МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Нижегородский государственный технический университет

им. Р.Е. Алексеева» (НГТУ)

Кафедра: «Цифровая экономика»

Дисциплина: «Численные методы»

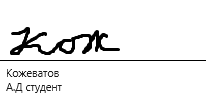
**Лабораторная работа №4**

**«Численные Методы решений нелинейных уравнений Вариант 7»**

Выполнил:

студент 3-го курса группы 21-САИ

Кожеватов Алексей Дмитриевич



Проверил:

д.ф.м.н., проф. Катаева Лилия Юрьевна

11.10.2023

Подпись преподавателя:\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород, 2023

# Содержание

[Содержание 2](#_Toc158425986)

[Постановка задачи 4](#_Toc158425987)

[Работа 1 4](#_Toc158425988)

[Работа 2 4](#_Toc158425989)

[Работа 3 4](#_Toc158425990)

[Работа 4 4](#_Toc158425991)

[Работа 5 4](#_Toc158425992)

[Работа 6 4](#_Toc158425993)

[Описание методов и инструментов, используемых для решения задачи. 5](#_Toc158425994)

[Работа 1 5](#_Toc158425995)

[Работа 2 5](#_Toc158425996)

[Работа 3 6](#_Toc158425997)

[Работа 4 6](#_Toc158425998)

[Работа 5 7](#_Toc158425999)

[Работа 6 7](#_Toc158426000)

[Таблица идентификаторов переменных, используемых для решения задачи 9](#_Toc158426001)

[Работа 1 9](#_Toc158426002)

[Работа 2 9](#_Toc158426003)

[Работа 3 10](#_Toc158426004)

[Работа 4 10](#_Toc158426005)

[Работа 5 11](#_Toc158426006)

[Работа 6 12](#_Toc158426007)

[Ручной счёт 12](#_Toc158426008)

[Работа 1 12](#_Toc158426009)

[Работа 2 15](#_Toc158426010)

[Работа 3 17](#_Toc158426011)

[Работа 4 18](#_Toc158426012)

[Работа 5 20](#_Toc158426013)

[Работа 6 21](#_Toc158426014)

[Реализация задачи в Microsoft Office Excel 2013 24](#_Toc158426015)

[Работа 1 24](#_Toc158426016)

[Работа 2 25](#_Toc158426017)

[Работа 3 27](#_Toc158426018)

[Работа 4 28](#_Toc158426019)

[Работа 5 29](#_Toc158426020)

[Работа 6 30](#_Toc158426021)

[Реализация задачи в Mathcad 15 31](#_Toc158426022)

[Работа 1 31](#_Toc158426023)

[Работа 2 33](#_Toc158426024)

[Работа 3 34](#_Toc158426025)

[Работа 4 35](#_Toc158426026)

[Работа 5 36](#_Toc158426027)

[Работа 6 37](#_Toc158426028)

[Реализация в С++ 38](#_Toc158426029)

[Работа 1 38](#_Toc158426030)

[Работа 2 43](#_Toc158426031)

[Работа 3 48](#_Toc158426032)

[Работа 4 52](#_Toc158426033)

[Работа 5 55](#_Toc158426034)

[Работа 6 59](#_Toc158426035)

[Реализация в Java 63](#_Toc158426036)

[Работа 1 63](#_Toc158426037)

[Работа 2 69](#_Toc158426038)

[Работа 3 72](#_Toc158426039)

[Работа 4 75](#_Toc158426040)

[Работа 5 78](#_Toc158426041)

[Работа 6 81](#_Toc158426042)

[Результаты и их анализ 85](#_Toc158426043)

[Работа 1 85](#_Toc158426044)

[Работа 2 86](#_Toc158426045)

[Работа 3 87](#_Toc158426046)

[Работа 4 87](#_Toc158426047)

[Работа 5 87](#_Toc158426048)

[Работа 6 88](#_Toc158426049)

[Вывод 89](#_Toc158426050)

[Список литературы 89](#_Toc158426051)

[Приложение 2 90](#_Toc158426052)

# Постановка задачи

## Работа 1

1. Отделить корни аналитически:
2. Отделить корни аналитически и уточнить один из них методом проб с точностью до 0,01:
3. Отделить корни графически:
4. Отделить корни графически и уточнить один из них методом проб с точностью до 0,01:

## Работа 2

1. Отделить корни уравнения графически и уточнить один из них методом хорд с точностью 0,001:
2. Отделить корни уравнения аналитически и уточнить один из них методом хорд с точностью 0,001:

## Работа 3

1. Отделить корни уравнения графически и уточнить один из них методом касательных с точностью 0,001:
2. Отделить корни уравнения аналитически и уточнить один из них методом касательных с точностью 0,001:

## Работа 4

Комбинированным методом хорд и касательных решить уравнение третьей степени, вычислив корни с точностью до 0,001:

## Работа 5

1. Отделить корни уравнения графически и уточнить один из них методом итераций с точностью до 0,001:
2. Отделить корни уравнения аналитически и уточнить один из них методом итераций с точностью до 0,001:

## Работа 6

1. Используя метод итераций, решить систему нелинейных уравнений с точностью до 0,001:
2. Используя метод Ньютона, решить систему нелинейных уравнений с точностью 0,001:

# Описание методов и инструментов, используемых для решения задачи.

## Работа 1

Задание 1:

Обозначить функцию f(x), найти её производную, а затем вычислить корень производной. Составить таблицу знаков функции f(x), полагая х равным: а) критическим значениям функции (корням производной) или близким к ним; б) граничным значениям (исходя из области допустимых значений неизвестного). Определить, какие корни имеет уравнение. Чтобы завершить операцию отделения корней, следует уменьшить промежутки, содержащие корни, так чтобы их длина была не больше 1. Для этого составим новую таблицу знаков функций. Найти промежутки, в которых заключены корни.

Задание 2:

Обозначить функцию f(x), найти её производную, а затем вычислить корень производной. Составить таблицу знаков функции f(x), уменьшить промежутки. Уточнить один из корней методом проб до сотых долей. Уточнение корня методом проб заключается в том, что путем последовательного деления отрезок, на котором отделен корень, уменьшается до тех пор, пока длина его не сделается меньше, чем заданная точность.

Задание 3:

Делим уравнение на две части, строим графики обеих функций и ищем по ним корни уравнения.

Задание 4:

Делим уравнение на две части, строим графики обеих функций и ищем по ним корни уравнения. Для уточнения методом проб выберем промежуток, на концах которого функция имеет разные знаки. Составляем таблицу. Для удобства расчетов перейдем к десятичным логарифмам.

## Работа 2

Метод хорд, или метод секущих, является итерационным численным методом для нахождения приближенных значений корня уравнения. Он оценивает следующее приближение путем проведения прямой через две точки на кривой.

Задание1:

Отделим корень графически. Построим графики функций, составим таблицы значений этих функций. Для уточнения корня методом хорд, определим знаки функции на концах промежутка и знак второй производной в этом промежутке. Для вычисления применять формулу (1)

Задание 2:

Отделить корни аналитически. Составить таблицу знаков функции f(x). Чтобы уточнить корень, находим вторую производную в промежутке, где вторая производная меньше нуля. Для вычислений применяем формулу (2)

## Работа 3

Задание 1:

Отделим корень графически. Построим графики функций, составим таблицы значений этих функций. Для уточнения методом касательных надо определить знаки функции на этих промежутках и знак второй производной. Метод касательных, использует линейную аппроксимацию функции в окрестности предполагаемого корня для нахождения более точного следующего приближения. За начальное приближение берем максимальное значение промежутка, где функция меньше нуля. Для вычислений используем формулу (3)

Заносим результаты в таблицу и ищем значения с разницей меньше 0,001

Задание 2:

Отделить корни аналитически. Составить таблицу знаков функции f(x). Чтобы уточнить корень, находим вторую производную в промежутке, где вторая производная меньше нуля. Для вычислений применяем формулу (4)

## Работа 4

Отделим корни аналитически и составим таблицу знаков функции f(x). Уменьшим промежутки, содержащие корни до длины, равной 1. Уточним корни комбинированным методом хорд и касательных. Для расчетов применяем формулы (5) и (6).

Где – значения корня соответственно по недостатку и избытку.

Все вычисления производим в таблице, обозначив формулы (7) и (8)

Для последнего промежутка используем формулы (9) и (10)

Все вычисления производим в таблице, обозначив формулы (11) и (12)

## Работа 5

Метод итераций, или метод последовательных приближений, предполагает выбор начального приближения и последующее вычисление следующего приближения на основе предыдущего.

Задание 1:

Найдем приближенные значения корней графически. Для этого делим уравнение на две части, и по графику ищем корни. Для уточнения его методом итераций приведем уравнение к виду x=φ(x). Функцию φ(x) будем искать из соотношения , считая, что , где Q=max и число k имеет тот же знак, что и в промежутке. За начальное приближение возьмем , все остальные приближения будем определять из равенства (13)

Вычисления располагаем в таблице

Задание 2:

Отделяем корни аналитически, находим производную, составляем таблицу. Приводим уравнение к виду x=φ(x), так чтобы . Пусть , тогда приближения считаем по формуле (14) Вычисления располагаем в таблицу.

## Работа 6

Задание 1:

Перепишем данную систему, выразив x и y. Отделение корней производим графически. Убедимся в том, что метод итераций применим для уточнения решения системы, для чего запишем ее в следующем виде: Проверяем условия сходимости. Вычисляем данную систему, за начальные приближения принимаем .

Задание 2:

Отделение корней производим графически. Для построения графиков функций составляем таблицу значений функций , входящих в первое и второе уравнение. Находим решения системы и уточняем одно из них. За начальное приближение берем значения . Метод Ньютона использует производные функции для приближенного решения уравнений. Он сходится быстрее, чем многие другие методы, но может быть более сложен в реализации. Уточнение корней проводим методом Ньютона в таблице до нахождения разницы с необходимым уточнением. Используем формулу (15). Вычисления производим в таблице.

Где ,

# Таблица идентификаторов переменных, используемых для решения задачи

## Работа 1

В таблице 1 представлены идентификаторы переменных, используемых для решения работы 1.

