**Московский авиационный институт (национальный исследовательский университет)**

**Институт №3.**

«Системы управления, информатика и электроэнергетика»

**Кафедра №304**

«Автоматизированные системы обработки информации и управления»

Отчет по домашней работе

по учебной дисциплине

«Численные методы»

Группа М30-207Б

Выполнил:

Гордеев Н.М.

Принял:

Яганов В.М.

**МОСКВА 2020**

Содержание

[Задание №1 (20.03.20) 4](#_Toc40892972)

[Решение I. 5](#_Toc40892973)

[Решение II. 5](#_Toc40892974)

[Решение III. (Для решения написал программу) 6](#_Toc40892975)

[Код программы 6](#_Toc40892976)

[Результат работы программы 8](#_Toc40892977)

[Задание №2 (03.04.20) 10](#_Toc40892978)

[Решение задачи №1 11](#_Toc40892979)

[I. 11](#_Toc40892980)

[II. Для решения дописал раннее написанную программу 11](#_Toc40892981)

[Дописанный код 11](#_Toc40892982)

[Результат работы программы 12](#_Toc40892983)

[III. Для решения дописал раннее дописанную программу 13](#_Toc40892984)

[Дописанный код 13](#_Toc40892985)

[Результат работы программы 14](#_Toc40892986)

[IV 14](#_Toc40892987)

[Решение задачи №2 15](#_Toc40892988)

[I. 15](#_Toc40892989)

[II. Для решения дописал раннее написанную программу 15](#_Toc40892990)

[Дописанный код 15](#_Toc40892991)

[Результат работы программы 15](#_Toc40892992)

[III. Для решения дописал раннее дописанную программу 16](#_Toc40892993)

[Дописанный код 16](#_Toc40892994)

[Результат работы программы 16](#_Toc40892995)

[IV 16](#_Toc40892996)

[Задание №3 (17.04.20) 17](#_Toc40892997)

[1. Для решения дописал ранее написанную программу 18](#_Toc40892998)

[Дописанный код 18](#_Toc40892999)

[Результат работы программы 19](#_Toc40893000)

[2. Для решения дописал ранее написанную программу 20](#_Toc40893001)

[Дописанный код 20](#_Toc40893002)

[Результат работы программы 20](#_Toc40893003)

[3. Для решения дописал ранее написанную программу 21](#_Toc40893004)

[Результат работы программы 21](#_Toc40893005)

[Задание №4 (24.04.20.) 22](#_Toc40893006)

[1. Для решения дописал ранее написанную программу 23](#_Toc40893007)

[Дописанный код 23](#_Toc40893008)

[Результат работы программы 24](#_Toc40893009)

[2. Для решения дописал ранее написанную программу 25](#_Toc40893010)

[Дописанный код 25](#_Toc40893011)

[Результат работы программы 25](#_Toc40893012)

[Задание №5 26](#_Toc40893013)

[1. Для решения дописал ранее написанную программу 27](#_Toc40893014)

[Дописанный код 27](#_Toc40893015)

[Результат работы программы 28](#_Toc40893016)

[Прямая и кривая в той же системе координат, где нанесены данные точки 28](#_Toc40893017)

[Задача 2. 29](#_Toc40893018)

# Задание №1 (20.03.20)

I. Выделить аналитическим способом корни следующих нелинейных уравнений:

а)

б)

в)

Метод. указания:

1. Выделить, значит определить интервалы переменной Х, в каждом из которых заведомо находится корень (один) данного уравнения.

2. Аналитически – это значит на границах искомых интервалов f(х) должна иметь разные знаки.

3. С помощью 1-х производных этот процесс можно ускорить:

а) находятся значения Х, в которых эти производные равны 0;

б) в этих точках и в определяются знаки f(x);

в) затем эти "грубые" (т.е. большие) интервалы уменьшаются

путем банального пересчета f(x).

II. Выделить графическим способом корни следующих нелинейных уравнений:

а)

б) 

Метод. указания:

1. Привести f(x), например, к виду
2. В системе Оху вычерчиваются две функции:

Точка их пересечения и будет искомым корнем (примерно!!!)

III. Найти **положительные** корни для уравнений I. (а,в) и II. (а,б) методом половинного деления. (Алгоритм, см. лекции)

Точность вычисления корней

## Решение I.

а)

= Ln5 - 6 = 0

Ln5 = 6

X = ; => у 2 корня

= 5 - 6 – 3 = -4 < 0

= + 6 – 3 = 3.2 > 0 Х0 ∈ (-1;)

= 25 + -12 – 3 = 10 > 0 Х1 ∈ (;2)

б)

= - 6 = 0

= 2

X = ± => у 3 корня

= -27 + 18 + 2 = -7 < 0

= -8 + 12 + 2 = 10 > 0 Х0 ∈ (-3;-2)

= 1 -6 + 2 = -3 < 0 Х1 ∈ (-2;1)

= 27 - 18 + 2 = 11 > 0 Х2 ∈ (1;3)

в)

= + 4x – 0.75 = 0

X1 = -1.5; X2 0.1667 => у 3 корня

= -27 + 18 + 2.25 -1 = -7.75 < 0

= -8 + 8 + 1.5 -1 = 0.5 > 0 Х0 ∈ (-3;-2)

= -1 = -1 < 0 Х1 ∈ (-2;0)

= 1 + 2 – 0.75 -1 = 1.25 > 0 Х2 ∈ (0;1)

## Решение II.

а)

;

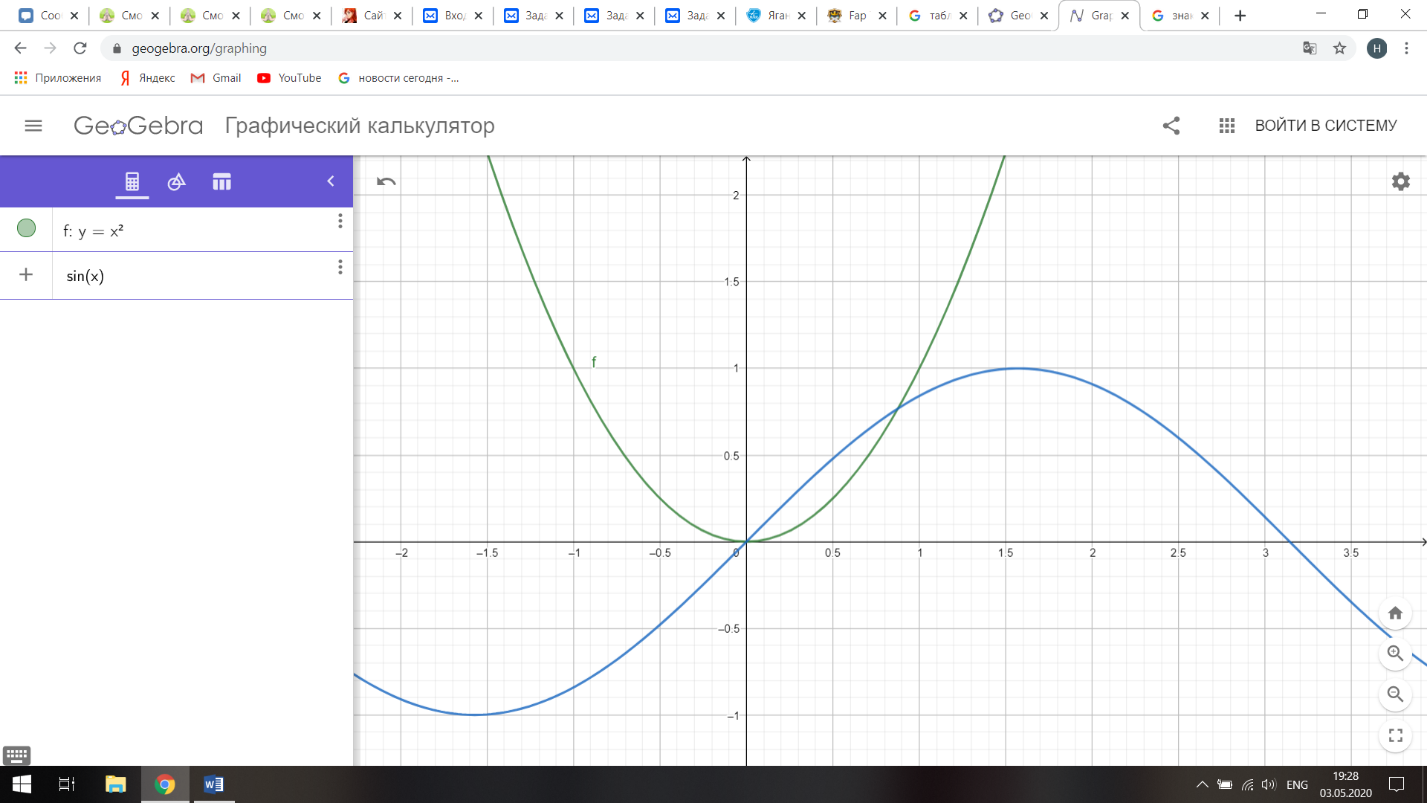
X1 = 0

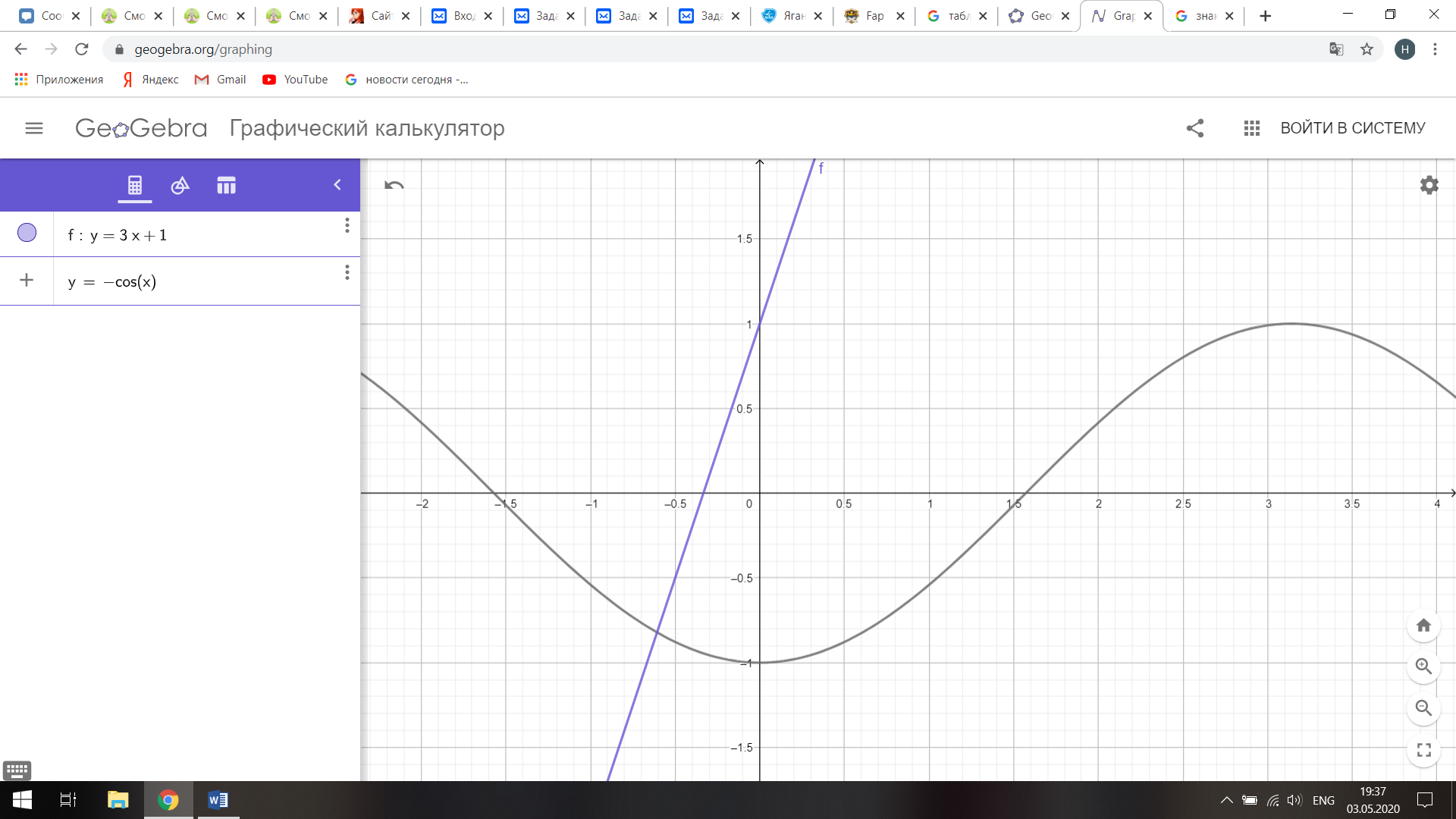
X2  0.8

б) 

