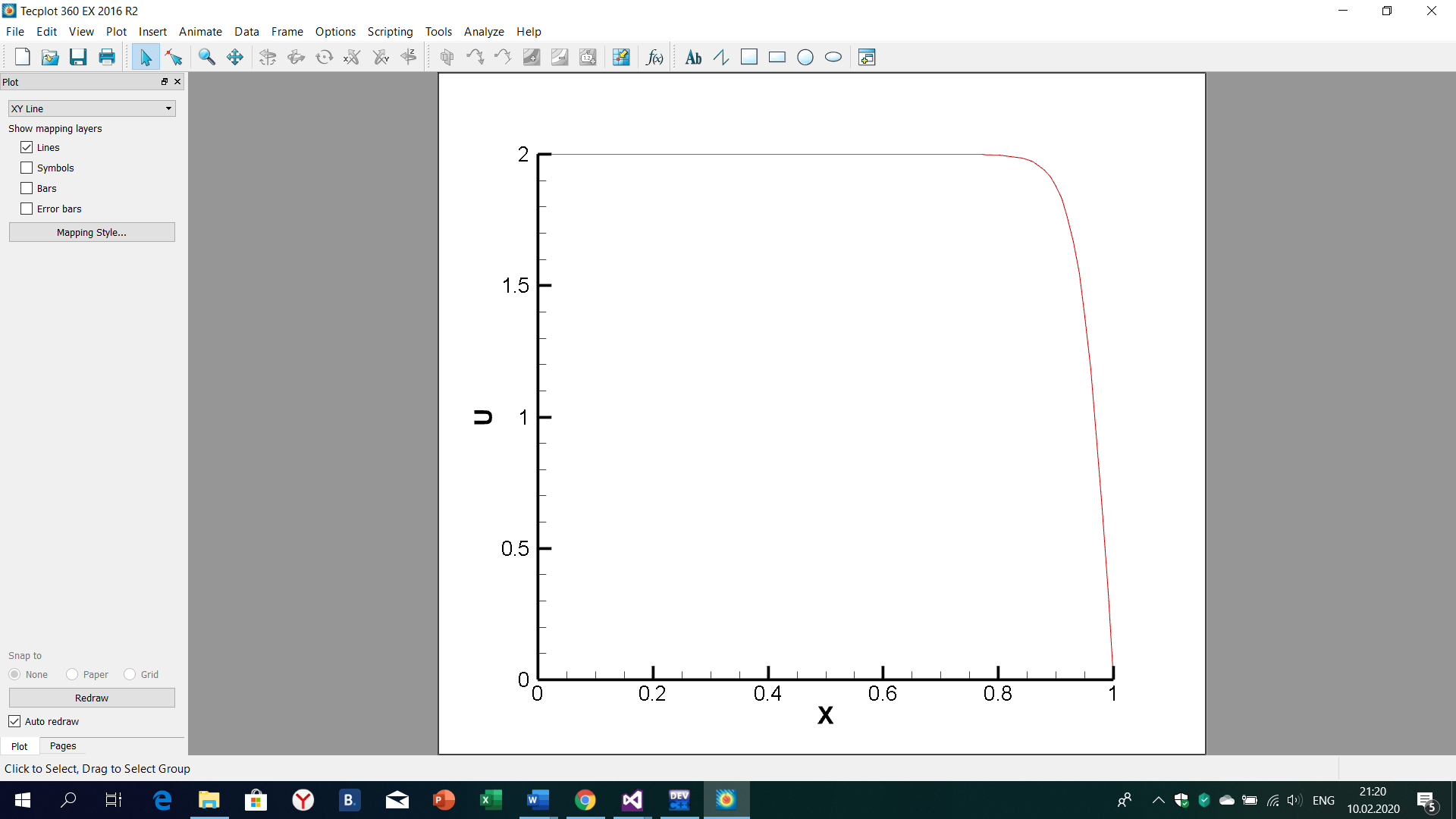
fout << "Zone T=" << "\"" << iter << "\"" << ", I=" << n << ", F=POINT" << endl;

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

fout << i \* dx << "\t" << u[i]<< "\t" << endl;

} }

}



2**. 2D heat conductivity**

#include <iostream>

#include <fstream>

#include <cmath>

using namespace std;

int main() {

int n = 101, iter = 0;

double dx = 1.0 / (n - 1),dy = 1.0 / (n - 1), dt, dif = 0.0, eps = pow(10,-6);

double u0 [n] [n], u[n][n];

dt = 0.5 / (1.0 / (dx \* dx) + 1.0 / (dy \* dy));

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

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

u0[i][j] = 0.0;

u[i][j] = 0.0;

}

}

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

{

u0[0] [i] = 1.0;

u[0][i] = 1.0;

}

do {

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

for (int j = 1; j < n - 1; j++)

u[i][j] = u0[i][j] + dt \* ((u0[i + 1][j] - 2. \* u0[i][j] + u0[i - 1][j]) / (dx \* dx) +

(u0[i][j + 1] - 2. \* u0[i][j] + u0[i][j - 1]) / (dy \* dy));

dif = 0.0;

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

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

if (dif< abs(u[i][j] - u0[i][j])) {

dif = abs(u[i][j] - u0[i][j]);

}

}

}

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

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

u0[i][j] = u[i][j];

}

}

iter++;

} while (dif > eps);

ofstream fout("task5.dat");

fout << "VARIABLES = \"X\",\"Y\",\"u\"" << endl;

fout << "ZONE I=" << n << ",J=" << n<< ",F=POINT" << endl;

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

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

fout << i \* dx << '\t' << j \* dy << '\t' << u[i][j] << endl;