Таблица 1

Таблица идентификаторов переменных работы 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| f(x) | f(x) | f(x) | f(x) | f(x) | Исходное нелинейное уравнение |
|  |  |  |  |  | Первая производная |
| signf(x) |  |  |  |  | Знак функции в заданной точке |
| a | a | a | a | a | Левая граница интервала |
| b | b | b | b | b | Правая граница интервала |

## Работа 2

В таблице 2 представлены идентификаторы переменных, используемых для решения работы 2.

Таблица 2

Таблица идентификаторов переменных работы 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| f(x) | f(x) | f(x) | f(x) | f(x) | Исходное нелинейное уравнение |
|  |  |  |  |  | Первая производная |
|  |  |  |  |  | Вторая производная |
| signf(x) |  |  |  |  | Знак функции в заданной точке |
| a | a | a | a | a | Левая граница интервала |
| b | b | b | b | b | Правая граница интервала |
| h | h | h | h | h | Шаг |

## Работа 3

В таблице 3 представлены идентификаторы переменных, используемых для решения работы 3.

Таблица 3

Таблица идентификаторов переменных работы 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| f(x) | f(x) | f(x) | f(x) | f(x) | Исходное нелинейное уравнение |
|  |  |  |  |  | Первая производная |
|  |  |  |  |  | Вторая производная |
| signf(x) |  |  |  |  | Знак функции в заданной точке |
| a | a | a | a | a | Левая граница интервала |
| b | b | b | b | b | Правая граница интервала |
| h | h | h | h | h | Шаг |

## Работа 4

В таблице 4 представлены идентификаторы переменных, используемых для решения работы 4.

Таблица 4

Таблица идентификаторов переменных работы 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| f(x) | f(x) | f(x) | f(x) | f(x) | Исходное нелинейное уравнение |
|  |  |  |  |  | Первая производная |
|  |  |  |  |  | Вторая производная |
| a | a | a | a | a | Левая граница интервала |
| b | b | b | b | b | Правая граница интервала |
| h | h | h | h | h | Шаг |

## Работа 5

В таблице 5 представлены идентификаторы переменных, используемых для решения работы 5.

Таблица 5

Таблица идентификаторов переменных работы 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| f(x) | f(x) | f(x) | f(x) | f(x) | Исходное нелинейное уравнение |
|  |  |  |  |  | Первая производная |
|  |  |  |  |  | Вторая производная |
| a | a | a | a | a | Левая граница интервала |
| b | b | b | b | b | Правая граница интервала |
| h | h | h | h | h | Шаг |
| Q | Q | Q | Q | Q | Функция для поиска k |
| k | k | k | k | k | Число для поиска |

## Работа 6

В таблице 6 представлены идентификаторы переменных, используемых для решения работы 6.

Таблица 6

Таблица идентификаторов переменных работы 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Математические  обозначения | Excel | Mathcad 15 | C++ | Java | Комментарий |
| F(x,y), G(x,y) | F(x,y), G(x,y) | F(x,y), G(x,y) | F(x,y), G(x,y) | F(x,y), G(x,y) | Исходная система нелинейных уравнений |
|  |  |  |  |  | Первая производная |
|  |  |  |  |  | Приближение для уточнения корней |
|  |  |  |  |  | Приближение для уточнения корня x |
|  |  |  |  |  | Приближение для уточнения корня y |

# Ручной счёт

## Работа 1

Задание 1:

x=0 критическая точка

|  |  |  |  |
| --- | --- | --- | --- |
| x | -*ꭃ* | 0 | +*ꭃ* |
| signf(x) | + | - | + |

Так как происходят две перемены знака функции, то уравнение имеет два действительных корня. Уменьшим промежутки до длины не больше 1.

|  |  |  |  |
| --- | --- | --- | --- |
| x | -*1* | 0 | +*1* |
| signf(x) | + | - | + |

Корни заключены в промежутках:

Задание 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | -*ꭃ* | -4 | 0 | 1 | +*ꭃ* |
| signf(x) | + | - | + | - | + |

Из таблицы видно, что уравнение имеет 4 действительных корня, заключенных в промежутках:

Уменьшим промежутки.

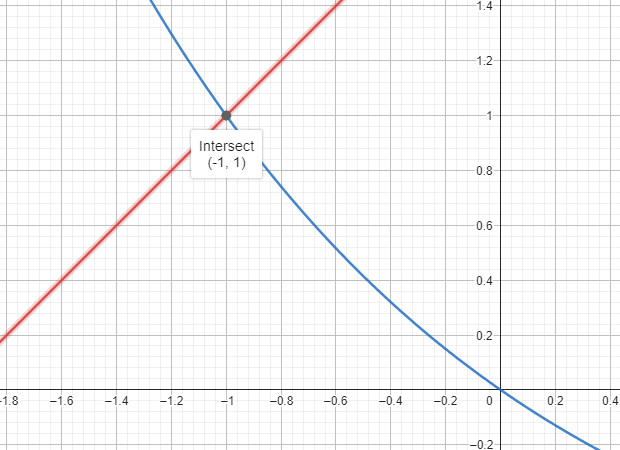
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | -*6* | -4 | 0 | 1 | +*6* |
| signf(x) | + | - | + | - | + |

Уточним один из корней методом проб с точностью до 0,001. Все вычисления произведем, используя следующую таблицу.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **a** | **b** | **x** | **F(a)** | **F(x)** | **F(a)\*F(x)** | **|F(x)|<e** |
| 0 | -6 | -4 | -5 | 127 | -92 | -11684 |  |
| 1 | -6 | -5 | -5,5 | 127 | -9,4375 | -1198,56 |  |
| 2 | -6 | -5,5 | -5,75 | 127 | 51,19141 | 6501,309 |  |
| 3 | -5,75 | -5,5 | -5,625 | 51,19141 | 19,09009 | 977,2484 |  |
| 4 | -5,625 | -5,5 | -5,562500 | 19,09009 | 4,393082 | 83,86432 |  |
| 5 | -5,5625 | -5,5 | -5,531250 | 4,393082 | -2,62884 | -11,5487 |  |
| 6 | -5,5625 | -5,53125 | -5,546875 | 4,393082 | 0,855252 | 3,757193 |  |
| 7 | -5,54688 | -5,53125 | -5,539063 | 0,855252 | -0,89349 | -0,76416 |  |
| 8 | -5,54688 | -5,53906 | -5,542969 | 0,855252 | -0,02079 | -0,01778 |  |
| 9 | -5,54688 | -5,54297 | -5,544922 | 0,855252 | 0,41681 | 0,356478 |  |
| 10 | -5,54492 | -5,54297 | -5,543945 | 0,41681 | 0,197904 | 0,082488 |  |
| 11 | -5,54395 | -5,54297 | -5,543457 | 0,197904 | 0,088529 | 0,01752 |  |
| 12 | -5,54346 | -5,54297 | -5,543213 | 0,088529 | 0,033862 | 0,002998 |  |
| 13 | -5,54321 | -5,54297 | -5,543091 | 0,033862 | 0,006533 | 0,000221 | корень |

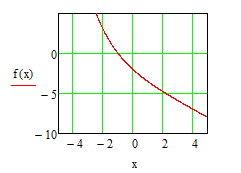
1

Задание 3:

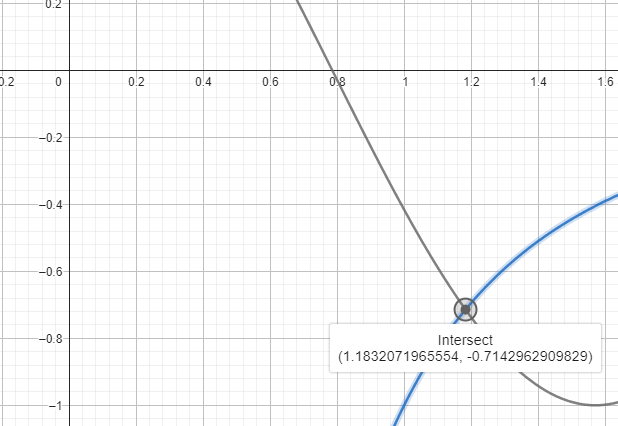
**

Из графика видно, что уравнение имеет один действительный корень:

|  |  |  |  |
| --- | --- | --- | --- |
| x | -*ꭃ* | -1 | *+ꭃ* |
| signf(x) | + | - | + |



Задание 4:



Из графика видно, что уравнение имеет один корень:

|  |  |  |
| --- | --- | --- |
| x | 1,1 | 1,2 |
| signf(x) | - | + |

Уточним его методом проб с точностью до 0,01. Вычисления произведем в таблице.

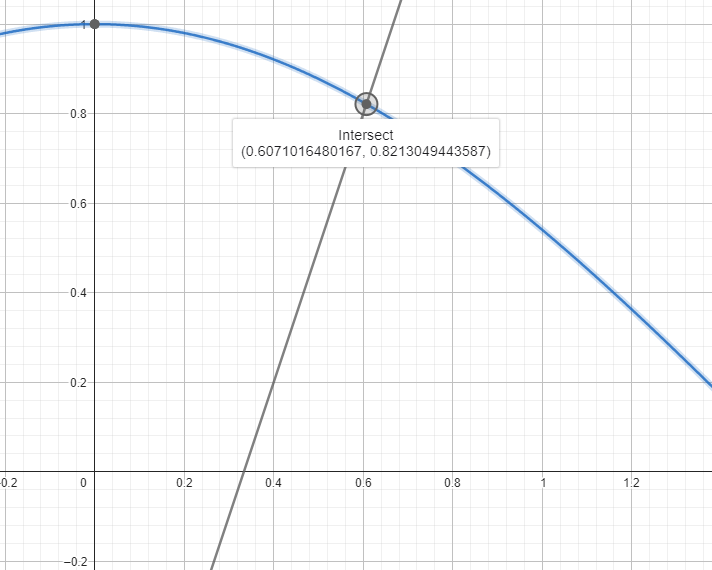
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **a** | **b** | **x** | **F(a)** | **F(x)** | **F(a)\*F(x)** | **|F(x)|<e** |
| 0 | 1,1 | 1,2 | 1,15 | 0,287914 | 0,11885 | 0,034219 |  |
| 1 | 1,15 | 1,2 | 1,175 | 0,11885 | 0,029817 | 0,003544 |  |
| 2 | 1,175 | 1,2 | 1,1875 | 0,029817 | -0,01571 | -0,00047 |  |
| 3 | 1,175 | 1,1875 | 1,18125 | 0,029817 | 0,007136 | 0,000213 | корень |

## Работа 2

Задание 1:

Построим графики функций по таблице.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **x** | 0 | 0,2 | 0,4 | 0,6 | 0,8 | 1 |
| **y1** | -1 | -0,4 | 0,2 | 0,8 | 1,4 | 2 |
| **y2** | 1 | 0,980067 | 0,921061 | 0,82533561 | 0,696707 | 0,540302 |



Таким образом, положительный корень уравнения заключен в промежутке: . Определим знаки функции на концах промежутка.