3x+1 =

X -0.6

а) б)



## Решение III. (Для решения написал программу)

### Код программы

#include <iostream>

#include <iomanip>

void a() { setlocale(LC\_ALL, "C"); }

void r() { setlocale(LC\_ALL, "Russian"); }

void N\_tabl(int n, char\*\* s);

void C\_tabl(int n, double\* N);

void K\_tabl(int n, double\* N);

double Fx1(double x) { return pow(5, x) - 6 \* x - 3; }

double Fx2(double x) { return pow(x, 3) - 6 \* x + 2; }

double Fx3(double x) { return pow(x, 3) + 2 \* x \* x - 0.75 \* x - 1; }

double Fx4(double x) { return sin(x) - x \* x; }

double Fx5(double x) { return 3\*x + cos(x) + 1; }

double ПоловинноеДеление(double a, double b, double (\*F)(double), double Eps);

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian"); // подключаю русский язык

double a = 5; // верхняя граница значений х

double b = 0; // нижняя граница значений х

double x = a; // переменная

cout << "1\nКорни 1 ур-я\n";

cout << "Ответ: x = " << ПоловинноеДеление(-1, 1, Fx1, 0.01) << endl;

cout << "Ответ: x = " << ПоловинноеДеление(1, 2, Fx1, 0.01) << endl;

cout << "\nКорни 2 ур-я \n";

cout << "Ответ: x = " << ПоловинноеДеление(-3, -2, Fx2, 0.01) << endl;

cout << "Ответ: x = " << ПоловинноеДеление(-2, 1, Fx2, 0.01) << endl;

cout << "Ответ: x = " << ПоловинноеДеление(1, 3, Fx2, 0.01) << endl;

cout << "\nКорни 3 ур-я \n";

cout << "Ответ: x = " << ПоловинноеДеление(-3, -2, Fx3, 0.01) << endl;

cout << "Ответ: x = " << ПоловинноеДеление(-2, 0, Fx3, 0.01) << endl;

cout << "Ответ: x = " << ПоловинноеДеление(0, 1, Fx3, 0.01) << endl;

cout << "\n\n2\nКорни 1 ур-я\n";

cout << "Ответ: x = " << ПоловинноеДеление(0.5, 1, Fx4, 0.01) << endl;

cout << "\nКорни 2 ур-я \n";

cout << "Ответ: x = " << ПоловинноеДеление(-1, 0, Fx5, 0.01) << endl;

}

double ПоловинноеДеление(double a1, double b1, double (\*F)(double), double Eps) {

const int n = 5;

int i = 0;

double N[n];

char\*\* s;

s = new char\* [n];

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

s[i] = new char[15];

char s0[15] = { " i " };

char s1[15] = { " a " };

char s2[15] = { " b " };

char s3[15] = { " x " };

char s4[15] = { " f(x) " };

s[0] = s0;

s[1] = s1;

s[2] = s2;

s[3] = s3;

s[4] = s4;

N\_tabl(n, s);

N[0] = i;

N[1] = a1;

N[2] = b1;

N[3] = a1;

N[4] = F(a1);

double a = a1;

double b = b1;

double x = a;

if (F(a) < F(b)) { a = b; b = x; x = a; }

while (abs(F(x)) > Eps)

{

C\_tabl(n, N);

i++;

if (F(x) < 0)

b = x;

else

a = x;

x = (a + b) / 2;

N[0] = i;

N[1] = a;

N[2] = b;

N[3] = x;

N[4] = F(x);

}

K\_tabl(n, N);

return x;

}

void N\_tabl(int n, char\*\* s)

{

a(); cout << char(218) << setfill(char(196)) << setw(15);

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

{

cout << char(194) << setfill(char(196)) << setw(15);

}

cout << char(191) << endl;

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

{

cout << char(179); r(); cout << s[i]; a();

}

cout << char(179) << endl;

cout << char(195) << setfill(char(196)) << setw(15);

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

{

cout << char(197) << setfill(char(196)) << setw(15);

}

cout << char(180) << endl;

}

void C\_tabl(int n, double\* N) {

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

{

cout << char(179) << " " << setfill(char(32)) << setw(12) << N[i] << " ";

}

cout << char(179) << endl;

cout << char(195);

cout << setfill(char(196)) << setw(15);

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

{

cout << char(197) << setfill(char(196)) << setw(15);

}

cout << char(180) << endl;

}

void K\_tabl(int n, double\* N) {

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

{

cout << char(179) << " " << setfill(char(32)) << setw(12) << N[i] << " ";

}

cout << char(179) << endl;

cout << char(192);

cout << setfill(char(196)) << setw(15);

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

{

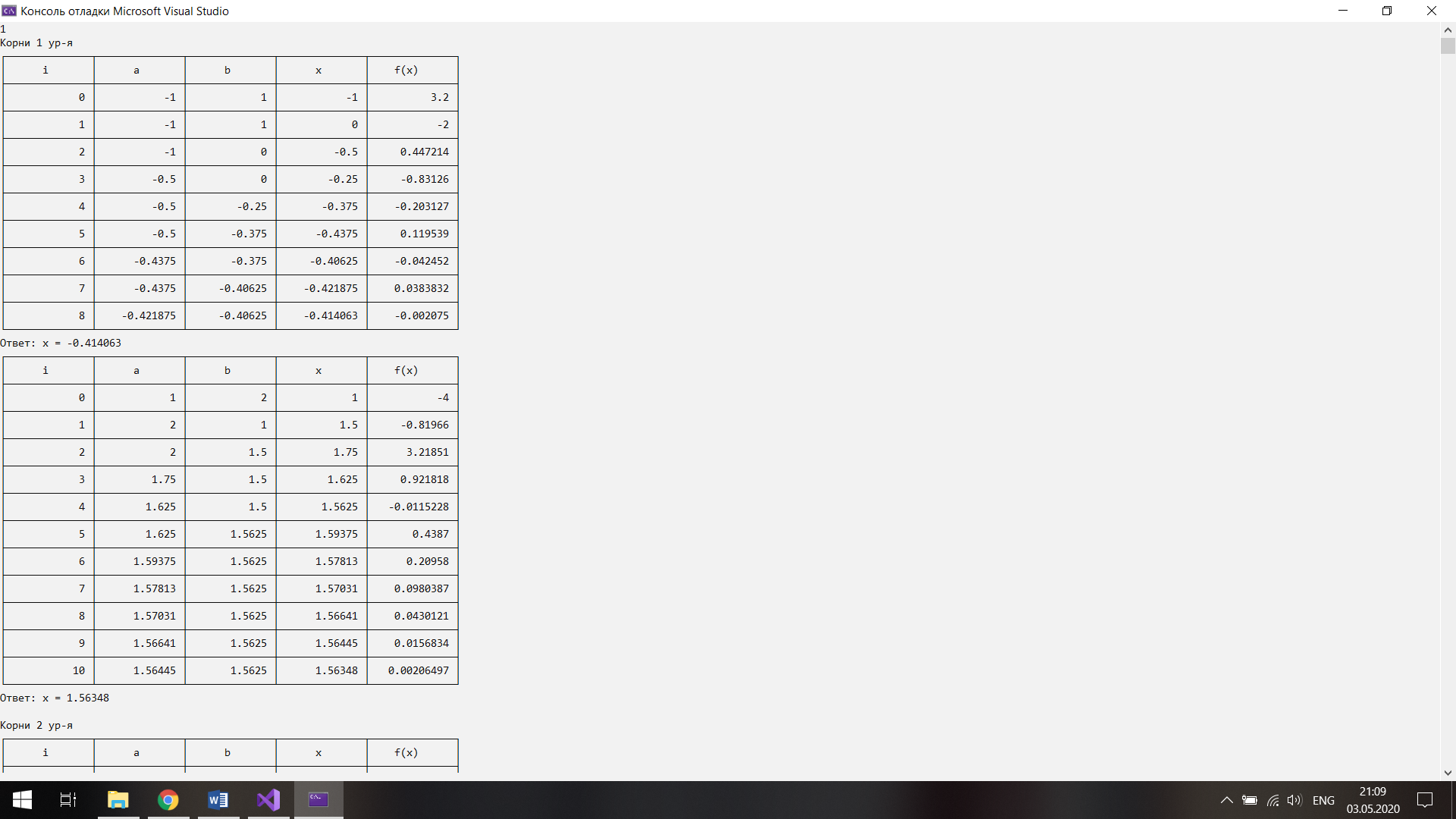
cout << char(193) << setfill(char(196)) << setw(15);

