**sinМИНОБРНАУКИ РОССИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»  
(БГТУ им. В.Г.Шухова)**

**Кафедра программного обеспечения вычислительной техники и автоматизированных систем**

Лабораторная работа №5

Дисциплина: Вычислительная математика

по теме «Решение системы дух нелинейных уравнений с двумя неизвестными методом Ньютона»

Выполнил: ст. группы ВТ-22  
Макаров Даниил Сергеевич

Проверила: Бондаренко Т.В.

**Белгород 2018**

**Лабораторная работа №5**

**Решение системы дух нелинейных уравнений с двумя неизвестными методом Ньютона**

**Цель работы:** изучение и получение практических навыков приближенного решения систем двух нелинейных уравнений с двумя неизвестными методом Ньютона.

**Задания к работе**

1. Записать для уравнений системы соответствующего варианта задания функции *F(x, y)* и *Φ(x, y)*. Построить в одной системе координат графики функций *F(x, y)* и *Φ(x, y)*. Найти точки пересечения графиков функций *F(x, y)* и *Φ(x, y),* которые соответствуют решениям системы уравнений.

2. Определить область содержащую одну из точек пересечения графиков функций *F(x, y)* и *Φ(x, y)*. Выбрать начальное приближение решения системы уравнений ― точку М0 с 41 координатами *(x0; y0)*, принадлежащую выбранной области.

3. Найти частные производные первого порядка по переменным x, y для функций *F(x, y)* и *Φ(x, y)*и вычислить значения производных в точке М0. Записать линеаризованную систему, соответствующую исходной нелинейной системе, для выбранного начального приближения *М0(x0; y0).*

4. Выполнить один шаг численного метода решения системы двух нелинейных уравнений с двумя неизвестными «вручную» и найти следующее приближение к решению системы уравнений (x1; y1). Выполнить проверку правила остановки с точностью ε=0,001.

5. Описать в модуле логическую функцию для приближенного решения системы двух нелинейных уравнений с двумя неизвестными методом Ньютона с заданной точностью ε.

Входными данными являются:

- функции уравнений системы и их частные производные

*f x, y, f / x, f / y,gx, y, g / x, g / y*;

- начальное приближение *(x0; y0);*

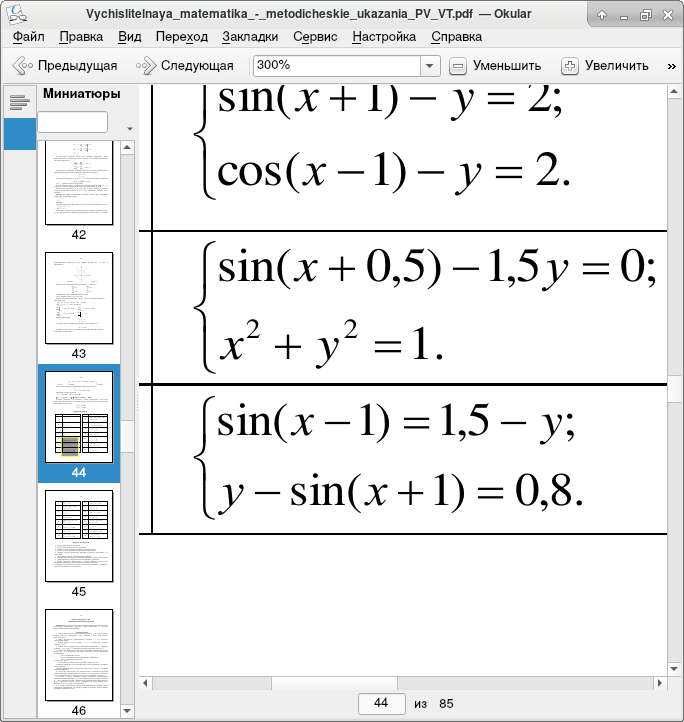
- точность решения ε;

- максимальное число итераций n.

Функция возвращает значение «истина», если приближенное решение системы с заданной точностью получено за число итераций, не превышающее n, при этом вычисленная пара *(x; y)* – приближенное решение системы двух уравнений с точность решения ε. В противном случае функция возвращает значение «ложь».

6. Составить программу для решения системы двух нелинейных уравнений соответствующего варианта задания.

**Задание варианта 7**



1.