при x принадлежащим промежутку.

Применим формулу (1) для значений b=0,1;

Вычисления удобно производить в таблице.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **2-xn** | **F(xn)** | **F(2)-F(xn)** | **h** | **|F(x)|<e** |
| 0 | 0,5 | 1,5 | -0,3775826 | 5,7937294 | -0,03259 |  |
| 1 | 0,53258545 | 1,46741455 | -0,2637408 | 5,67988764 | -0,0217 |  |
| 2 | 0,554289 | 1,445711 | -0,1874063 | 5,60355313 | -0,01491 |  |
| 3 | 0,569196 | 1,430804 | -0,134747 | 5,55089383 | -0,01046 |  |
| 4 | 0,579654 | 1,420346 | -0,0976917 | 5,51383856 | -0,00745 |  |
| 5 | 0,587101 | 1,412899 | -0,0712468 | 5,48739361 | -0,00536 |  |
| 6 | 0,592462 | 1,407538 | -0,0521822 | 5,46832907 | -0,00389 |  |
| 7 | 0,596351 | 1,403649 | -0,0383374 | 5,45448422 | -0,00284 |  |
| 8 | 0,599188 | 1,400812 | -0,0282294 | 5,44437625 | -0,00208 |  |
| 9 | 0,601266 | 1,398734 | -0,0208209 | 5,4369677 | -0,00153 |  |
| 10 | 0,602793 | 1,397207 | -0,0153753 | 5,43152211 | -0,00112 |  |
| 11 | 0,603918 | 1,396082 | -0,0113641 | 5,42751094 | -0,00083 |  |
| 12 | 0,604747 | 1,395253 | -0,0084049 | 5,42455177 | -0,00061 |  |
| 13 | 0,605359 | 1,394641 | -0,0062193 | 5,42236618 | -0,00045 |  |
| 14 | 0,605812 | 1,394188 | -0,0046037 | 5,42075058 | -0,00033 |  |
| 15 | 0,606147 | 1,393853 | -0,0034087 | 5,41955557 | -0,00025 |  |
| 16 | 0,606395 | 1,393605 | -0,0025244 | 5,41867125 | -0,00018 |  |
| 17 | 0,606578 | 1,393422 | -0,0018698 | 5,41801662 | -0,00014 |  |
| 18 | 0,606714 | 1,393286 | -0,0013851 | 5,4175319 | -0,0001 |  |
| 19 | 0,606814 | 1,393186 | -0,0010261 | 5,41717292 | -7,4E-05 |  |
| 20 | 0,606889 | 1,393111 | -0,0007602 | 5,41690703 | -5,5E-05 | корень |

Задание 2:

;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | -*ꭃ* |  | 1 | +*ꭃ* |
| signf(x) | - | - | + | + |

Уравнение имеет один действительный корень в промежутке

на промежутке

Применим формулу (2) для значений

Вычисления располагаем в таблице

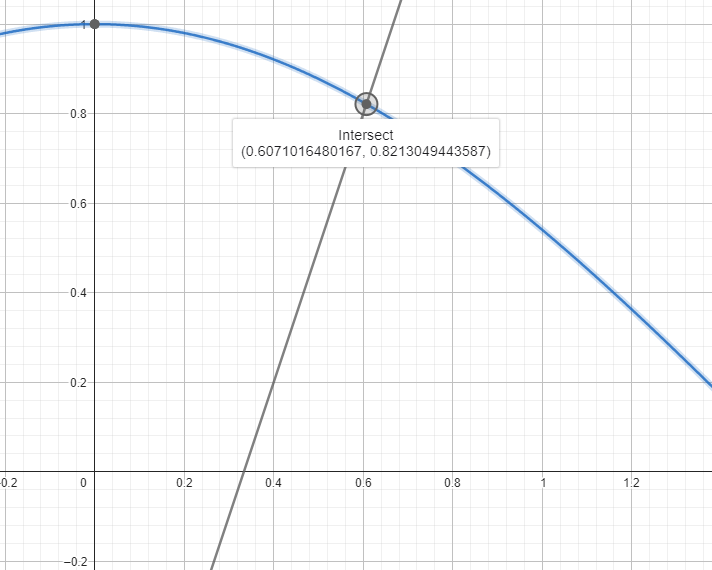
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **F(xn)** | **F(xn)-F(a)** | **xn-a** | **h** | **|F(x)|<e** |
| 0 | 1 | 0,5 | 1,275 | 0,5 | -0,30392 |  |
| 1 | 0,803922 | -0,1492149 | 0,6257851 | 0,30392157 | -0,37639 |  |
| 2 | 0,876390 | 0,06492622 | 0,8399262 | 0,37638991 | -0,3473 |  |
| 3 | 0,847295 | -0,0244902 | 0,7505098 | 0,34729501 | -0,35863 |  |
| 4 | 0,858628 | 0,00977821 | 0,7847782 | 0,35862772 | -0,35416 |  |
| 5 | 0,854159 | -0,0038183 | 0,7711817 | 0,35415928 | -0,35591 |  |
| 6 | 0,855913 | 0,00150414 | 0,7765041 | 0,35591282 | -0,35522 |  |
| 7 | 0,855223 | -0,0005905 | 0,7744095 | 0,35522339 | -0,35549 | корень |

## Работа 3

Задание 1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **x** | 0 | 0,2 | 0,4 | 0,6 | 0,8 | 1 |
| **y1** | -1 | -0,4 | 0,2 | 0,8 | 1,4 | 2 |
| **y2** | 1 | 0,980067 | 0,921061 | 0,82533561 | 0,696707 | 0,540302 |

Построим графики функций по таблице.



Таким образом, положительный корень уравнения заключен в промежутке: . Определим знаки функции на концах промежутка.

при x принадлежащим промежутку.

Составим таблицу и по формуле (3) найдем значения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** | **xn** | **F(xn)** | **F(xn)/F'(x0)** | **|F(x)|<e** |
| 0 | 1 | 1,459698 | 0,419525692 |  |
| 1 | 0,580474 | -0,09478 | -0,027240243 |  |
| 2 | 0,607715 | 0,002189 | 0,000628992 |  |
| 3 | 0,607086 | -5,7E-05 | -1,65111E-05 | корень |

Задание 2:

;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | -*ꭃ* |  | 1 | +*ꭃ* |
| signf(x) | - | - | + | + |

Уравнение имеет один действительный корень в промежутке

на промежутке

, то

Составим таблицу и по формуле (4) найдем значения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **F(xn)** | **F'(xn)** | **h** | **|F(x)|<e** |
| 0 | 1 | 0,5 | 3,9 | 0,1282051 |  |
| 1 | 0,871795 | 0,05049 | 3,128796844 | 0,0161371 |  |
| 2 | 0,855658 | 0,000729 | 3,038713745 | 0,0002399 | корень |

## Работа 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | -*ꭃ* | -3 | 0 | +*ꭃ* |
| signf(x) | - | + | - | + |

Из таблицы видно, что уравнение имеет 3 действительных корня, заключенных в промежутках:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | -*4* | -3 | -2 | 1 | 0 | 1 | 2 |
| signf(x) | - | + | + | + | - | - | + |

при

По формулам (5-8) составим таблицу со значениями. За начальные значения примем ,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **x1n-xn** | **f(xn)** | **f'(xn)** | **f(x1n)-f(xn)** | **h1n** | **|F(xn)| <e** |
| **x1n** | **f(x1n)** |  |  | **h2n** |
| 0 | -4 | 1 | -5 | 24 | 11 | -0,20833 |  |
| -3 | 6 |  | -0,45455 |
| 1 | -3,791667 | 0,246212 | -0,63296 | 18,01042 | 3,630703 | -0,03514 |  |
| -3,545455 | 2,997746 |  | -0,04292 |
| 2 | -3,756523 | 0,007779 | -0,0169 | 17,05137 | 0,13183 | -0,00099 |  |
| -3,748743 | 0,114934 |  | -0,001 |
| 3 | -3,755532 | 6,15E-06 | -1,3E-05 | 17,02454 | 0,000105 | -7,8E-07 | корень |
| -3,755526 | 9,14E-05 |  | -7,8E-07 |

при

По формулам (5-8) составим таблицу со значениями. За начальные значения примем ,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **x1n-xn** | **f(xn)** | **f'(xn)** | **f(x1n)-f(xn)** | **h1n** | **|F(xn)| <e** |
|  | **x1n** |  | **f(x1n)** |  |  | **h2n** |
| 0 | -2,5 | 0,5 | 4 | -7,5 | -5 | -0,53333 |  |
| -2 | -1 | -0,4 |
| 1 | -1,96666667 | -0,13333 | -1,40326 | -12,1933 | 1,571259 | 0,115084 |  |
| -2,1 | 0,168 | 0,119077 |
| 2 | -2,0817508 | -0,00399 | -0,04013 | -11,4694 | 0,045743 | 0,003499 |  |
| -2,08574392 |  | 0,00561 | 0,003503 |  |
| 3 | -2,085250 | -4,3E-06 | -4,3E-05 | -11,4449 | 4,88E-05 | 3,74E-06 | корень |
|  | -2,08525418 |  | 6,01E-06 | 3,74E-06 |  |

при

По формулам (9-12) составим таблицу со значениями. За начальные значения примем ,