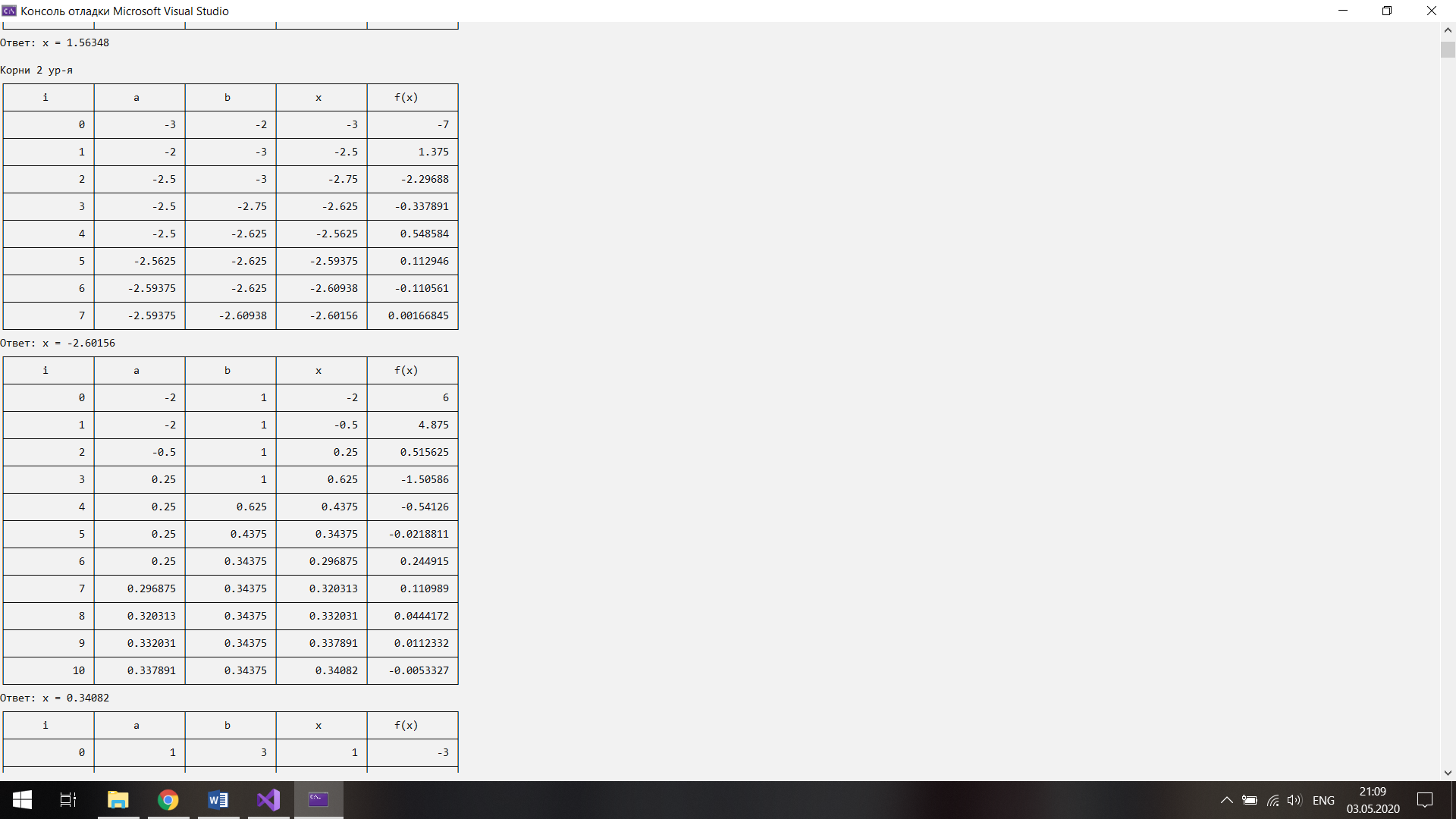
}

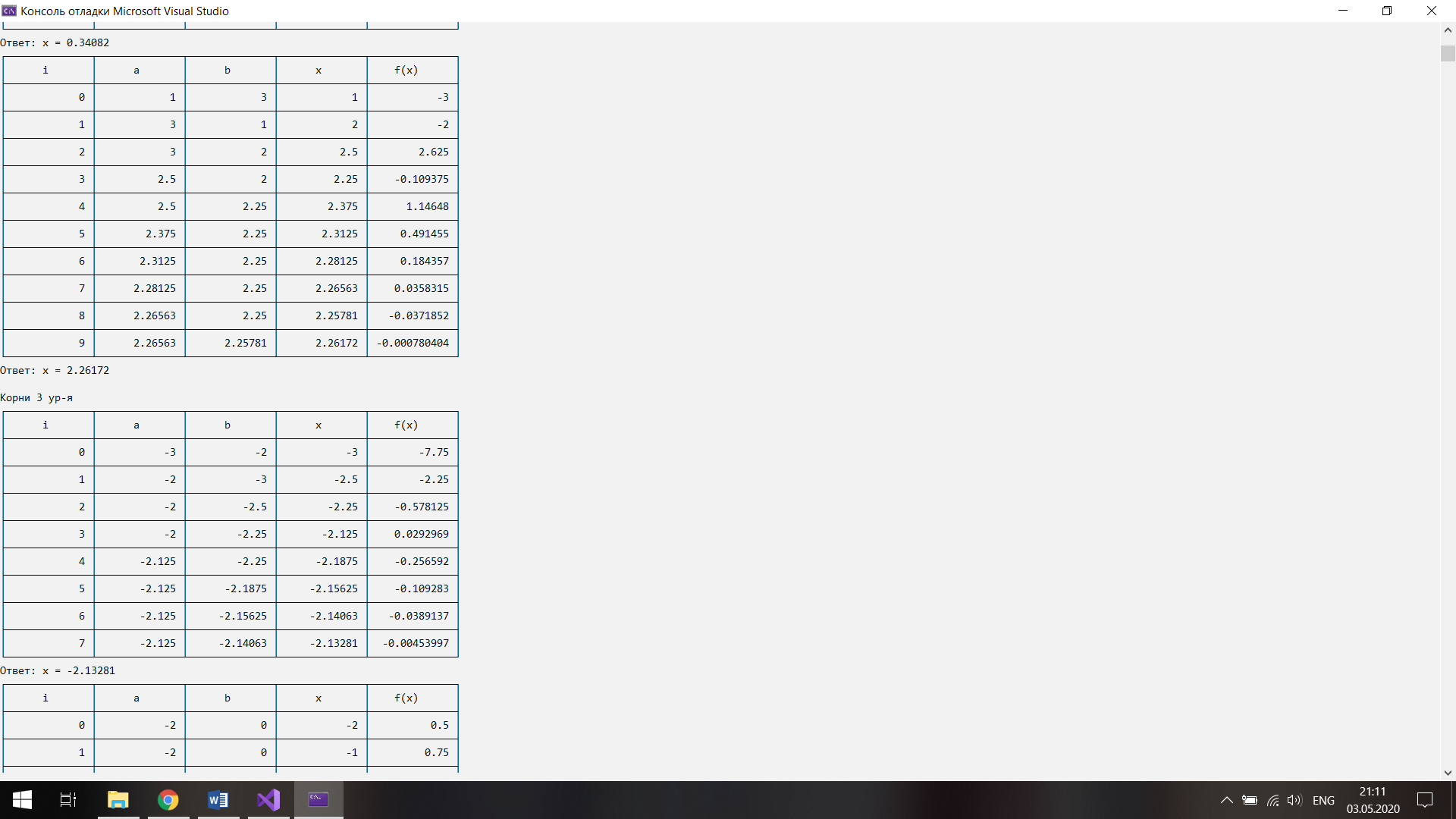
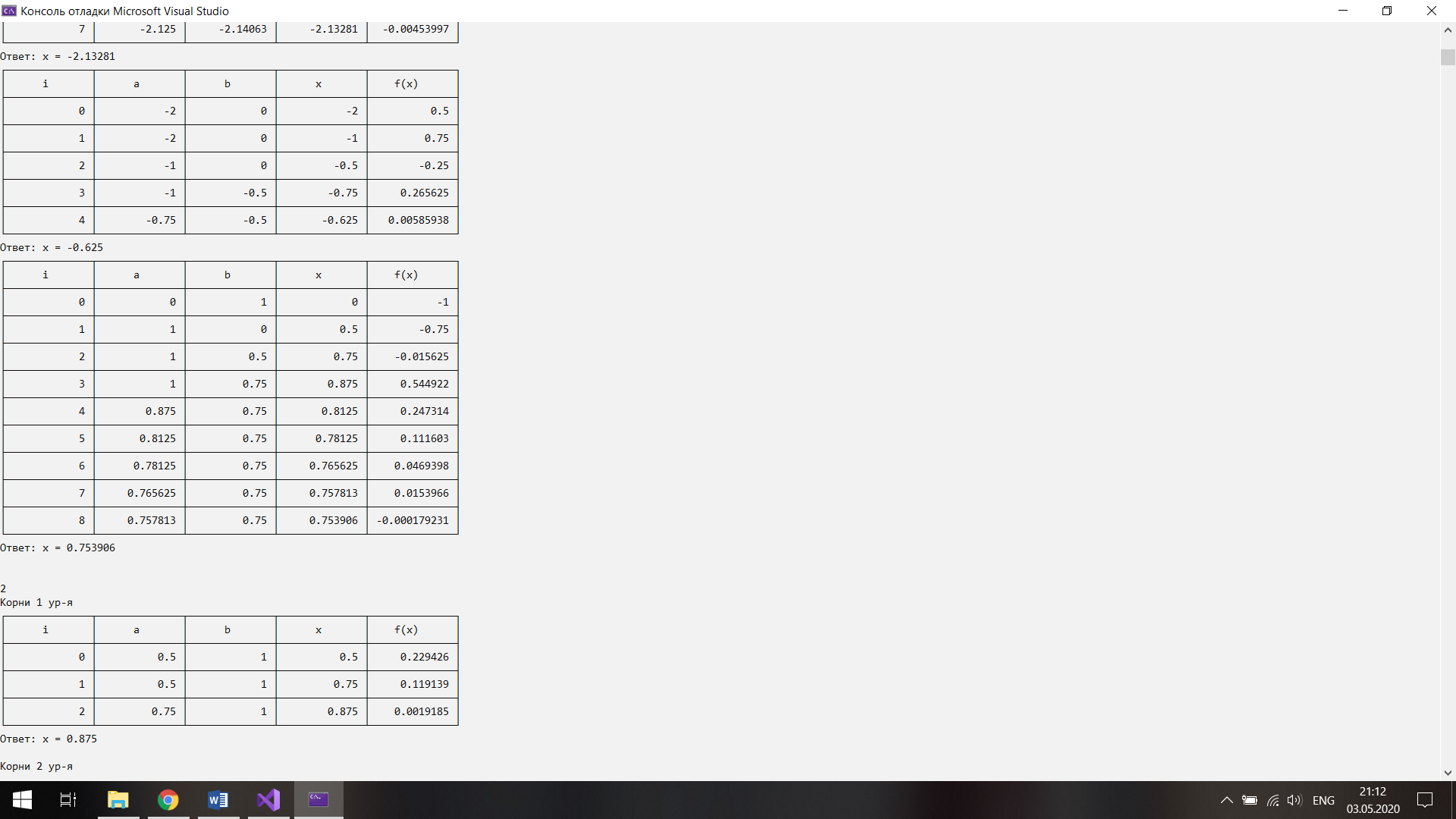
cout << char(217) << endl; r();

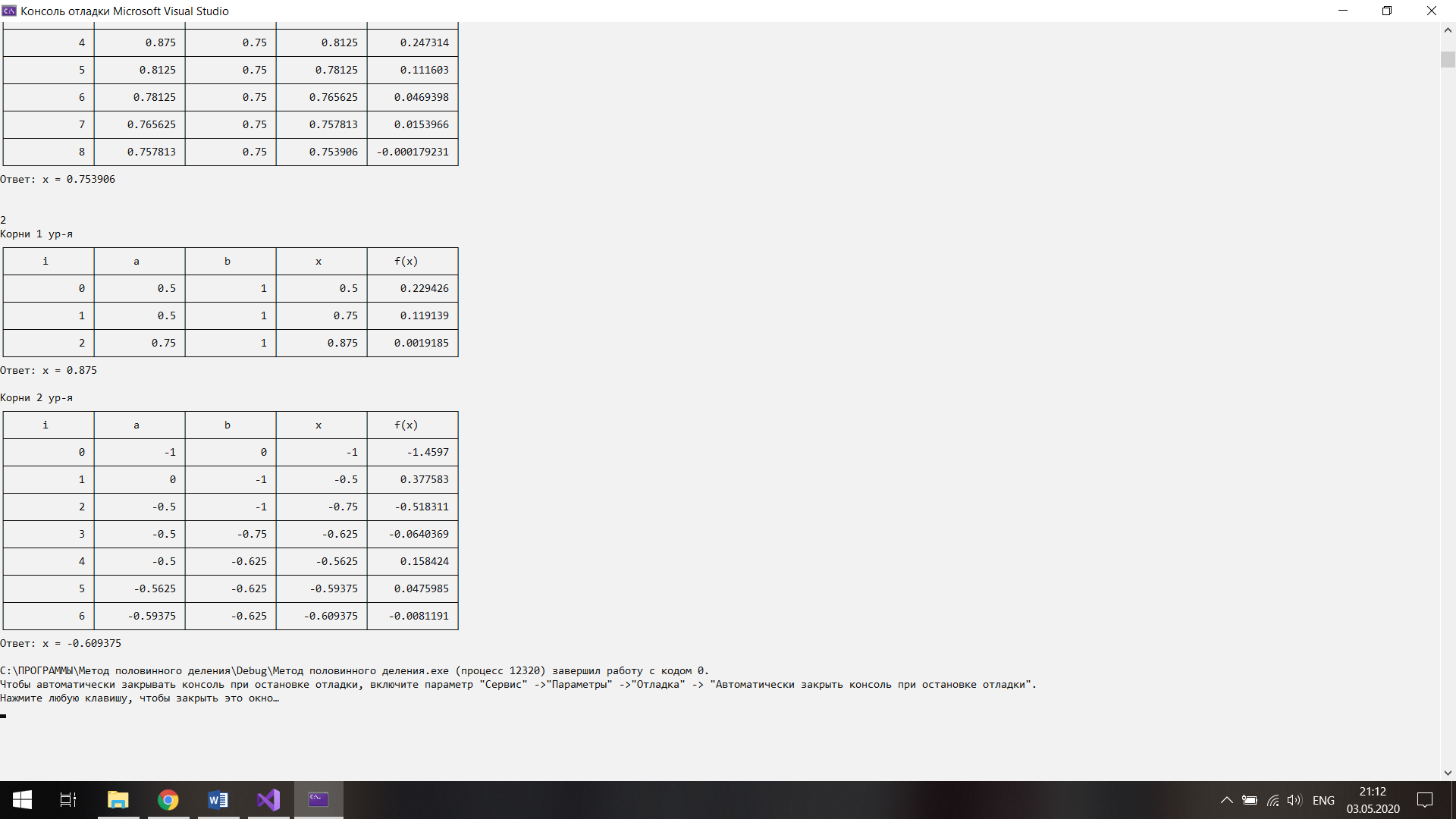
}

### Результат работы программы









# Задание №2 (03.04.20)

Тема: Метод простой итерации и метод Ньютона.