Точки пересечения графиков (0,7047, 1,7911)+2\*I\*pi где I принадлежит R

(2,437, 0,509 )+2\*J\*pi где J принадлежит R

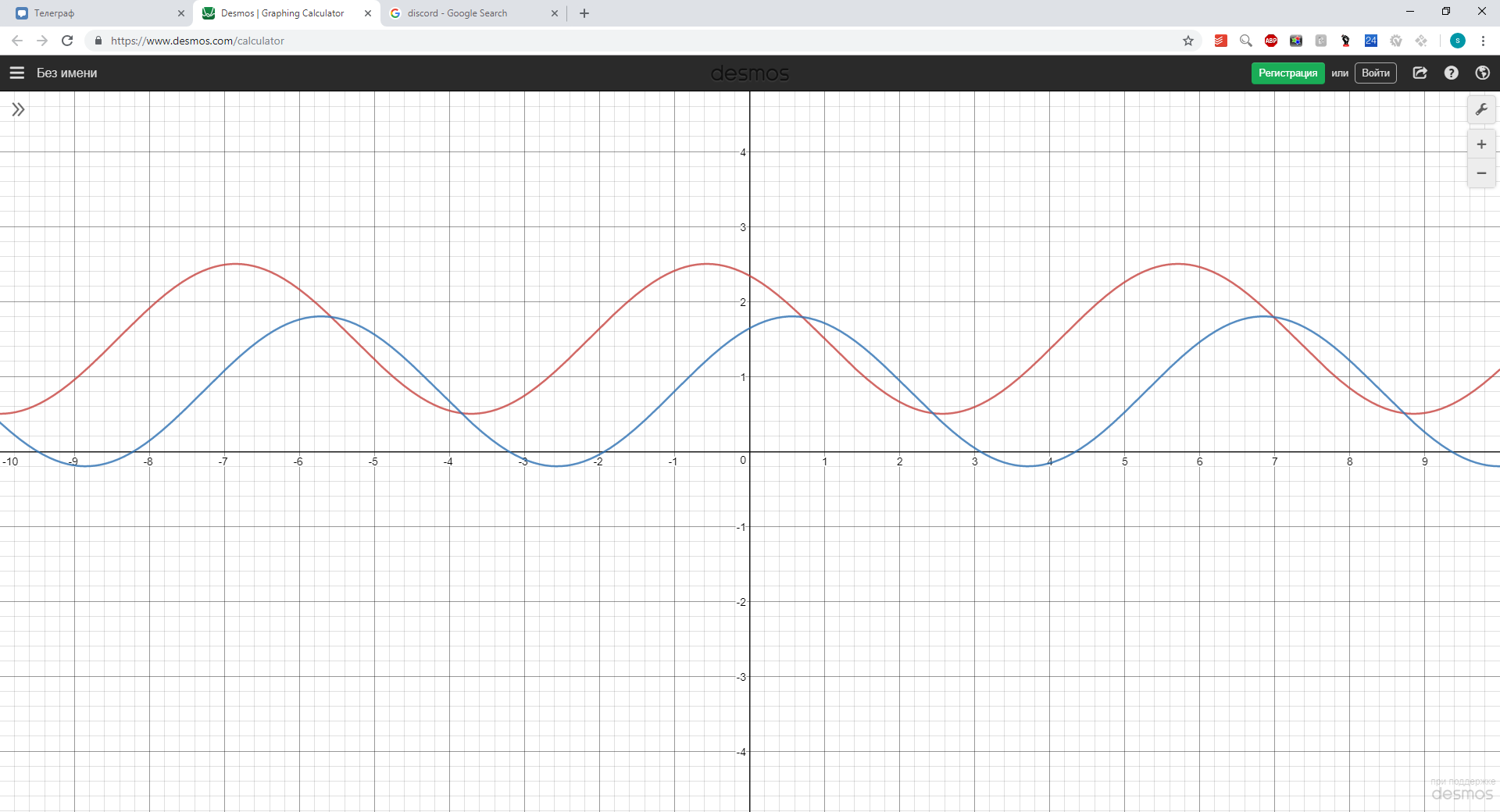


График функций Ф и F

2. Области х пренадлежит [0.5;1]; y пренадлежит [1,5;2]

M0(1;2)