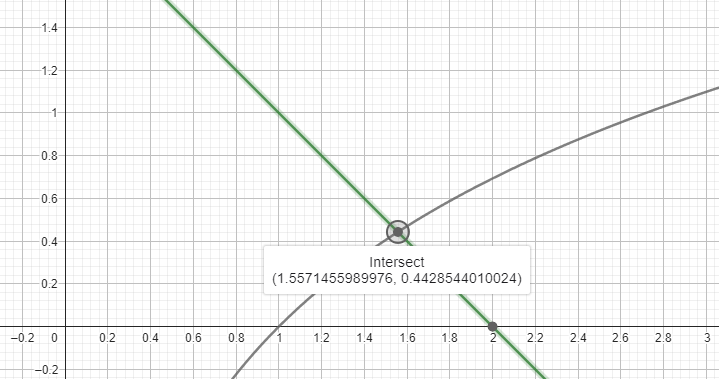
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **x1n-xn** | **f(xn)** | **f'(xn)** | **f(x1n)-f(xn)** | **h1n** | **|F(xn)| <e** |
|  | **x1n** |  | **f(x1n)** |  |  | **h2n** |  |
| 0 | 1 | 1 | -10 | 24 | 41 | -0,41667 |  |
| 2 |  | 31 | -0,2439 |
| 1 | 1,416666667 | -0,17276 | 2,748843 | 37,54167 | -5,97384 | 0,073221 |  |
| 1,243902439 | -3,225 | 0,079497 |
| 2 | 1,343445554 | -0,00628 | 0,093038 | 35,0111 | -0,21905 | 0,002657 |  |
| 1,337169773 | -0,12601 | 0,002666 |  |
| 3 | 1,340788163 | -8,1E-06 | 0,00012 | 34,92046 | -0,00028 | 3,45E-06 | корень |
| 1,340780018 | -0,00016 | 3,45E-06 |  |

## Работа 5

Задание 1:

Построим графики функций по таблице.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **x** | 0,5 | 0,75 | 1 | 1,25 | 1,5 | 1,75 | 2 |
| **y1** | -0,69315 | -0,28768 | 0 | 0,22314355 | 0,405465 | 0,559616 | 0,693147181 |
| **y2** | 1,5 | 1,25 | 1 | 0,75 | 0,5 | 0,25 | 0 |



Из графика видно, что уравнение имеет один действительный корень, заключенный в промежутке

*)*

на промежутке

Возьмем k=2

За начальное приближение возьмем , остальные приближения определим с помощью формулы (13). Результаты занесем в таблицу.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** | **xn** | **F(xn)** | **a(xn)** | **|F(x)|<e** |
| 0 | 0,5 | -2,19315 | 1,596574 |  |
| 1 | 1,596574 | 0,064433 | 1,564357 |  |
| 2 | 1,564357 | 0,011832 | 1,558441 |  |
| 3 | 1,558441 | 0,002127 | 1,557378 |  |
| 4 | 1,557378 | 0,000381 | 1,557187 | корень |

Задание 2:

;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | -*ꭃ* | 1 | 1.1 | +*ꭃ* |
| signf(x) | - | - | + | + |

Уравнение имеет один действительный корень в промежутке

на промежутке

Возьмем k=3

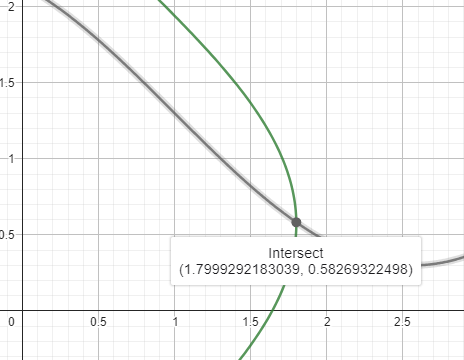
Составим таблицу и по формуле (14) найдем значения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** | **xn** | **F(xn)** | **a(xn)** | **|F(x)|<e** |
| 0 | 1 | -0,2 | 1,066667 |  |
| 1 | 1,066667 | 0,012741 | 1,060296 |  |
| 2 | 1,060296 | -0,00871 | 1,064652 |  |
| 3 | 1,064652 | 0,005931 | 1,061687 |  |
| 4 | 1,061687 | -0,00405 | 1,063712 |  |
| 5 | 1,063712 | 0,00276 | 1,062332 |  |
| 6 | 1,062332 | -0,00188 | 1,063274 |  |
| 7 | 1,063274 | 0,001284 | 1,062632 |  |
| 8 | 1,062632 | -0,00088 | 1,06307 | корень |

## Работа 6

Задание 1:

Перепишем систему в виде и построим графики.



Из графика видно, что система имеет одно решение, заключенное в области D

D: ;

Убедимся в том, что метод итераций применим для решения системы, перепишем ее в виде.

Так как , , то в области D имеем

Условия сходимости выполняются. Вычисления производим по формулам:

За начальное приближение принимаем

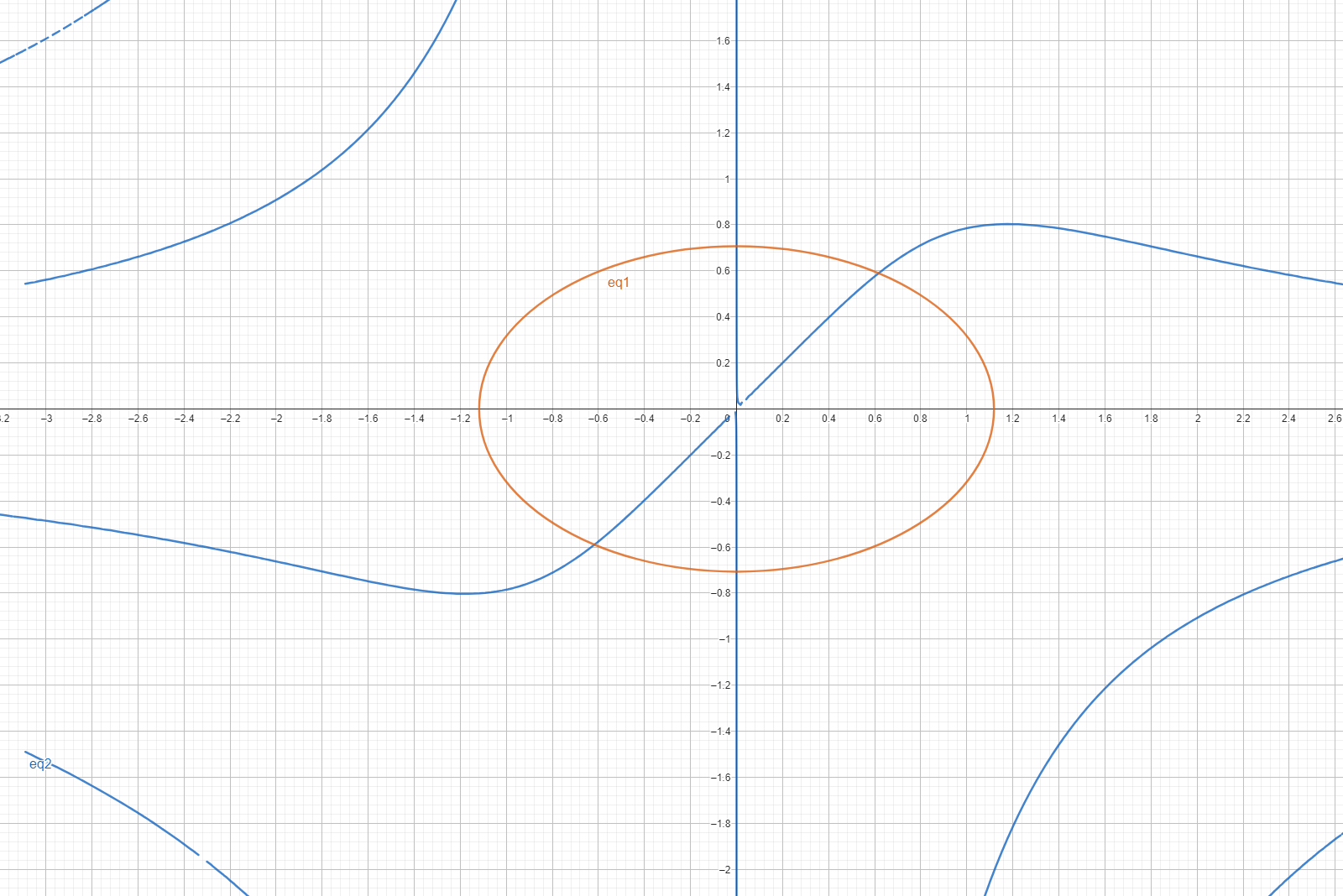
Вычисления удобно располагать в таблице.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **yn** | **y** | **x** | **|F(x)|<e** |
| 0 | 1,75 | 0,55 | 0,61836 | 1,79978 |  |
| 1 | 1,799784 | 0,618361 | 0,58279 | 1,79887 |  |
| 2 | 1,798869 | 0,582795 | 0,58343 | 1,79993 |  |
| 3 | 1,799928 | 0,583432 | 0,58269 | 1,79992 |  |
| 4 | 1,799920 | 0,582694 | 0,58270 | 1,79993 | корень |

,

Задание 2: :

Отделение корней произведем графически.