**Задача №1**

I. Выделить аналитическим способом **один положительный** корень.

II. Уточнить этот корень методом простой итерации с точностью

Метод. указания:

Привести уравнение f(x)=0 любым способом к итерационной формуле вида

Чтобы при этом выполнялось условие сходимости (см. лекцию), эту итерационную формулу удобно записать в виде , где . Максимальное значение ищется на выделенном интервале. Тогда тут

III. Решить задачу №1 методом Ньютона с точностью .

(Проверить задачу на сходимость!! См. лекцию)

IV. Сравнить результаты полученные двумя разными методами.

**Задача № 2.**

I. Выделить графическим способом корень нелинейного уравнения.

II. Решить задачу методом простой итерации с точностью

( С предварительной проверкой на сходимость!)

III. Решить задачу № 2 методом Ньютона с точностью .

( С предварительной проверкой на сходимость!)

IV. Сравнить результаты.

## Решение задачи №1

## I.

= -5

= 8

Ответ: Х0 ∈ (1;2)

## II. Для решения дописал раннее написанную программу

= 4x3 – 2

= 1– 2 = -1

= 32– 2 = 30

λ = -2/30 = -0.0667

### Дописанный код

Глобальная область:

double Fx6(double x) { return pow(x, 4) - 2 \* x - 4; }

double ПростойИтерации(double х, double a, double (\*F)(double), double Eps);

main:

cout << "Ответ: x = " << ПростойИтерации(2, -0.0667, Fx6, 0.001) << endl;

Реализация ф-ий:

double ПростойИтерации(double x1, double a, double (\*F)(double), double Eps) {

const int n = 5;

int i = 0;

double N[n];

char\*\* s;

s = new char\* [n];

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

s[i] = new char[15];

char s0[15] = { " i " };

char s1[15] = { " x " };

char s2[15] = { " a " };

char s3[15] = { " f(x) " };

char s4[15] = { " fl(x) " };

s[0] = s0;

s[1] = s1;

s[2] = s2;

s[3] = s3;

s[4] = s4;

N\_tabl(n, s);

N[0] = i;

N[1] = x1;

N[2] = a;

N[3] = F(x1);

N[4] = x1 + a\*F(x1);

double x = x1;

while (abs(F(x)) > Eps)

{

C\_tabl(n, N);

i++;

x = x + a \* F(x);

N[0] = i;

N[1] = x;

N[2] = a;

N[3] = F(x);

N[4] = x + a \* F(x);

}

K\_tabl(n, N);

return x;

}

### Результат работы программы



## III. Для решения дописал раннее дописанную программу

Задача №1

= 4x3 – 2

### Дописанный код

Глобальная область:

double Fx16(double x) { return 4\*pow(x, 3) - 2; }

double Ньютон(double x1, double (\*F)(double), double (\*F1)(double), double Eps);

main:

cout << "Ответ: x = " << Ньютон(2, Fx6, Fx16, 0.001) << endl;

Реализация ф-ий:

double Ньютон(double x1, double (\*F)(double), double (\*F1)(double), double Eps) {

const int n = 5;

int i = 0;

double N[n];

char\*\* s;

s = new char\* [n];

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

s[i] = new char[15];

char s0[15] = { " i " };

char s1[15] = { " xn " };

char s2[15] = { " f(x) " };

char s3[15] = { " f1(x) " };

char s4[15] = { " xn+1 " };

s[0] = s0;

s[1] = s1;

s[2] = s2;

s[3] = s3;

s[4] = s4;

N\_tabl(n, s);

N[0] = i;

N[1] = x1;

N[2] = F(x1);

N[3] = F1(x1);

N[4] = x1 - F(x1) / F1(x1);

double x = x1;

while (abs(F(x)) > Eps)

{

C\_tabl(n, N);

i++;

x = x - F(x) / F1(x);

N[0] = i;

N[1] = x;

N[2] = F(x);

N[3] = F1(x);

N[4] = x - F(x) / F1(x);

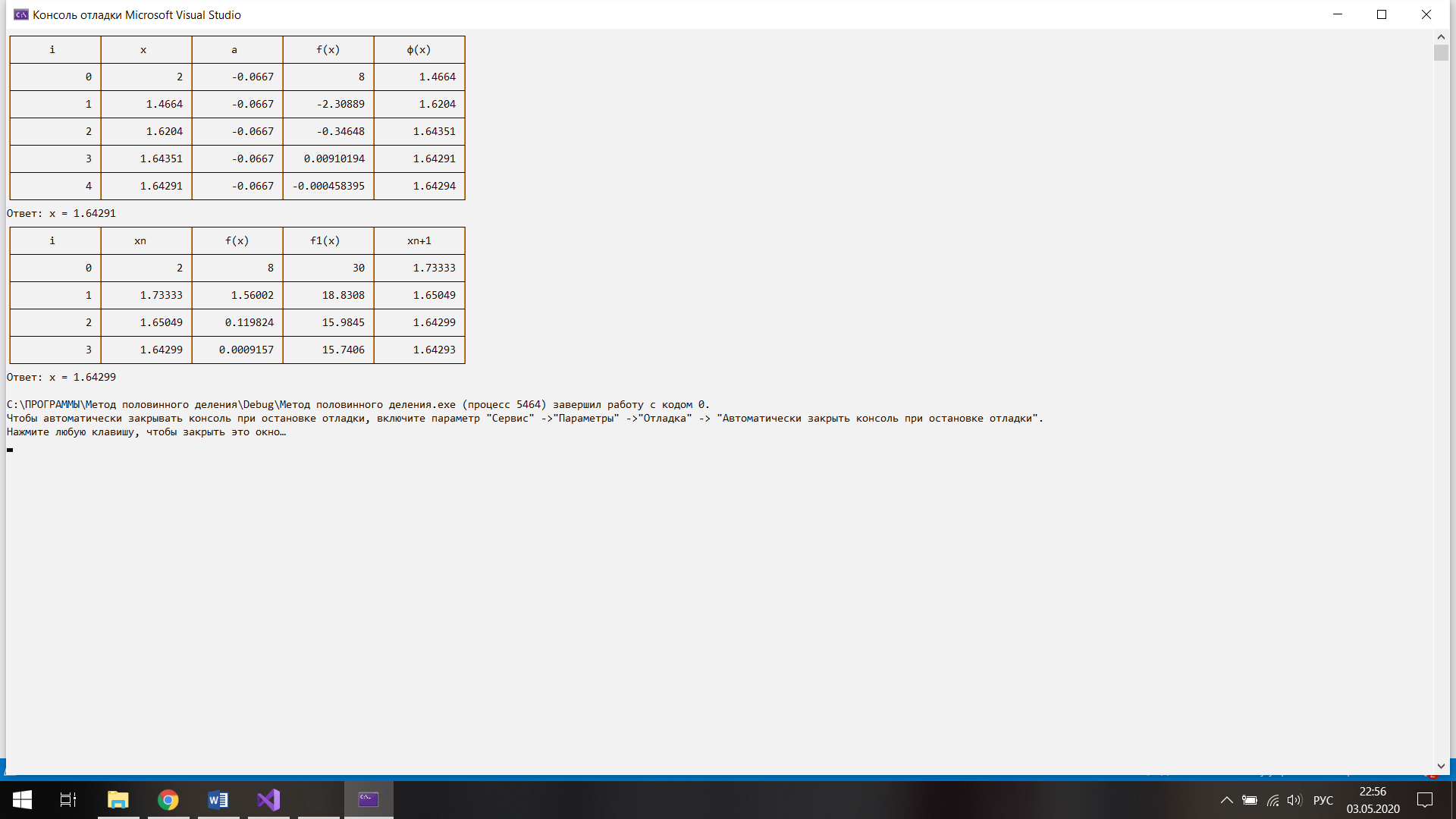
}

K\_tabl(n, N);

return x;

}

### Результат работы программы



## IV

Метод ньютона показал более быструю сходимость, чем метод простой итерации.

## Решение задачи №2

## I.

Ответ: Х0 ∈ (пi/3;1)

## II. Для решения дописал раннее написанную программу

= 2cos2x – 2x

= -2.8

= -1– 2 = -3

λ = 2/3 = 0.667

### Дописанный код

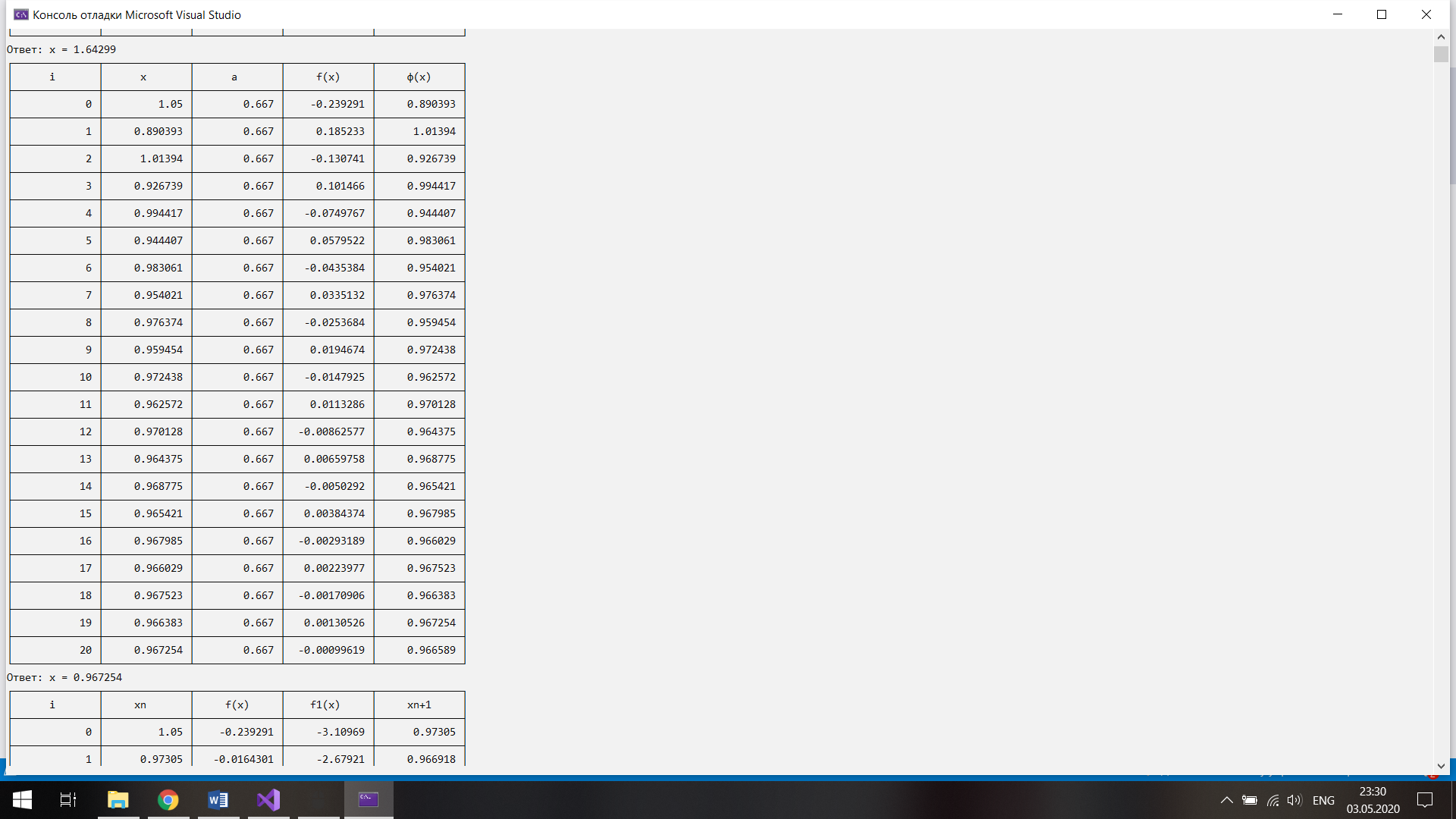
Глобальная область:

double Fx7(double x) { return sin(2\*x) - x \* x; }

main:

cout << "Ответ: x = " << ПростойИтерации(1.05, 0.667, Fx7, 0.001) << endl;