3. Частные производные первого порядка

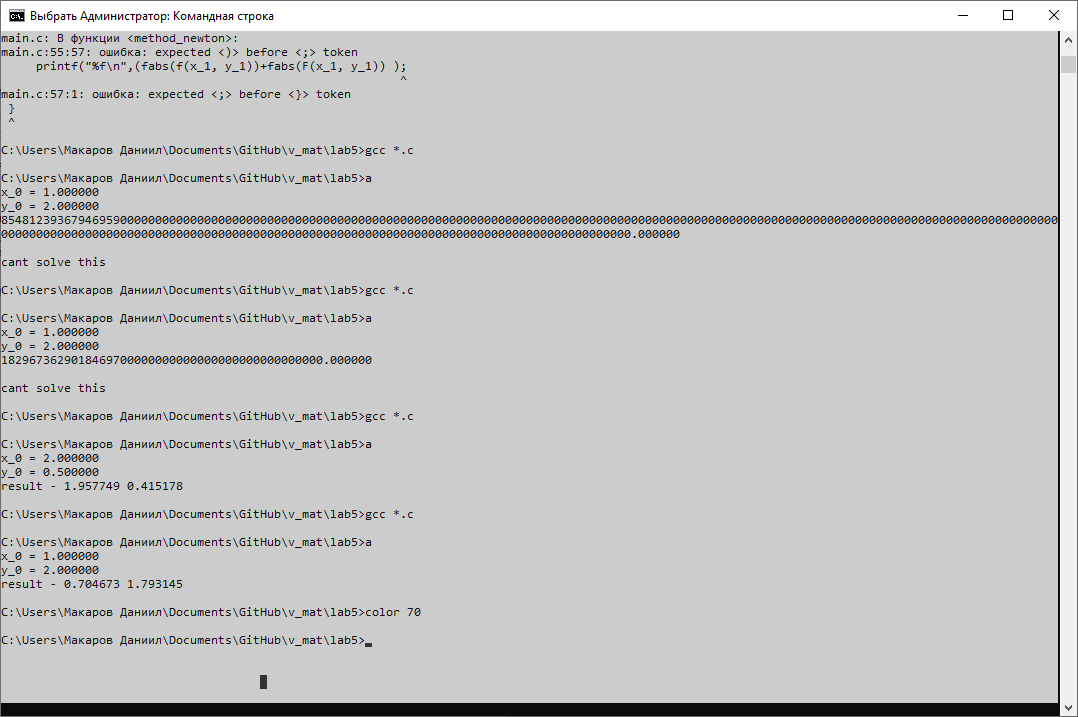
Значение производных в точке М0

Значение функций в точке М0

Решение системы .

.36

**Результат работы программы**



**Исходный код программы**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

typedef double (\*func)(double, double);

double f(double x,double y){

return 1.5-sin(x-1)-y;

}

double df\_dx(double x,double y){

return -cos(x-1);

}

double df\_dy(double x,double y){

return -1;

}

double F(double x,double y){

return 0.8+sin(x+1)-y;

}

double dF\_dx(double x,double y){

return cos(x+1);

}

double dF\_dy(double x,double y){

return -1;

}

int method\_newton(func f,func F,func df\_dx,func df\_dy,func dF\_dx,func dF\_dy,double eps,unsigned count,double x\_0,double y\_0,double \*r\_x,double \*r\_y){

int i=0;

double x\_1,y\_1,det,det1,det2,h,l;

while(i<=count){

double k[2][3]={{df\_dx(x\_0, y\_0),df\_dy(x\_0, y\_0),-f(x\_0, y\_0)},

{dF\_dx(x\_0, y\_0),dF\_dy(x\_0, y\_0),-F(x\_0, y\_0)}

};

det = k[0][0] \* k[1][1] - k[0][1] \* k[1][0];

if (det == 0) return 0;

det1 = k[0][2] \* k[1][1] - k[0][1] \* k[1][2];

det2 = k[0][0] \* k[1][2] - k[0][2] \* k[1][0];

h = det1/det;

l = det2/det;

x\_1 = x\_0+h;

y\_1 = y\_0+l;

if (fabs(f(x\_1, y\_1))+fabs(F(x\_1, y\_1)) < eps){

\*r\_x = x\_1;

\*r\_y = y\_0;

return 1;

}

x\_0 = x\_1;

y\_0 = y\_1;

i++;

}

printf("%f\n",(fabs(f(x\_1, y\_1))+fabs(F(x\_1, y\_1))));

return 0;

}

int main(){

double x\_0,y\_0,x,y;

x\_0=1.0;

y\_0=2.0;

printf("x\_0 = %f\ny\_0 = %f\n",x\_0,y\_0);

if(method\_newton(f,F,df\_dx,df\_dy,dF\_dx,dF\_dy,0.001,200,x\_0,y\_0,&x,&y)){

printf("result - %f %f\n",x,y);

}else{

printf("\ncant solve this\n");

}

return 0;

}