Система имеет 4 решения. Уточним одно из них, принадлежащее области:, За начальное приближение примем

Имеем:

C помощью формулы (15) находим приближение по методу Ньютона

Вычисления удобно располагать в таблице.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **xn** | **xn^2** | **xn+yn** | **tg(xn\*yn)** | **F(xn,yn)** | **F'x(xn,yn)** | **F'y(xn,yn)** | **delta(n)** | **delta(hn)** | **hn** |
| **yn** | **yn^2** | **G(xn,yn)** | **G'x(xn,yn)** | **G'y(xn,yn)** | **delta(kn)** | **kn** |
| 0 | 0,000000 | 0,000000 | -0,70000 | 0,00000 | 0,00000 | -0,70000 | 0,00000 | 1,96000 | 0,00000 | 0,00000 |
| -0,700000 | 0,490000 | -0,02000 | 0,00000 | -2,80000 | -0,01400 | -0,00714 |
| 1 | 0,000000 | 0,000000 | -0,70714 | 0,00000 | 0,00000 | -0,70714 | 0,00000 | 2,00020 | 0,00000 | 0,00000 |
| -0,707143 | 0,500051 | 0,00010 | 0,00000 | -2,82857 | 0,00007 | 0,00004 |
| 2 | 0,000000 | 0,000000 | -0,70711 | 0,00000 | 0,00000 | -0,70711 | 0,00000 | 2,00000 | 0,00000 | 0,00000 |
| -0,707107 | 0,500000 | 0,00000 | 0,00000 | -2,82843 | 0,00000 | 0,00000 |

,

# Реализация задачи в Microsoft Office Excel 2013

## Работа 1

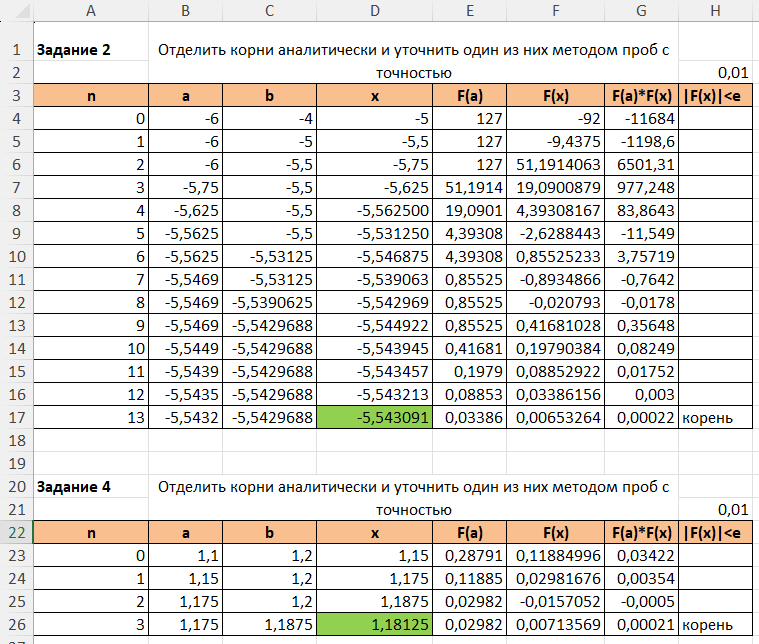


Рисунок 1 - Решение работы 1 без формул

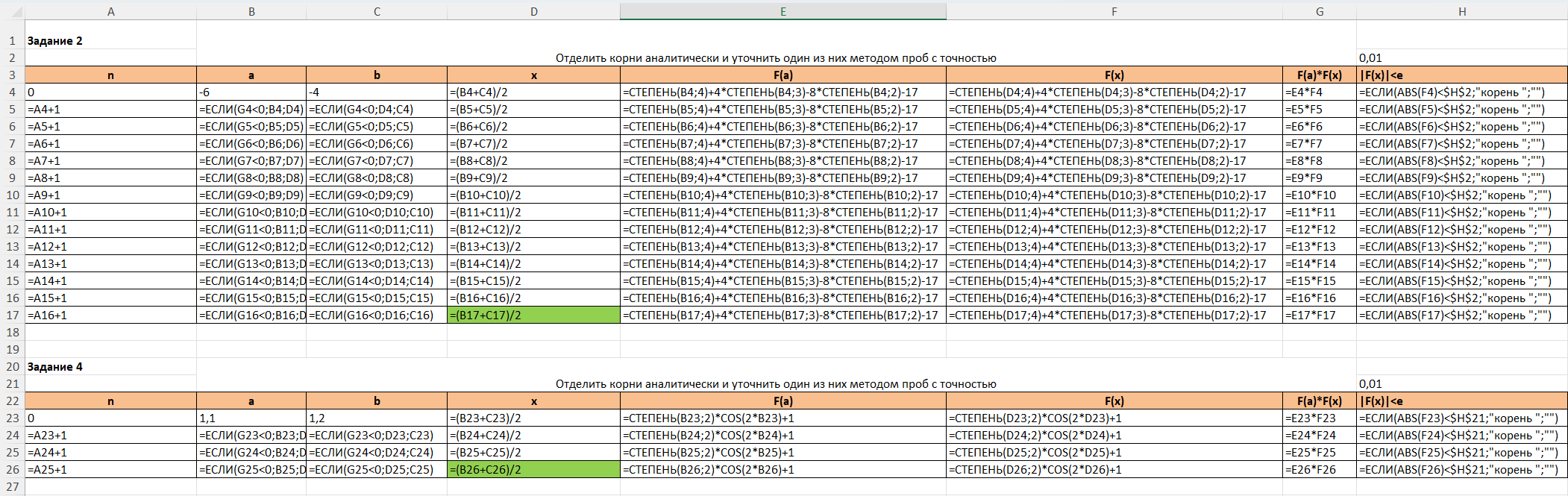
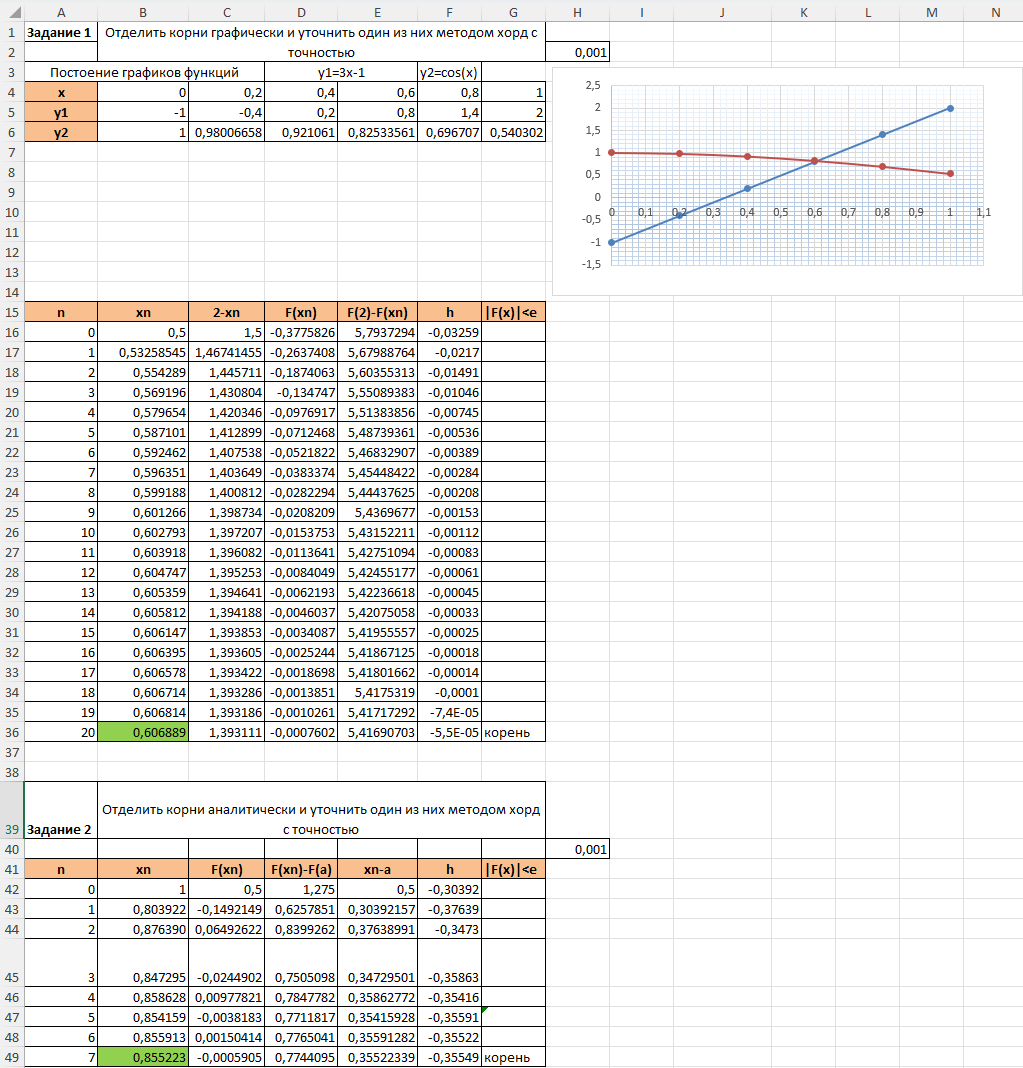