### Результат работы программы



## III. Для решения дописал раннее дописанную программу

Задача №1

= 2cos2x – 2x

### Дописанный код

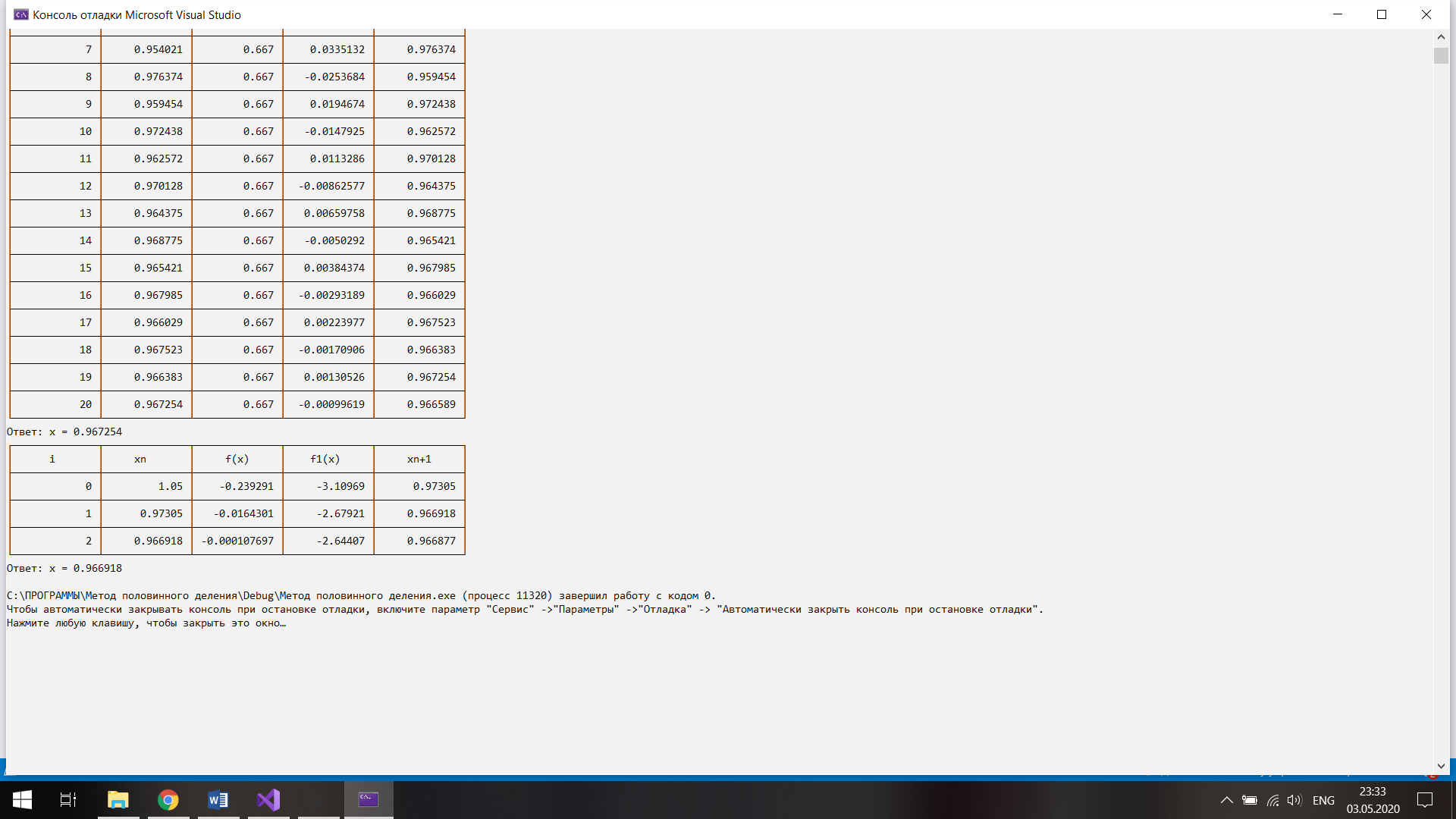
Глобальная область:

double Fx17(double x) { return 2 \* cos(2\*x) - 2\*x; }

main:

cout << "Ответ: x = " << Ньютон(1.05, Fx7, Fx17, 0.001) << endl;

### Результат работы программы



## IV

Метод ньютона показал более быструю сходимость, чем метод простой итерации.

# Задание №3 (17.04.20)

1. Решить нелинейную задачу из предыдущего

задания , т.е методом хорд.

Точность

2. Найти корни методом простых итераций для след. системы:

Точность

3. Методом Ньютона найти корни в системе уравнений:

+

Точность

## 1. Для решения дописал ранее написанную программу

### Дописанный код

Глобальная область:

double МетодХорд(double a, double b, double (\*F)(double), double Eps);

main:

cout << "Ответ: x = " << МетодХорд(1, 2, Fx6, 0.01) << endl;

Реализация ф-ий:

double МетодХорд(double a1, double b1, double (\*F)(double), double Eps) {

const int n = 5;

int i = 0;

double N[n];

char\*\* s;

s = new char\* [n];

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

s[i] = new char[15];

char s0[15] = { " i " };

char s1[15] = { " a " };

char s2[15] = { " b " };

char s3[15] = { " x " };

char s4[15] = { " f(x) " };

s[0] = s0;

s[1] = s1;

s[2] = s2;

s[3] = s3;

s[4] = s4;

N\_tabl(n, s);

N[0] = i;

N[1] = a1;

N[2] = b1;

N[3] = a1;

N[4] = F(a1);

double a = a1;

double b = b1;

double x = b;

bool t = 0;

if (a < b) { b = a; a = x; x = b; }

if (F(a) > 0) t = 1;

while (abs(F(x)) > Eps)

{

C\_tabl(n, N);

i++;

if(t)

x = x - F(x) / (F(x) - F(a)) \* (x - a);

else

x = x - F(x) / (F(b) - F(x)) \* (b - x);

N[0] = i;

N[1] = a;

N[2] = b;

N[3] = x;

N[4] = F(x);

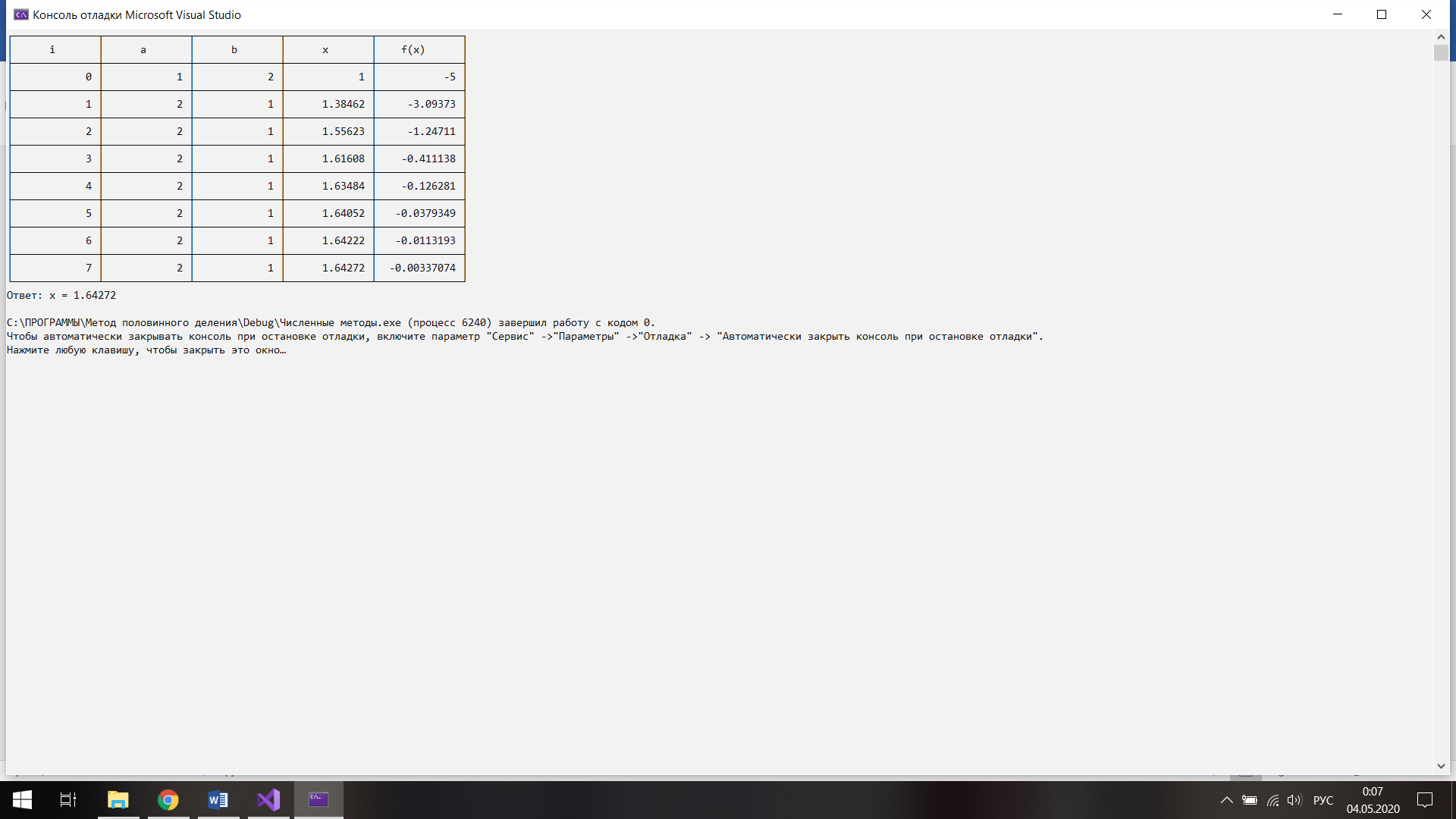
}

K\_tabl(n, N);

return x;

