**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-6**

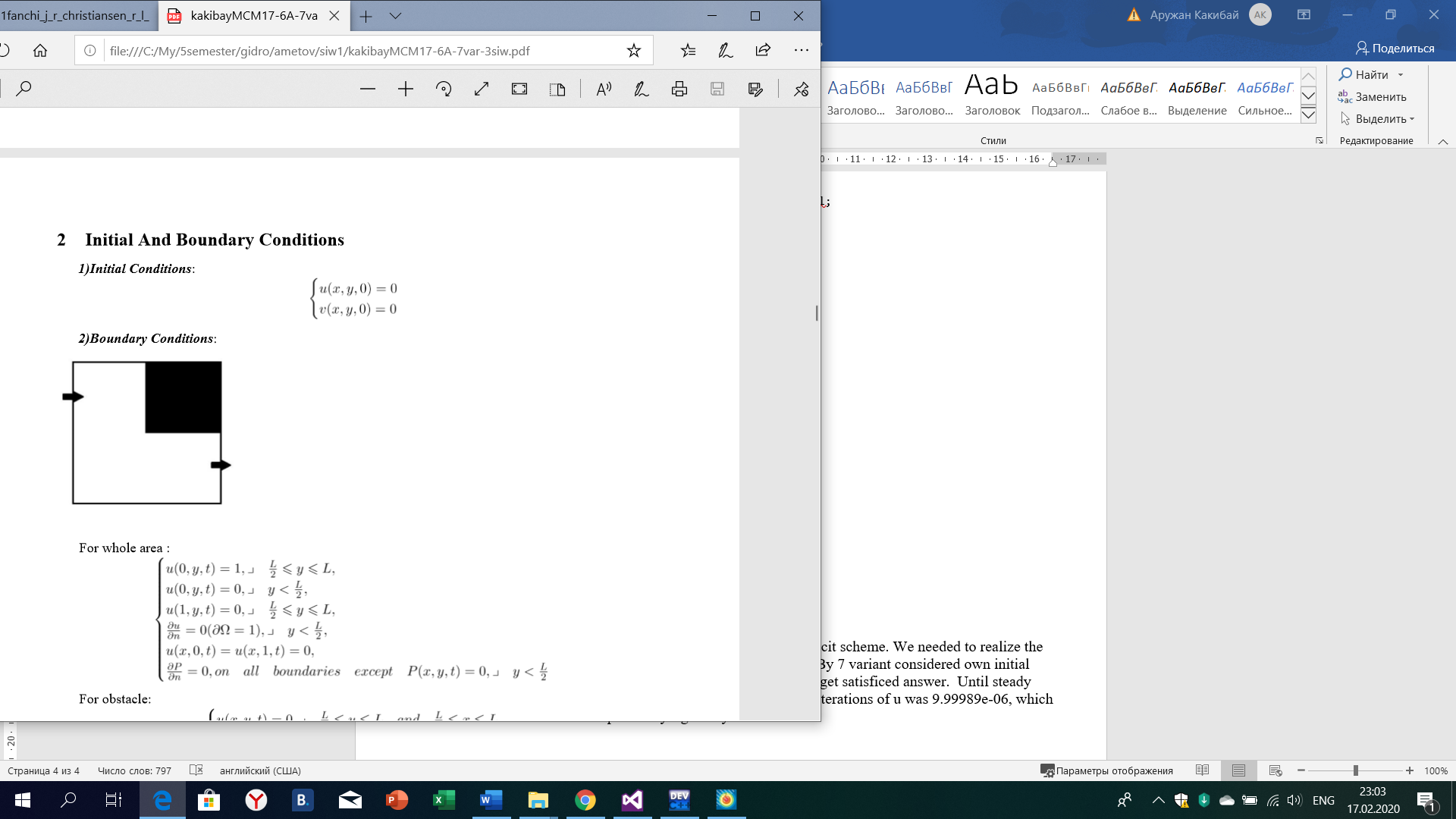
**Done by:** Kakibay A

**Checked by:** Zhandaulet E.

**Almaty**

**2020**

**2D Burgers equation**



#include<iostream>

#include<fstream>

#include<cmath>

using namespace std;

int main() {

int const n = 31;

int iter = 0;

double dx = 1.0 / (n - 1), dy = 1.0 / (n - 1), dt, dif = 0.0, dif1 = 0.0, eps = pow(10, -5), Re = 15;

double \*\*u0 = new double \*[n];

double \*\*u = new double \*[n];

double \*\*v = new double \*[n];

double \*\*v0 = new double \*[n];