Рисунок 2- Решение работы 1 c формулами

## Работа 2

Рисунок 3 - Решение работы 2 без формул

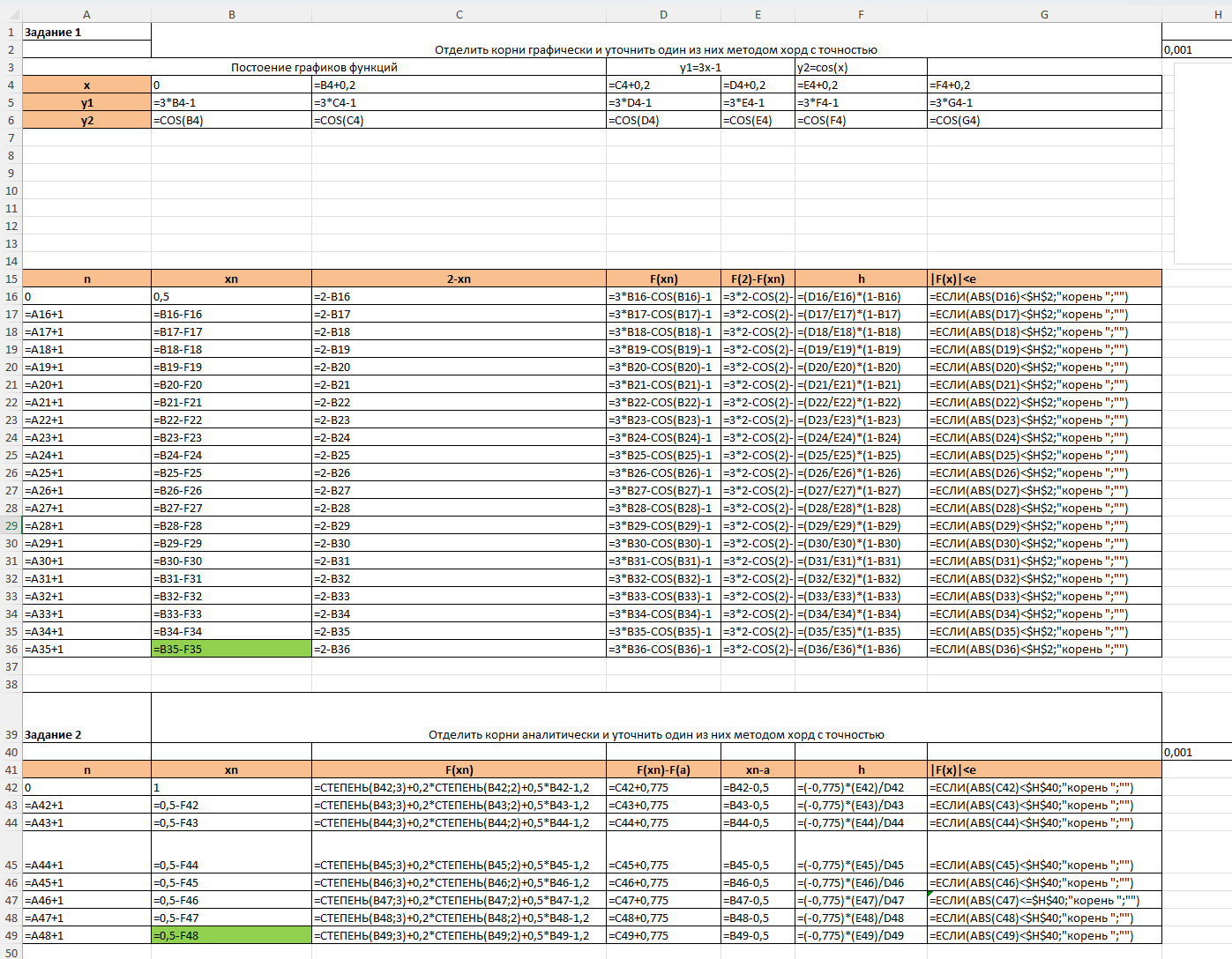


Рисунок 4- Решение работы 2 с формулами

## Работа 3

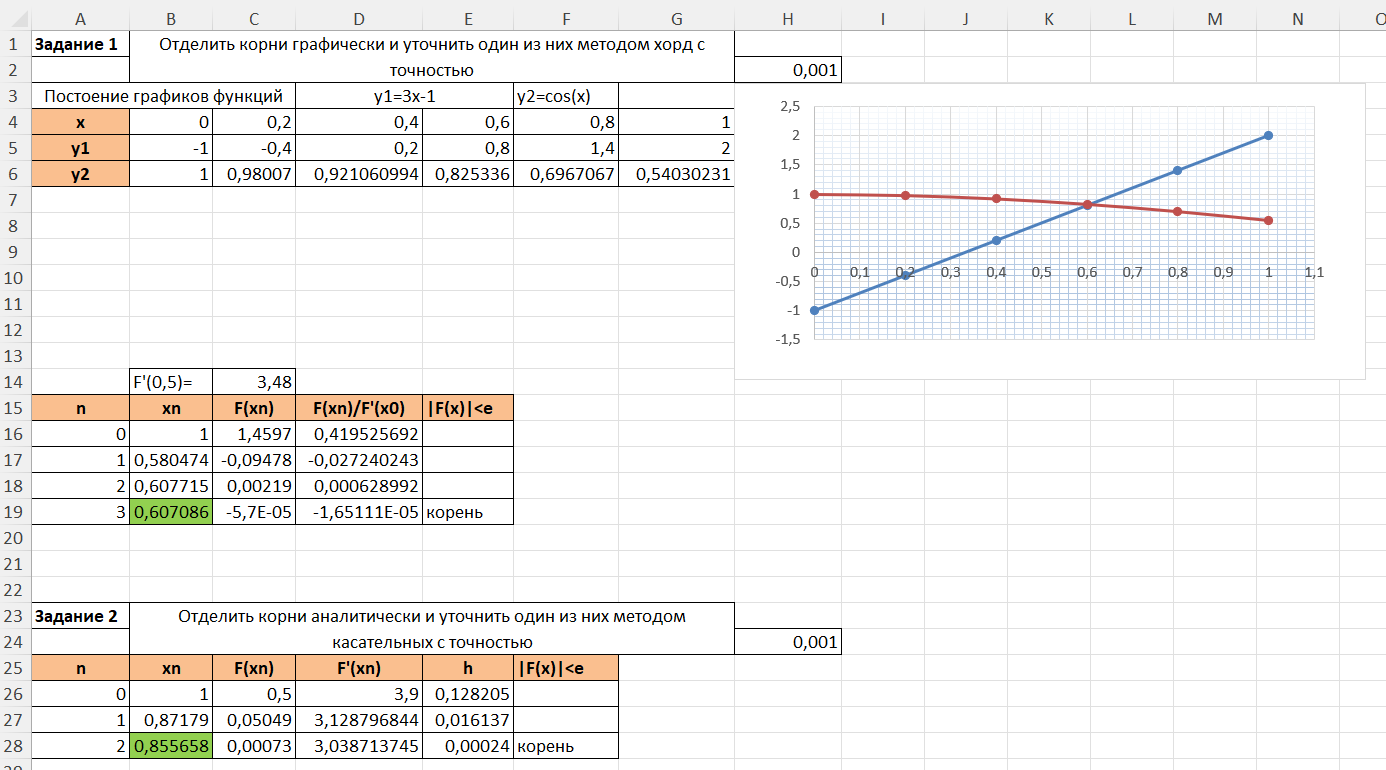


Рисунок 5- Решение работы 3 без формул

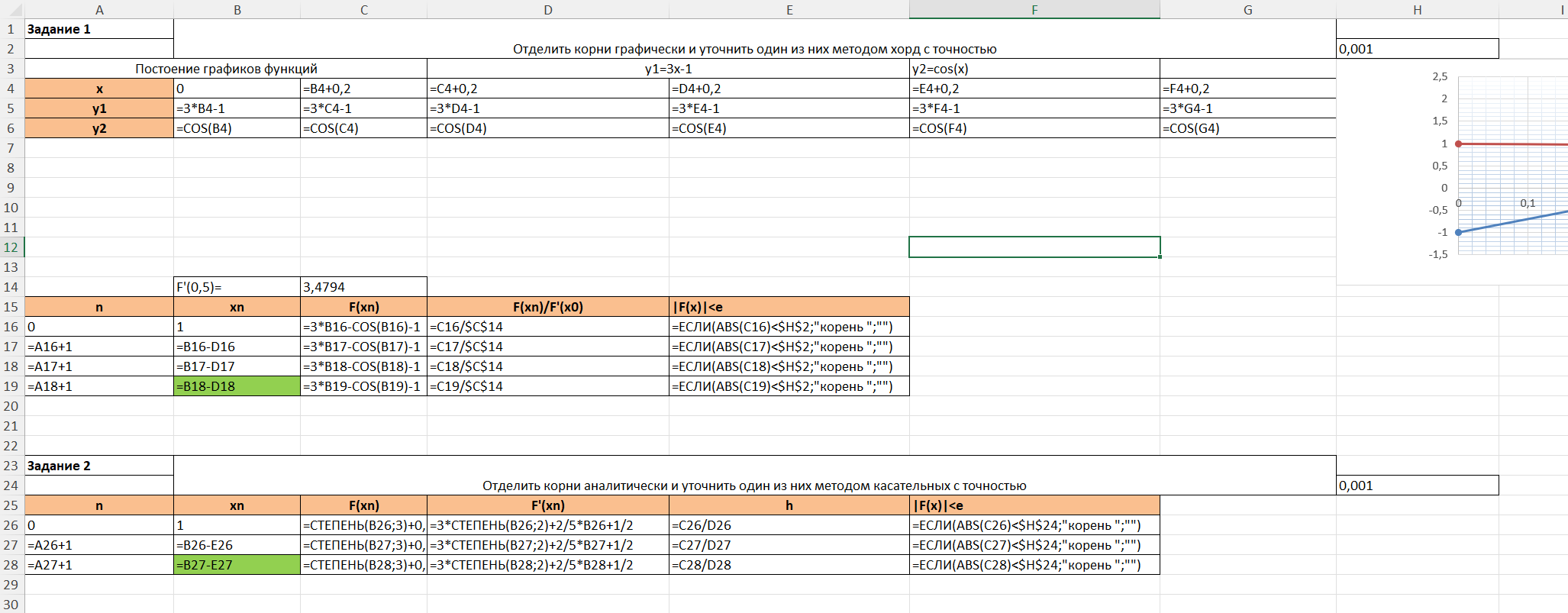


Рисунок 6 - Решение работы 3 с формулами