}

### Результат работы программы



## 2. Для решения дописал ранее написанную программу

Подставляю ф-ю в программу

Y(0) < 0

Y(1)> 0

### Дописанный код

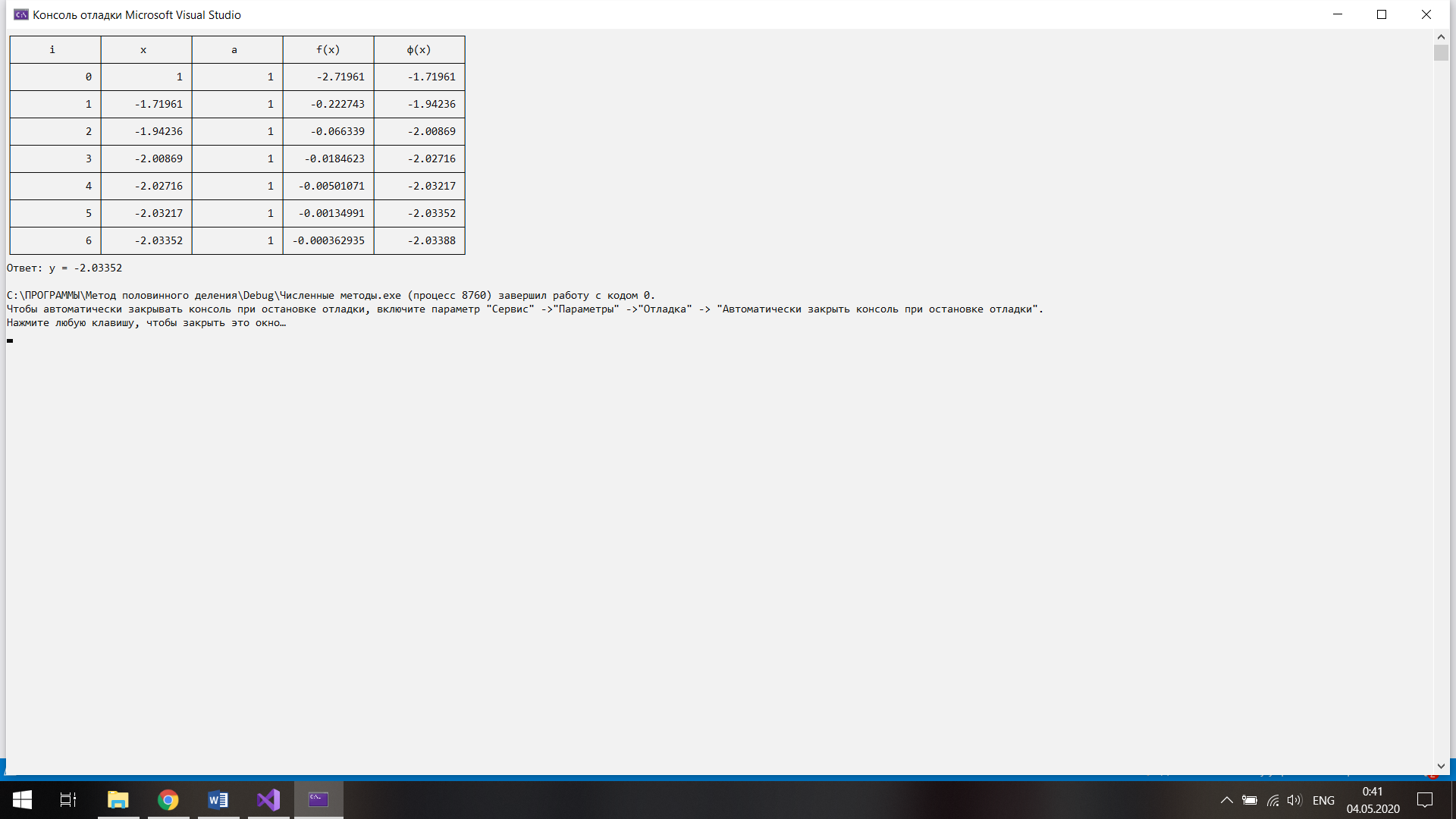
Глобальная область:

double Fx8(double x) { return sin(0.3 + cos(x)/3 -0.6) - 1.6 -x; }

main:

cout << "Ответ: y = " << ПростойИтерации(1, 1, Fx8, 0.001) << endl;

### Результат работы программы



X = = 0.1512

## 3. Для решения дописал ранее написанную программу

+

Вводим уравнение X2 + (3-x)2 – 9 =у

У, = 2х – 2(3-х)

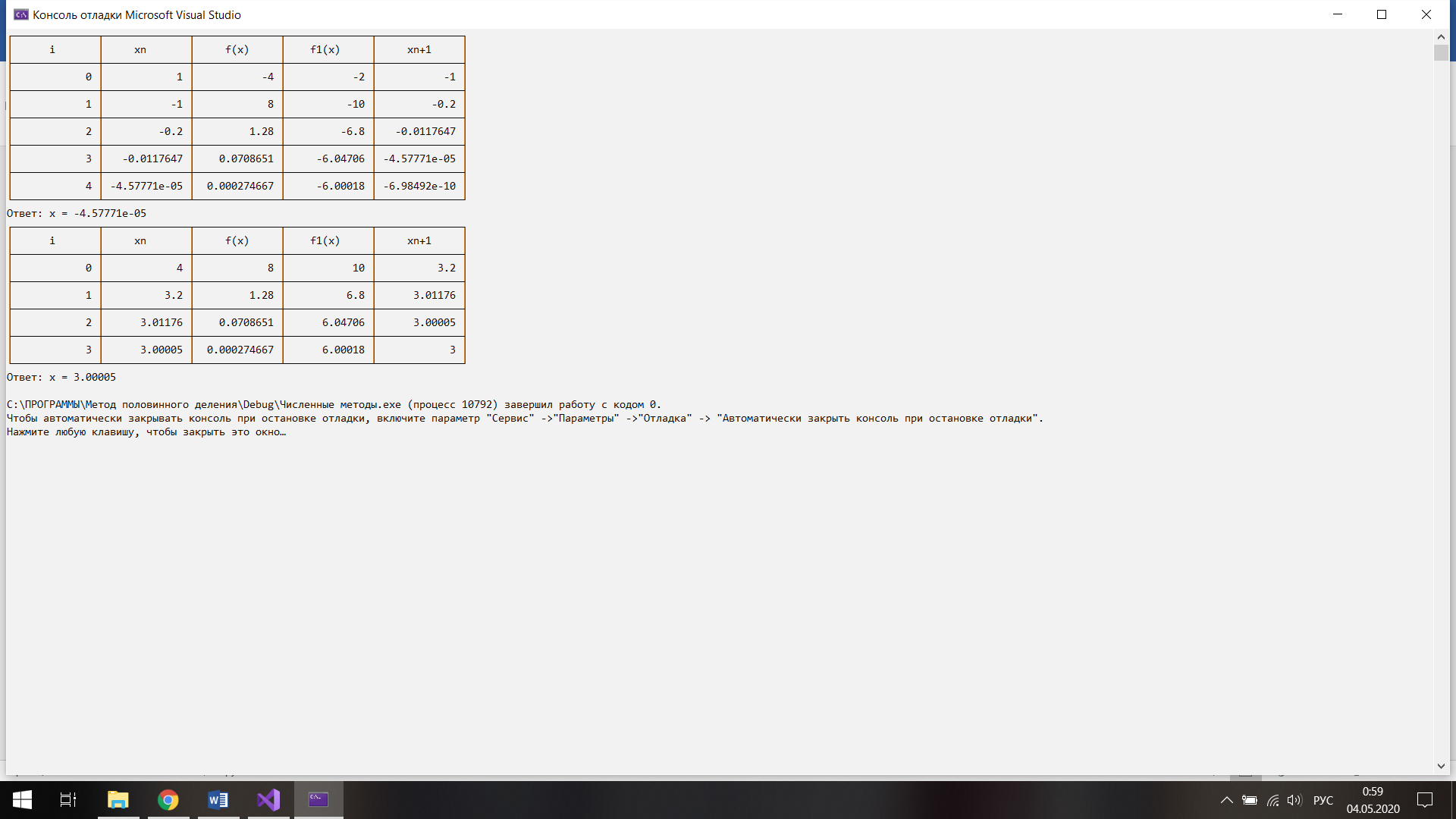
double Fx9(double x) { return x\*x + (3 - x)\*(3 - x) - 9; }

double Fx19(double x) { return 2x - 2\*(3 - x); }

cout << "Ответ: x = " << Ньютон(1, Fx9, Fx19, 0.001) << endl;

cout << "Ответ: x = " << Ньютон(4, Fx9, Fx19, 0.001) << endl;

### Результат работы программы



Корни очивидны.

Ответ (0;3), (3;0).

# Задание №4 (24.04.20.)

№1

Заданы значения функции  в узлах .Найти значения функции  при  с помощью интерполяционных формул Ньютона.

|  |  |  |
| --- | --- | --- |
|  | **1** | **2** |
| 1,0 | 0,8 | 1,1 |
| 1,2 | 1,8 | 2,2 |
| 1,4 | 2,9 | 3,0 |
| 1,6 | 4,0 | 4,1 |
| 1,8 | 4,9 | 4,9 |
| 2,0 | 6,1 | 5,9 |

№2

Заданы значения  функции  в точках . Найти значение функции  при . Задачу решить с помощью интерполяционного многочлена Лагранжа.

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | | **2** | |
|  |  |  |  |
| 0 | 11 | 0 | 11 |
| 2 | 13 | 1 | 12 |
| 3 | 13 | 3 | 13 |
| 5 | 14 | 5 | 14 |
|  | |  | |

## 1. Для решения дописал ранее написанную программу

### Дописанный код

Глобальная область:

const int m4 = 6;

double х3[m4] = { 1, 1.2, 1.4, 1.6, 1.8, 2 };

double у3[m4] = { 0.8, 1.8, 2.9, 4.0, 4.9, 6.1 };

double у4[m4] = { 1.1, 2.2, 3.0, 4.1, 4.9, 5.9 };

double ИнтерполяцияНьютон(double\* x, double\* y, double\* a, int k, double t);

main:

double ab4[m3];

cout << "Ответ: y(1.1) = " << ИнтерполяцияНьютон(х3, у3, ab4, m4, 1.1) << endl;

cout << "Ответ: y(1.1) = " << ИнтерполяцияНьютон(х3, у4, ab4, m4, 1.1) << endl;

Реализация ф-ий:

double Px(double\* x, double\* a, int k, double t)

{

double S = 0;

double P = 1;

for (size\_t i = 0; i < k; i++)

{

for (size\_t j = 0; j < i; j++)

P = P \* (t - x[j]);

S = S + a[i]\*P;

P = 1;

}

return S;

}

double ИнтерполяцияНьютон(double\* x, double\* y, double\* a, int k, double t) {

const int n = 5;

double N[n];

char\*\* s;

s = new char\* [n];

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

s[i] = new char[15];

char s0[15] = { " i " };

char s1[15] = { " x " };

char s2[15] = { " y " };

char s3[15] = { " P(xi) " };

char s4[15] = { " a " };

s[0] = s0;

s[1] = s1;

s[2] = s2;

s[3] = s3;

s[4] = s4;

N\_tabl(n, s);

N[0] = 0;

N[1] = x[0];

N[2] = y[0];

N[3] = Px(x, a, 0, x[0]);

N[4] = y[0];

double S = 0;

double P = 1;

for (size\_t i = 0; i < k; i++)

a[i] = 0;

for (size\_t i = 0; i < k; i++)

{

C\_tabl(n, N);

for (size\_t j = 0; j < i; j++)

P = P \* (x[i] - x[j]);

a[i] = (y[i]- Px(x, a, i, x[i])) / P;

P = 1;

N[0] = i;

N[1] = x[i];

N[2] = y[i];

N[3] = Px(x, a, i, x[i]);

N[4] = a[i];