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

u[i] = new double[n];

u0[i] = new double[n];

v[i] = new double[n];

v0[i] = new double[n];

}

dt = 0.00001;

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

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

u0[i][j] = 0.0;

u[i][j] = 0.0;

v[i][j] = 0.0;

v0[i][j] = 0.0;

}

}

for (int i = 0; i <= n - 1; i++) {

u[i][0] = 0.0;

v[i][0] = 0.0;

u[i][n - 1] = 0.0;

v[i][n - 1] = 0.0;

u[0][i] = 0.0;

v[0][i] = 0.0;

u[n - 1][i] = 0.0;

v[n - 1][i] = 0.0;

/\*u0[i][0] = 0.0;

v0[i][0] = 0.0;

u0[i][n - 1] = 0.0;

v0[i][n - 1] = 0.0;

u0[0][i] = 0.0;

v0[0][i] = 0.0;

u0[n - 1][i] = 0.0;

v0[n - 1][i] = 0.0;\*/

}

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

if (i<(n - 1) / 2.0) {

u[n - 1][i] = u[n - 2][i];

v[n - 1][i] = v[n - 2][i];

}

else if ((i >= (n - 1) / 2.0) && (i < (n - 1))) {

u0[0][i] = 1.0;

v0[0][i] = 0.0;

v[0][i] = 0.0;

u[0][i] = 1.0;

}

}

do {

for (int j = n / 2; j < (n - 1); j++) {

v0[0][j] = 0.0;//-1

u0[0][j] = 1.0;

}

for (int j = n / 2; j < (n - 1); j++) {//input

u[0][j] = 1.0;

v[0][j] = 0.0;

}

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

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

if ((j >= (n - 1) / 2.0) && (j <= (n - 1)) && (i >= (n - 1) / 2.0) && (i <= (n - 1))) {

v[i][j] = 0.0;

u[i][j] = 0.0;

}

else {

u[i][j] = u0[i][j] + dt \* (-u0[i][j] \* (u0[i + 1][j] - u0[i][j]) / (dx)-v0[i][j] \* (u0[i][j + 1] - u0[i][j]) / (dx)+

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

v[i][j] = v0[i][j] + dt \* (-u0[i][j] \* (v0[i + 1][j] - v0[i][j]) / (dx)-v0[i][j] \* (v0[i][j + 1] - v0[i][j]) / (dx)+

((v0[i][j + 1] - 2.0 \* v0[i][j] + v0[i][j - 1]) / (dx\*dx) + (v0[i + 1][j] - 2.0 \* v0[i][j] + v0[i - 1][j]) / (dx\*dx)) / Re);

}

}

}

for (int j = 1; j < n / 2; j++) {//output

u[n - 1][j] = u[n - 2][j];

v[n - 1][j] = v[n - 2][j];

}

dif = -100.0;

dif1 =-100.0;

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

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

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

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

}

if (dif1< fabs(v[i][j] - v0[i][j])) {

dif1 = fabs(v[i][j] - v0[i][j]);

}

}

}

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

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

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

v0[i][j] = v[i][j];

}

}

iter++;

} while (dif > eps || dif1>eps);

ofstream fout("task6.dat");

fout << "VARIABLES = \"X\",\"Y\",\"u\", \"v\"" << 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' << u0[i][j] << '\t' << v0[i][j] << '\t' << endl;