## Работа 4

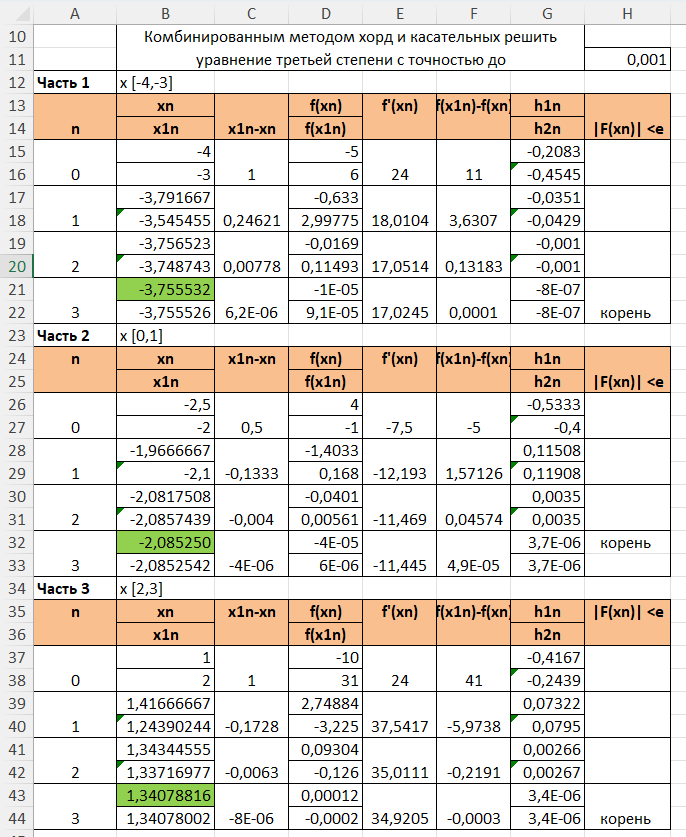


Рисунок 7-Решение работы 4 без формул

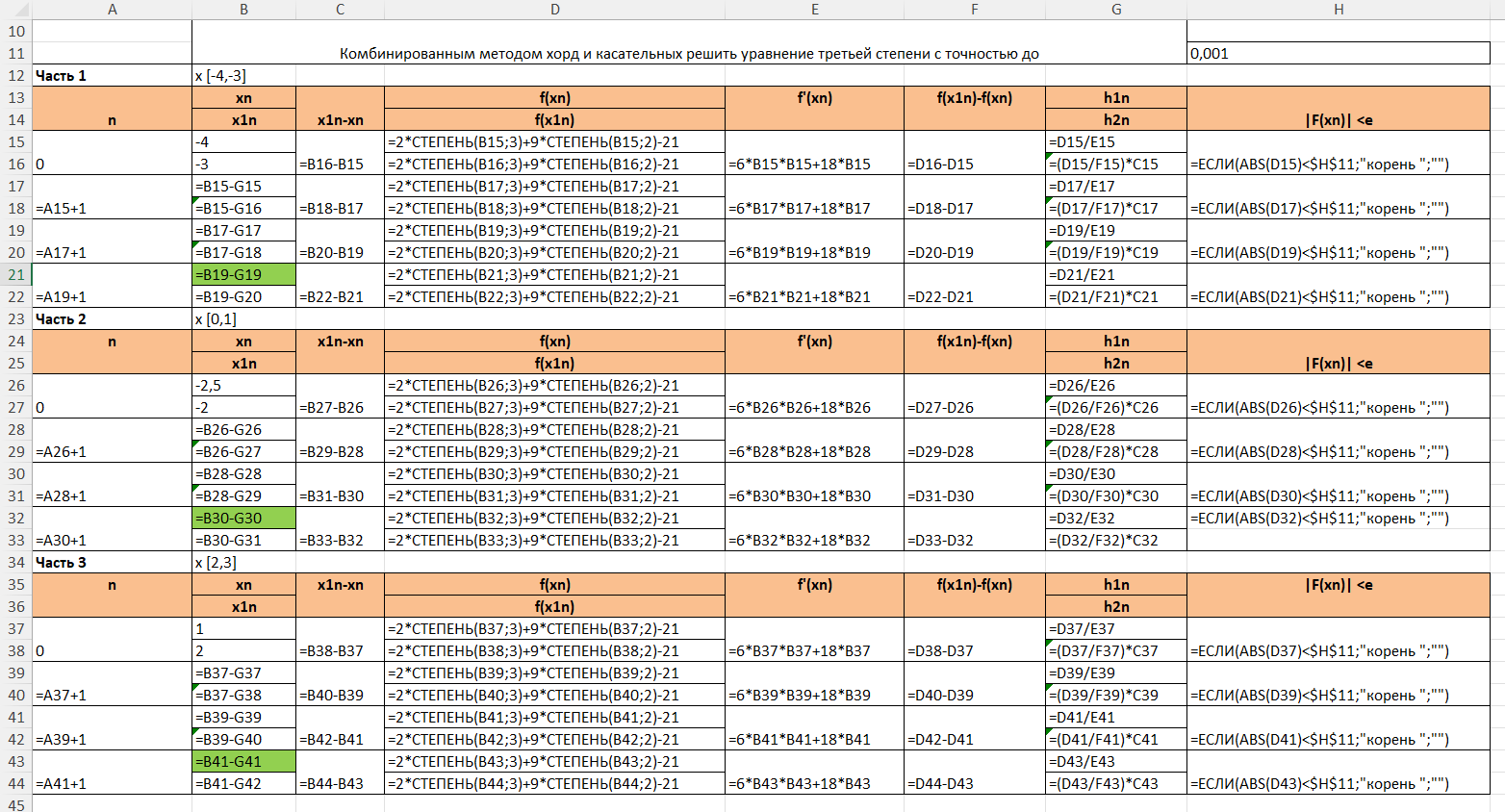
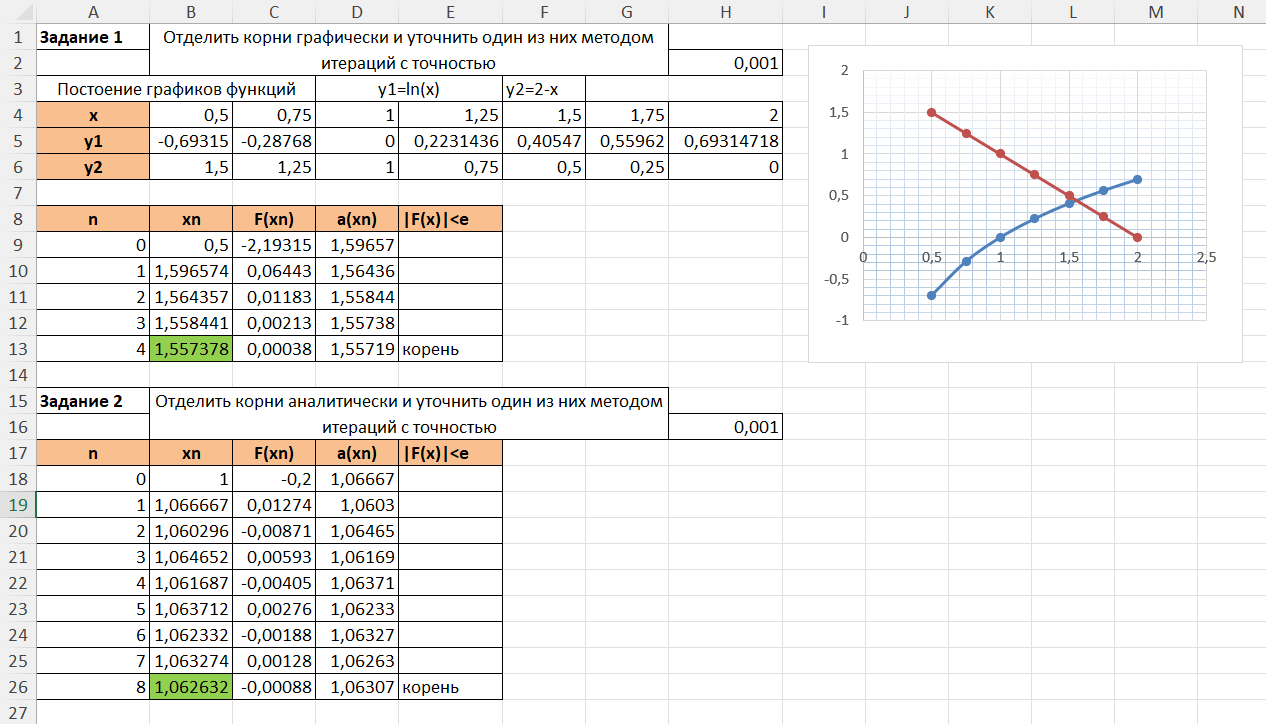


Рисунок 8 - Решение работы 4 с формулами

## Работа 5

  
Рисунок 9 - Решение работы 5 без формул

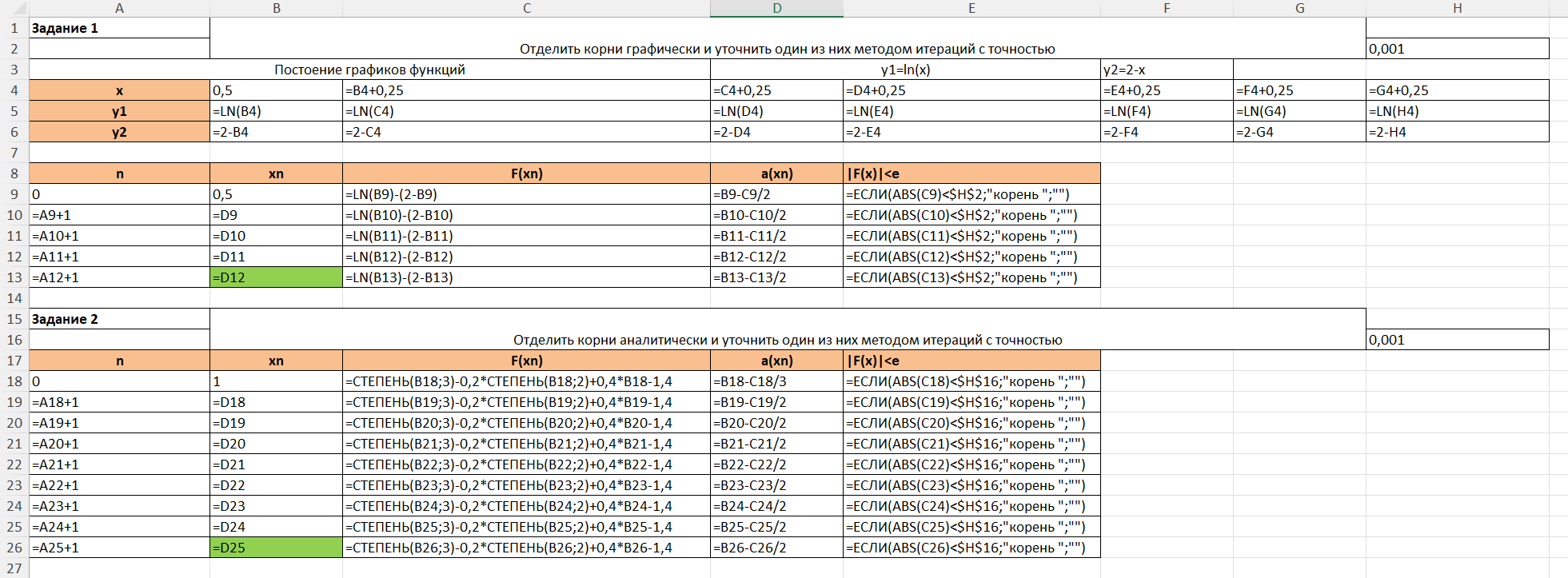
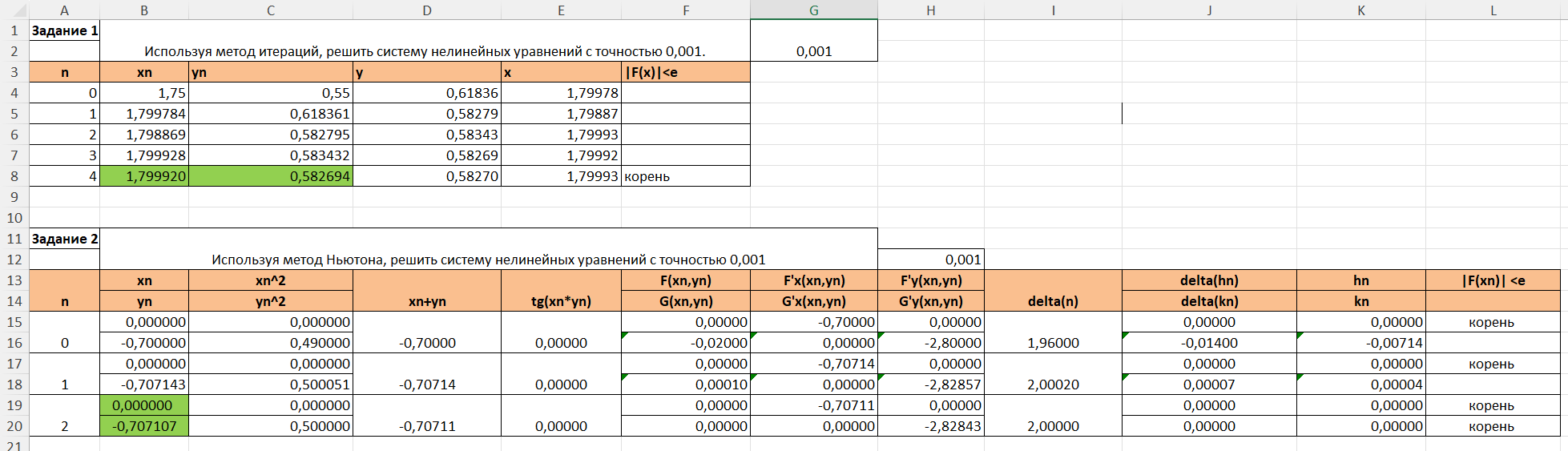