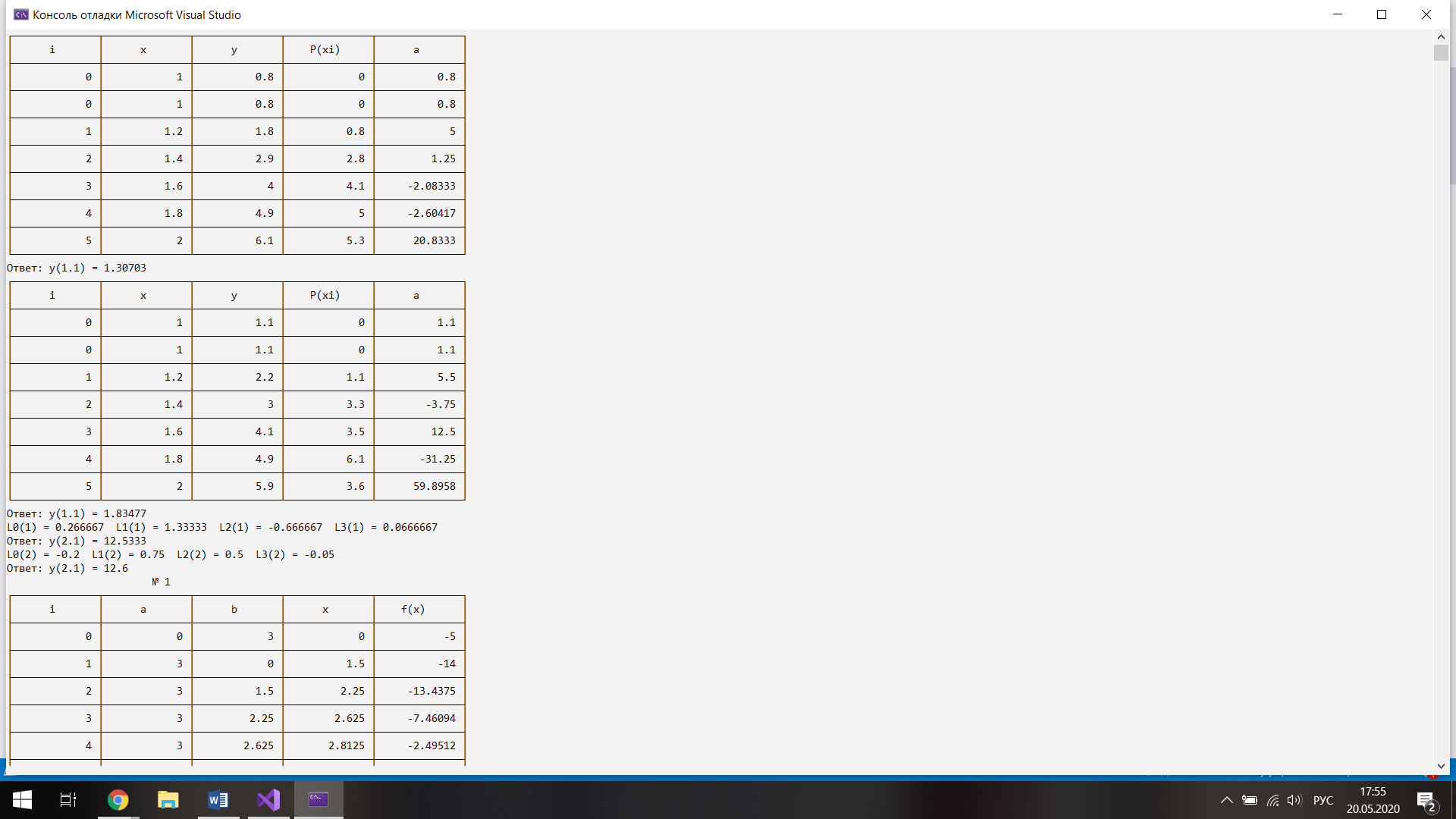
}

K\_tabl(n, N);

return Px(x, a, k, t);

}

### Результат работы программы



## 2. Для решения дописал ранее написанную программу

### Дописанный код

Глобальная область:

const int m5 = 4;

double х5[m5] = { 0, 2, 3, 5 };

double у5[m5] = { 11,13,13,14 };

double х6[m5] = { 0, 1, 3, 5 };

double у6[m5] = { 11,12,13,14 };

double ИнтерполяцияЛогранж(double\* x, double\* y, double\* a, int k, double t);

main:

double ab5[m5];

cout << "Ответ: y(2.1) = " << ИнтерполяцияЛогранж(х5, у5, ab5, m5, 1) << endl;

cout << "Ответ: y(2.1) = " << ИнтерполяцияЛогранж(х6, у6, ab5, m5, 2) << endl;

Реализация ф-ий:

double ИнтерполяцияЛогранж(double\* x, double\* y, double\* a, int k, double t) {

double S = 0;

double P = 1;

for (size\_t i = 0; i < k; i++)

{

for (size\_t j = 0; j < k; j++)

if (i != j)

P \*= (t - x[j]) / (x[i] - x[j]);

cout << "L" << i << "(" << t << ") = " << P << " ";

S += y[i] \* P;

P = 1;

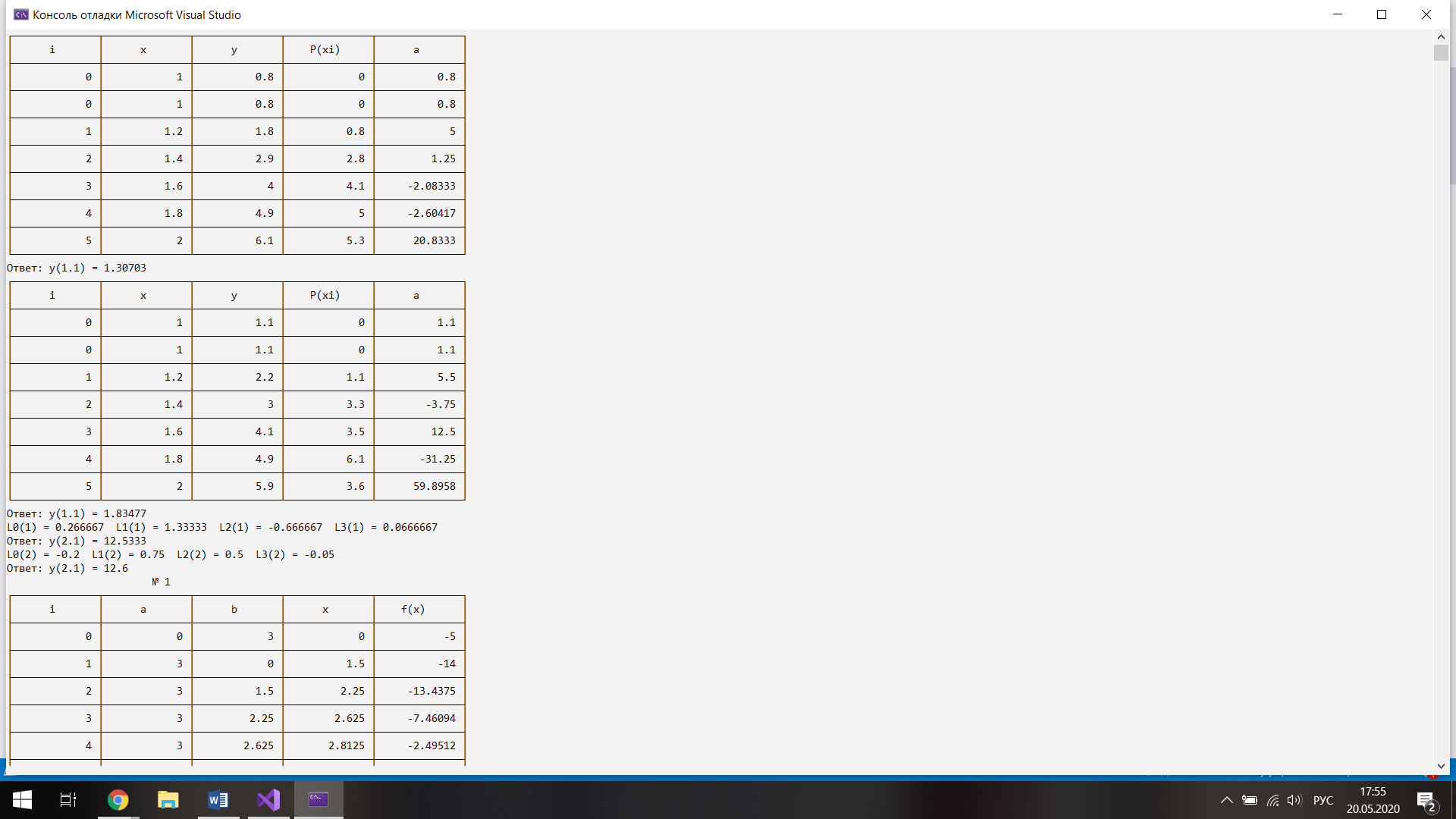
}

cout << "\n";

return S;

}

### Результат работы программы



# Задание №5

Задача 1.

По заданным значениям  и  найти прямую  и параболу

методом наименьших квадратов. Найти погрешность. Построить прямую и кривую в той же системе координат, где нанесены данные точки.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Х | 0.4 | 0.95 | 1.12 | 1.24 | 2.34 | 2.78 | 3.7 |
| Y | 0.96 | 2.23 | 2.38 | 2.98 | 4.77 | 6.07 | 7.77 |

Задача 2.

Найти 1-ю и 2-ю производную для функции заданной в виде таблицы в точке

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | -1 | 0 | 1 | 2 |
| Y | 0 | 1 | 2 | 3 |

## 1. Для решения дописал ранее написанную программу

### Дописанный код

Глобальная область:

double х7[7] = { 0.4, 0.95,1.12,1.24,2.34,2.78,3.7 };

double у7[7] = { 0.96,2.23,2.38,2.98,4.77,6.07,7.77 };

void НаимКвЛин(double\* x, double\* y, double\* a, int k, bool t = 1);

main:

double ab6[3];

НаимКвЛин(х7, у7, ab6, 7);

cout << "Ответ: (а;b) = (" << ab6[1] << ";" << ab6[0] << ")" << endl;

НаимКвЛин(х7, у7, ab6, 7, 0);

cout << "Ответ: (а;b;c) = (" << ab6[2] << ";" << ab6[1] << ";" << ab6[0] << ")" << endl;

Реализация ф-ий:

void НаимКвЛин(double\* х, double\* у, double\* a, int k, bool t) {

double t1 = 0;

double x = 0;

double x2 = 0;

double x3 = 0;

double x4 = 0;

double y = 0;

double xy = 0;

double x2y = 0;

for (size\_t i = 0; i < k; i++)

{

x += х[i];

x2 += х[i] \* х[i];

x3 += х[i] \* х[i] \* х[i];

x4 += х[i] \* х[i] \* х[i] \* х[i];

y += у[i];

xy += у[i] \* х[i];

x2y += у[i] \* х[i] \* х[i];

}

if (t) {

a[1] = (k \* xy - x \* y) / (k \* x2 - x \* x);

a[0] = (y - a[1] \* x) / k;

for (size\_t i = 0; i < k; i++)

{

t1 += pow(х[i] \* a[1] + a[0] - у[i],2);

cout << х[i] << " " << у[i] << " " << х[i] \* a[1] + a[0] << " " << pow(х[i] \* a[1] + a[0] - у[i], 2) << "\n";

}

}

else {

a[0] = (y \* x2 \* x4 + x \* x3 \* x2y + xy \* x3 \* x2 - x2 \* x2 \* x2y - x \* xy \* x4 - x3 \* x3 \* y) / (k \* x2 \* x4 + x \* x3 \* x2 + x \* x3 \* x2 - x2 \* x2 \* x2 - x \* x \* x4 - x3 \* x3 \* k);

a[1] = (k \* xy \* x4 + y \* x3 \* x2 + x \* x2y \* x2 - x2 \* xy \* x2 - y \* x \* x4 - x3 \* x2y \* k) / (k \* x2 \* x4 + x \* x3 \* x2 + x \* x3 \* x2 - x2 \* x2 \* x2 - x \* x \* x4 - x3 \* x3 \* k);

a[2] = (k \* x2 \* x2y + x \* xy \* x2 + x \* x3 \* y - y \* x2 \* x2 - x \* x \* x2y - xy \* x3 \* k) / (k \* x2 \* x4 + x \* x3 \* x2 + x \* x3 \* x2 - x2 \* x2 \* x2 - x \* x \* x4 - x3 \* x3 \* k);

for (size\_t i = 0; i < k; i++)

{

t1 += pow(х[i] \* х[i] \* a[2] + х[i] \* a[1] + a[0] - у[i],2);

cout << х[i] << " " << у[i] << " " << х[i] \* х[i] \* a[2] + х[i] \* a[1] + a[0] << " " << pow(х[i] \* х[i] \* a[2] + х[i] \* a[1] + a[0] - у[i], 2) << "\n";