}

}

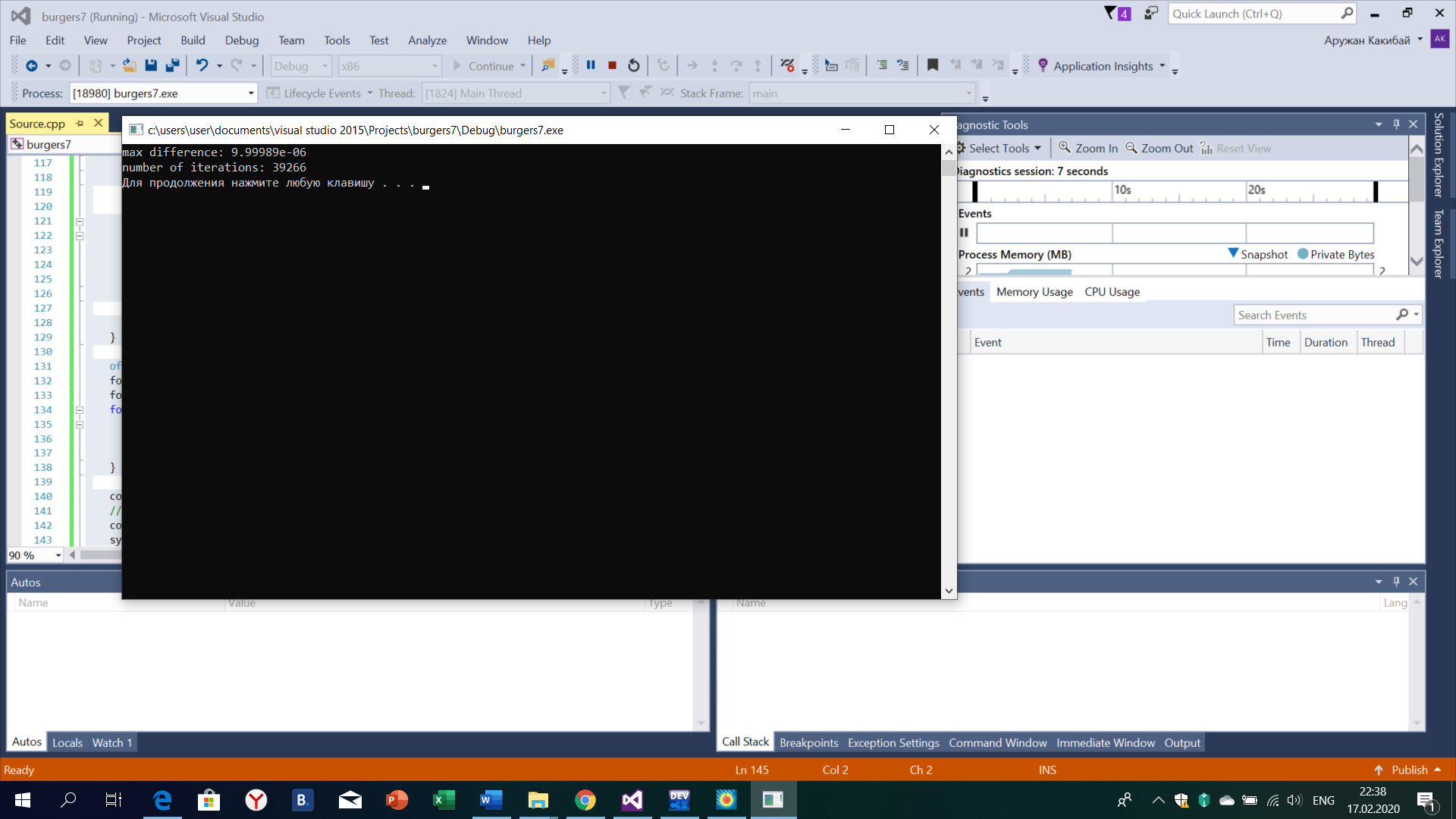
cout << "max difference: " << dif << endl;