}

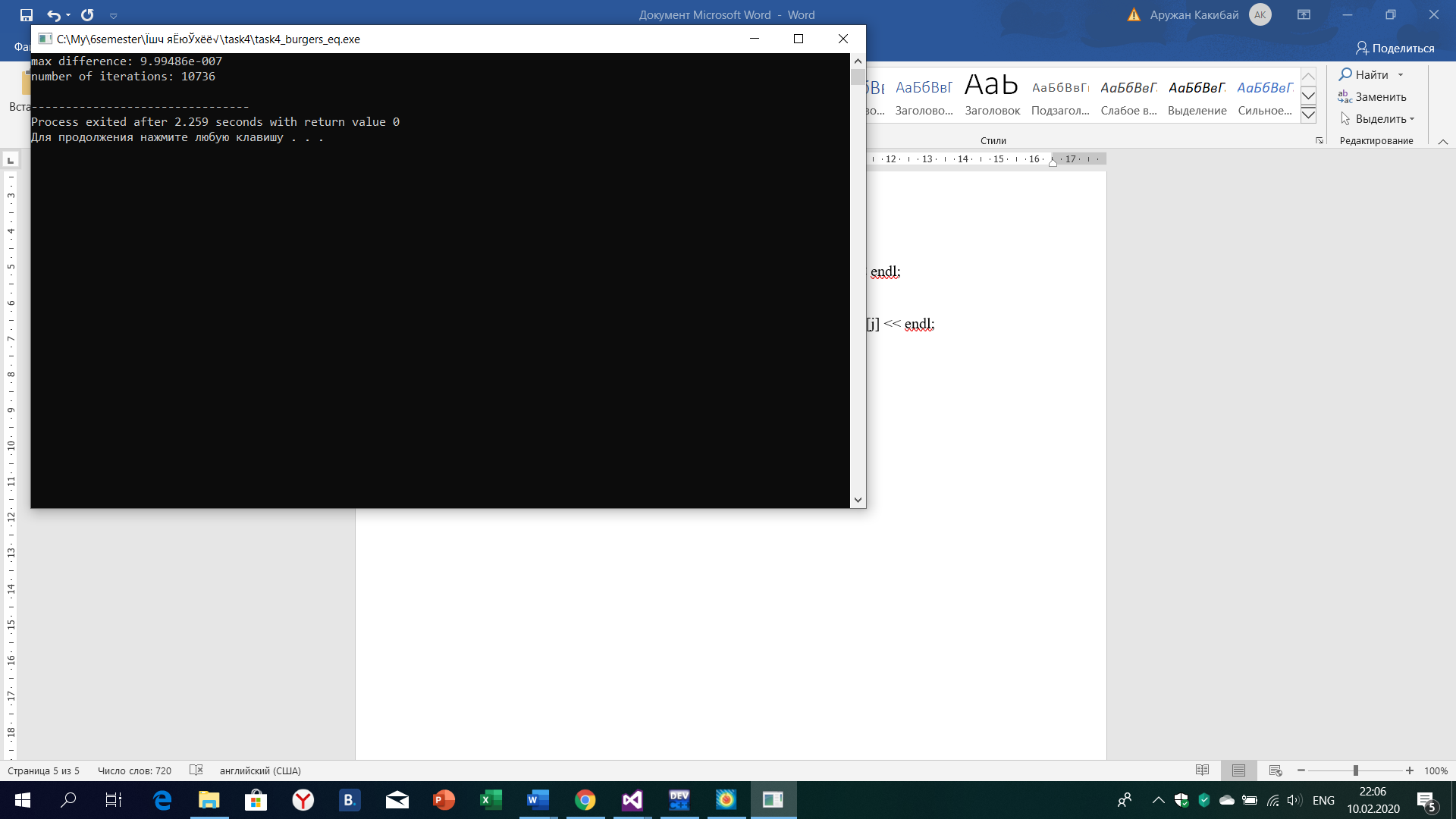
}

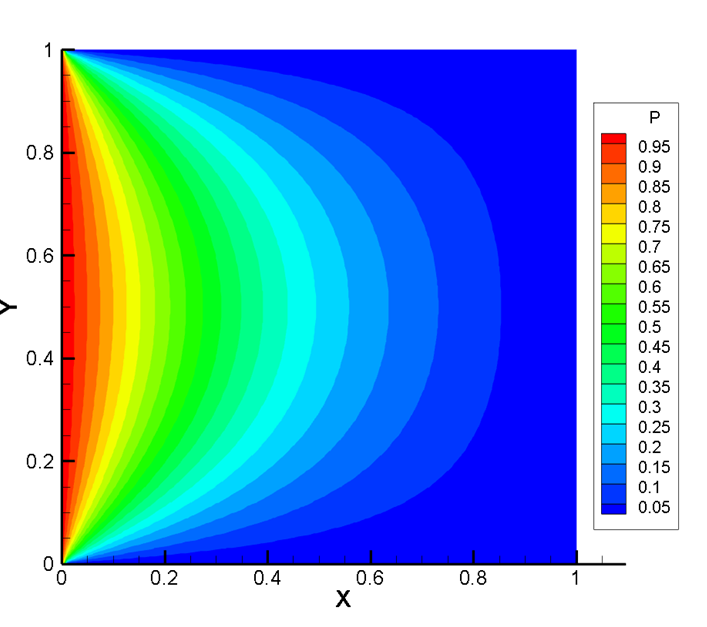
cout << "max difference: " << dif << endl;

cout << "number of iterations: " << iter << endl;

return 0;

}





Conclusion

We considered one-dimensional Burgers equation, and have solved by sweep method. Our main task was to take right A, B, C, D, also we used forward step for approximation. Second task was an explicit method for a two-dimensional parabolic equation. We needed to take satisfying step sizes of dx and dy.