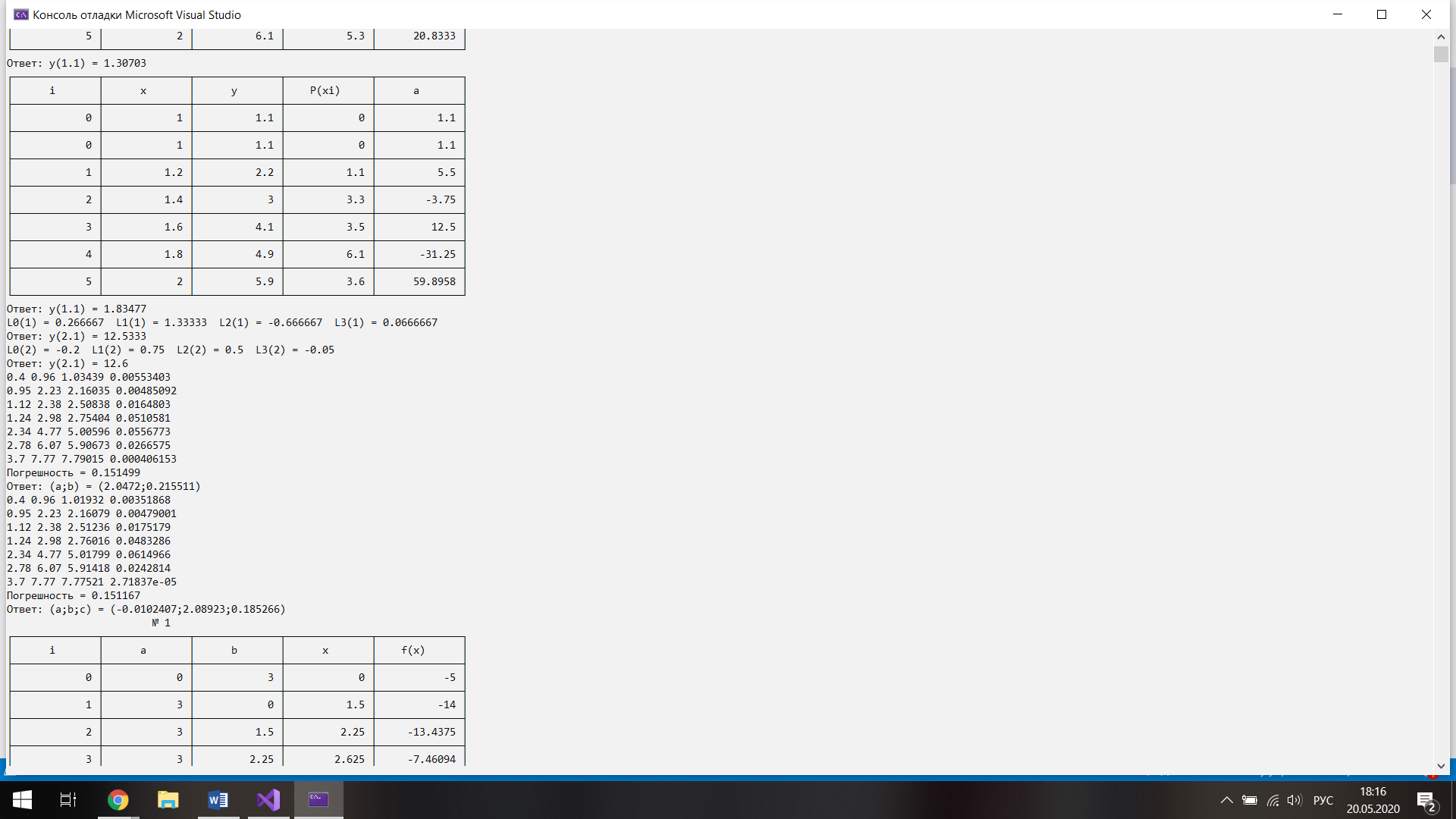
}

}

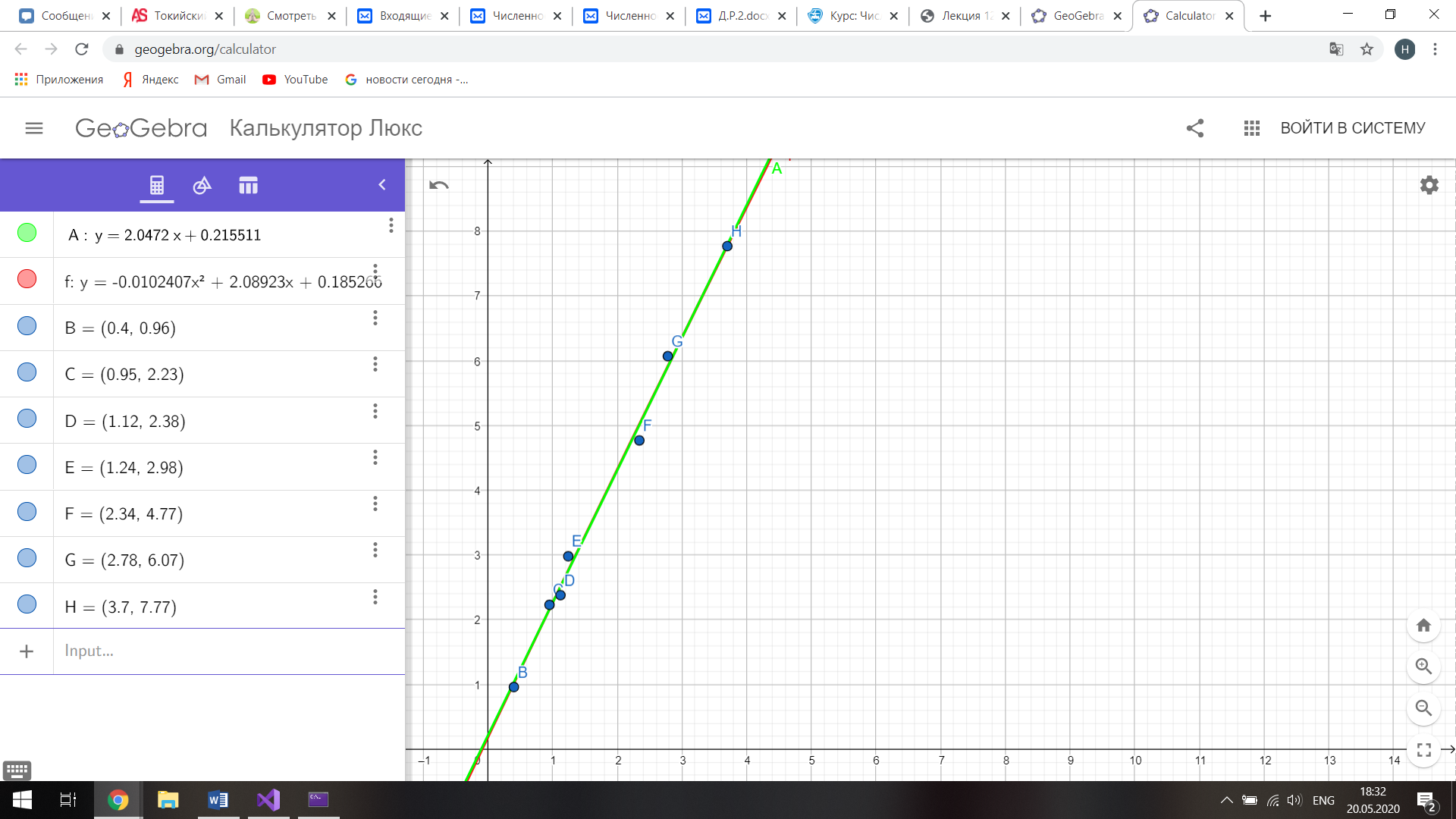
cout << "Погрешность = "<< pow((t1/k),0.5) <<"\n";

}

### Результат работы программы



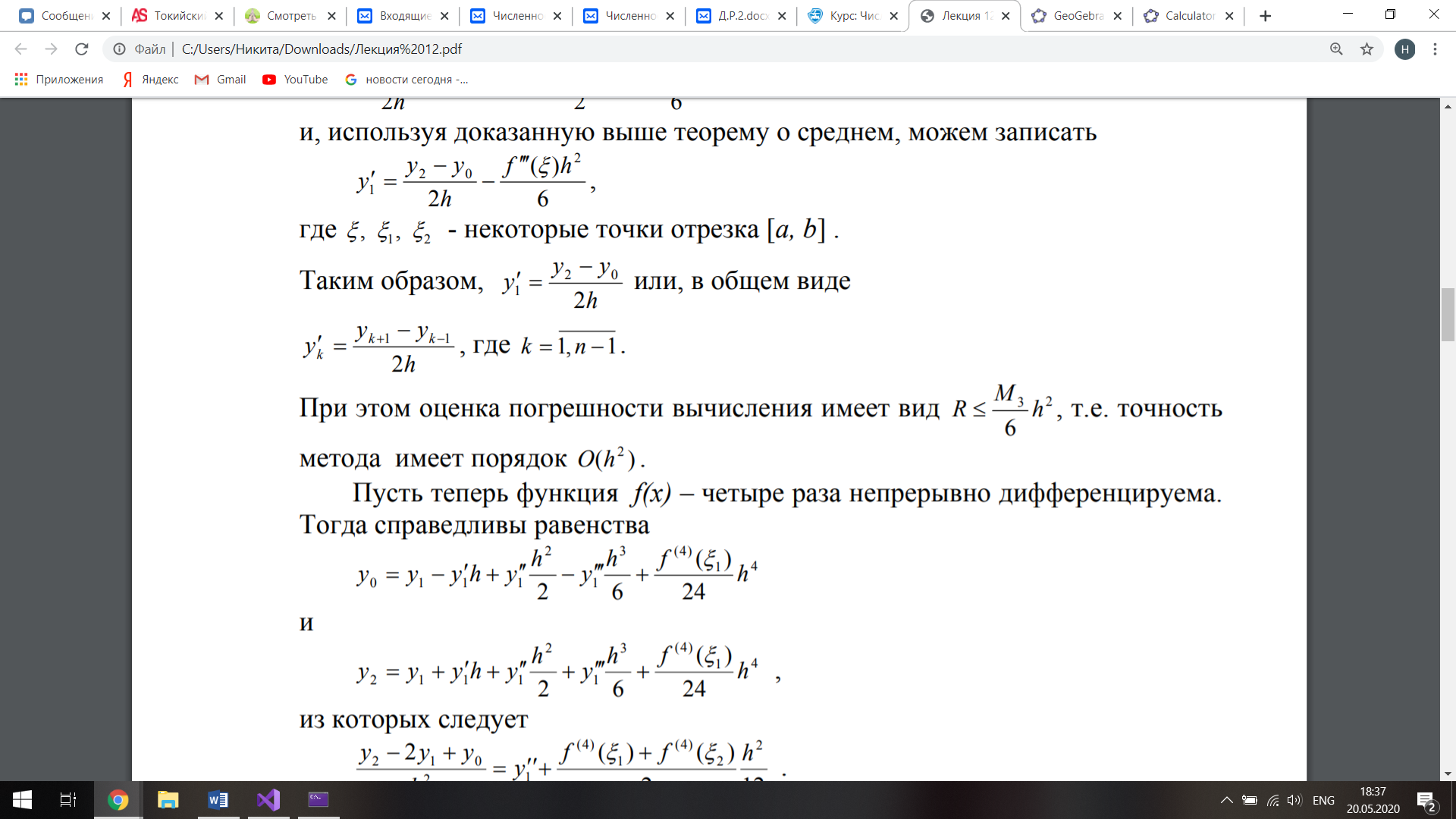
### Прямая и кривая в той же системе координат, где нанесены данные точки



## Задача 2.

Найти 1-ю и 2-ю производную для функции заданной в виде таблицы в точке

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | -1 | 0 | 1 | 2 |
| Y | 0 | 1 | 2 | 3 |



Y,(0.5) = = 1

|  |  |  |  |
| --- | --- | --- | --- |
| X | -0.5 | 0.5 | 1.5 |
| Y, | 1 | 1 | 1 |

Y,,(0.5) = = 0

Ответ Y,(0.5) = 1, Y,,(0.5) = 0.