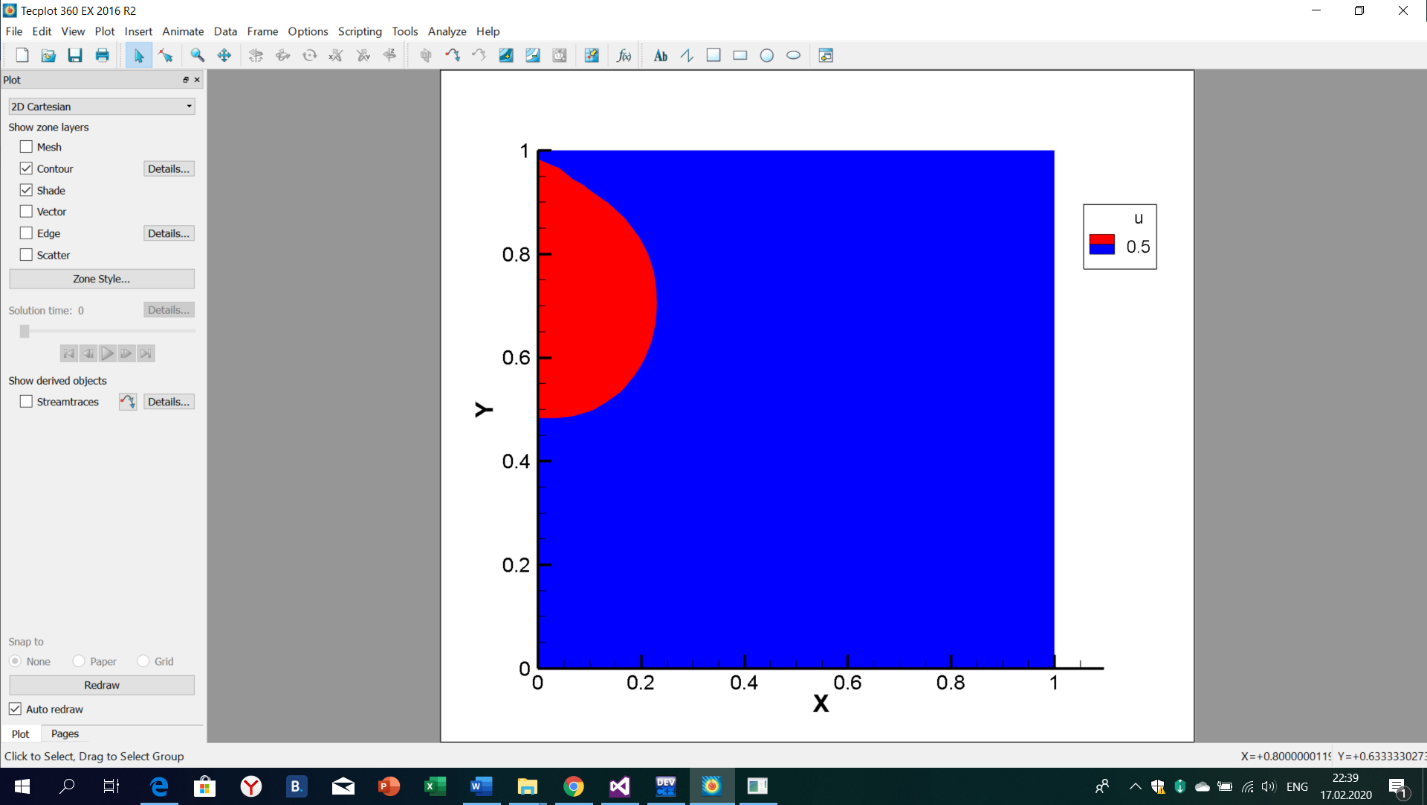
cout << "number of iterations: " << iter << endl;

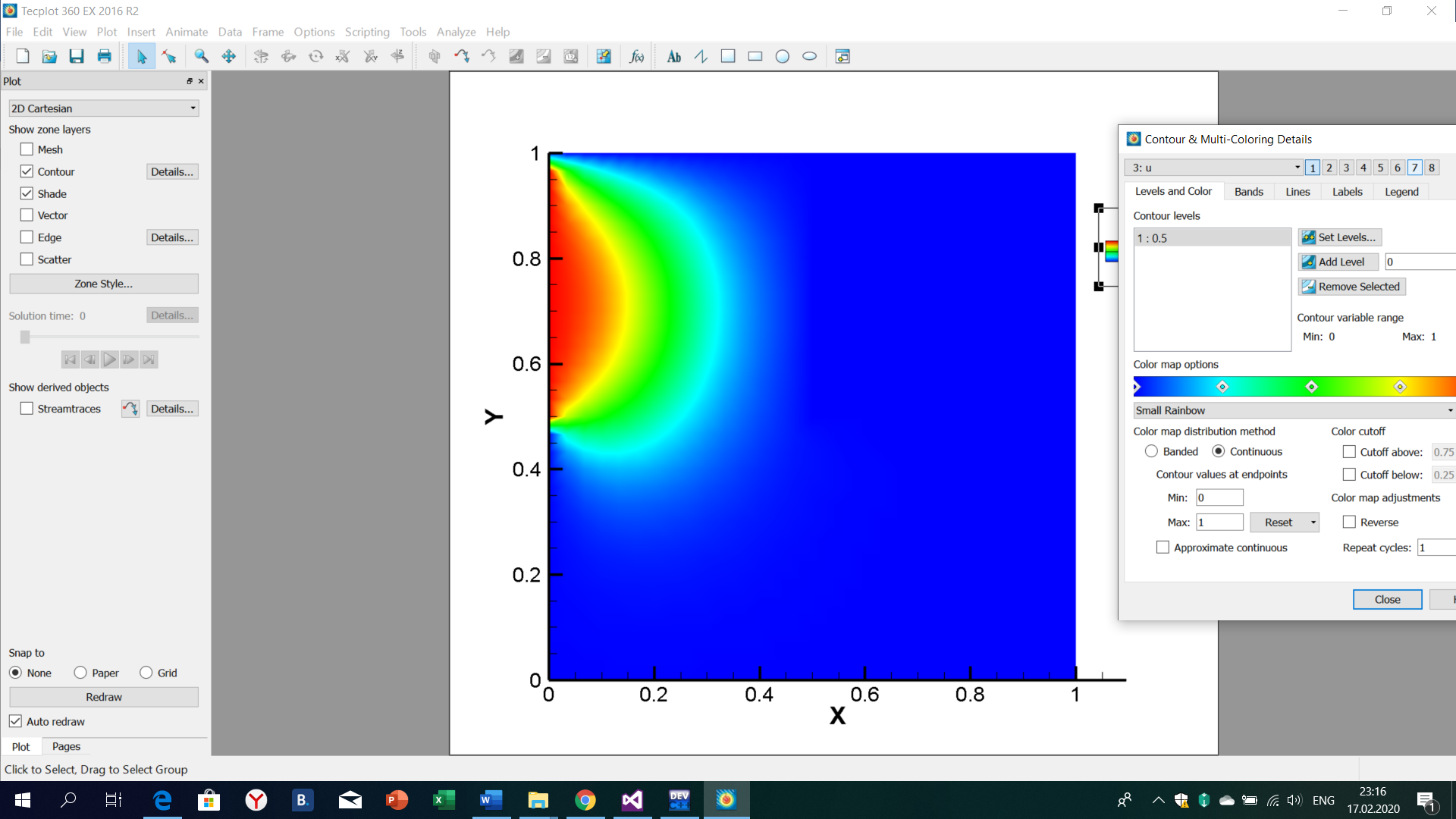
system("pause");

return 0;

}







**Conclusion:**

We approximated 2 dimensional Burgers equation by explicit scheme. We needed to realize the square with obstacle body and square has inlet and outlet. By 7 variant considered own initial condition and boundary conditions which are supported to get satisficed answer. Until steady state was got 39266 iterations and difference between last iterations of u was 9.99989e-06, which means that we solved equation by right way.