Рисунок 10 Решение работы 5 с формулами

## Работа 6

Рисунок 11- Решение работы 6 без формул

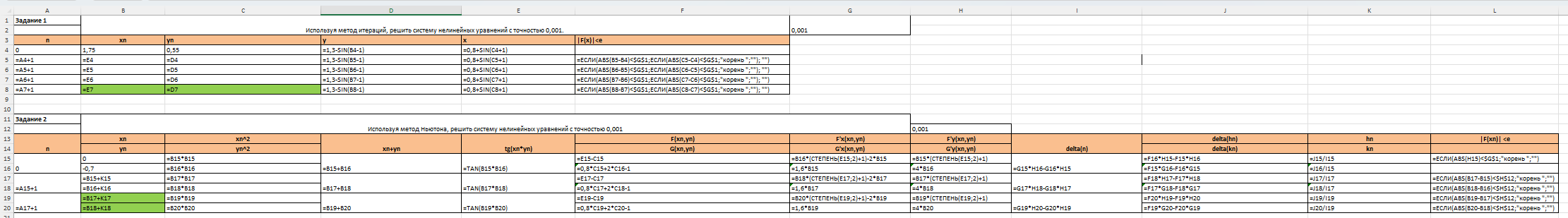


Рисунок 12 - Решение работы 6 с формулами

# Реализация задачи в Mathcad 15

### **Работа 1**

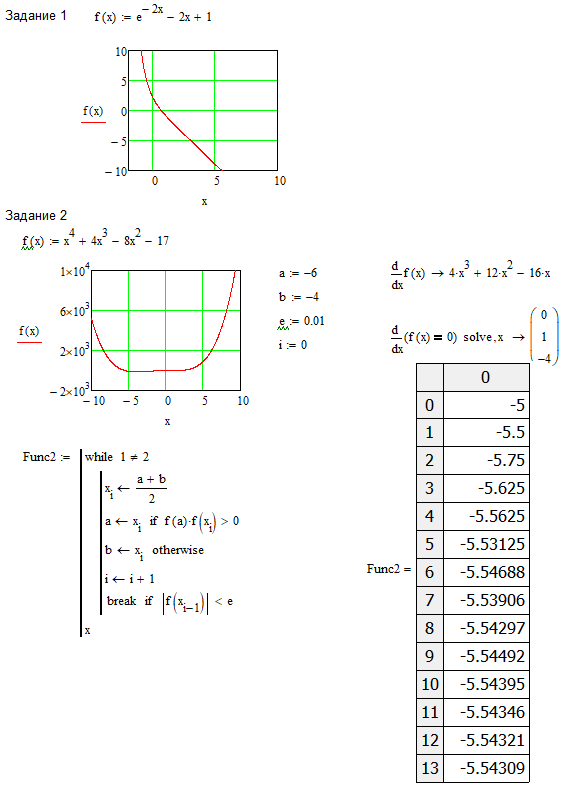


Рисунок 13 – работа 1 часть 1в Mathcad

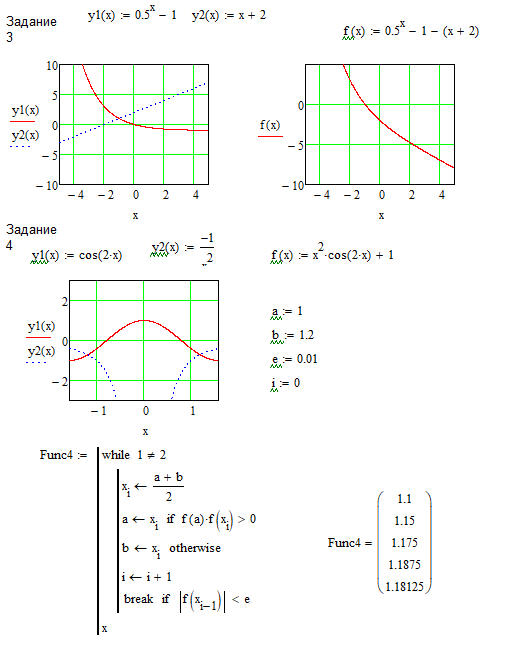


Рисунок 14 – работа 1 часть 2 в Mathcad

## Работа 2

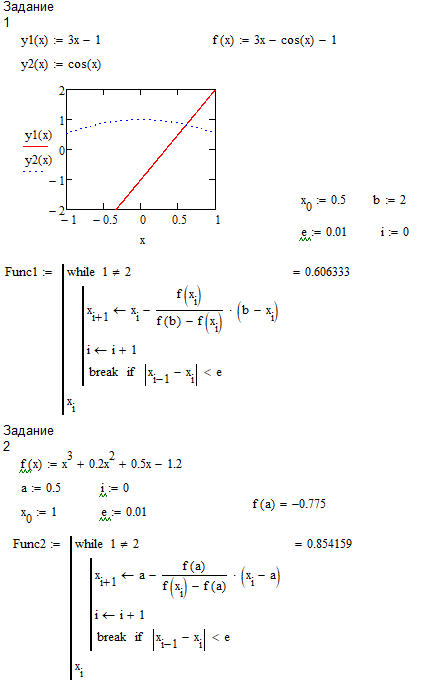


Рисунок 15 – работа 2 в Mathcad

## Работа 3

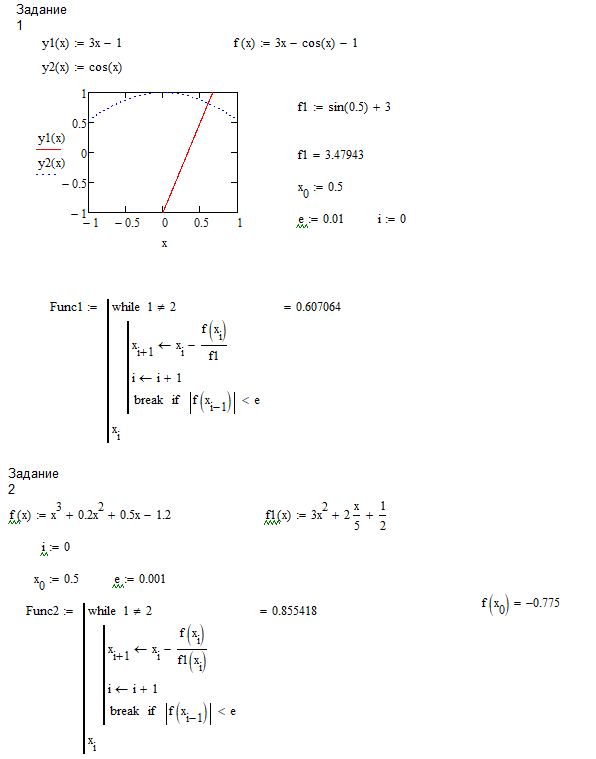


Рисунок 16 – работа 3 в Mathcad

## Работа 4

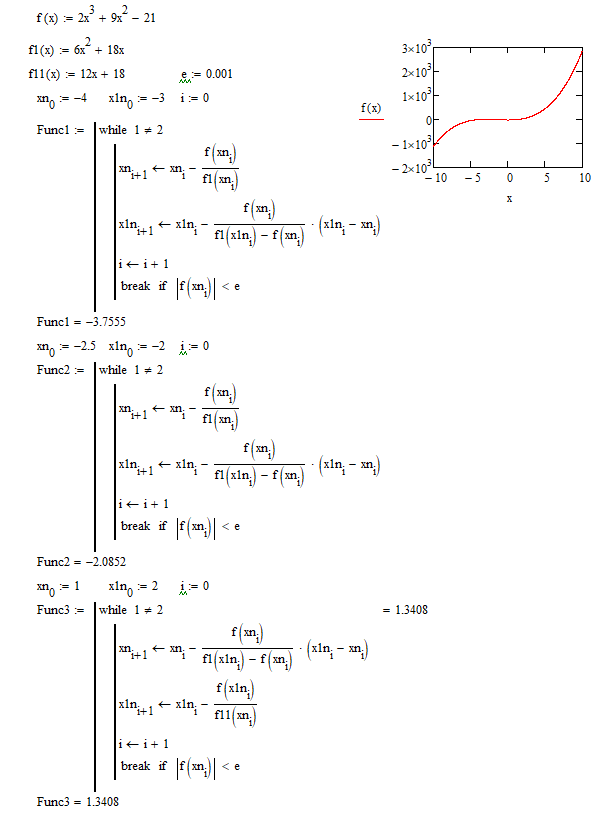


Рисунок 17 – работа 4 в Mathcad

## Работа 5

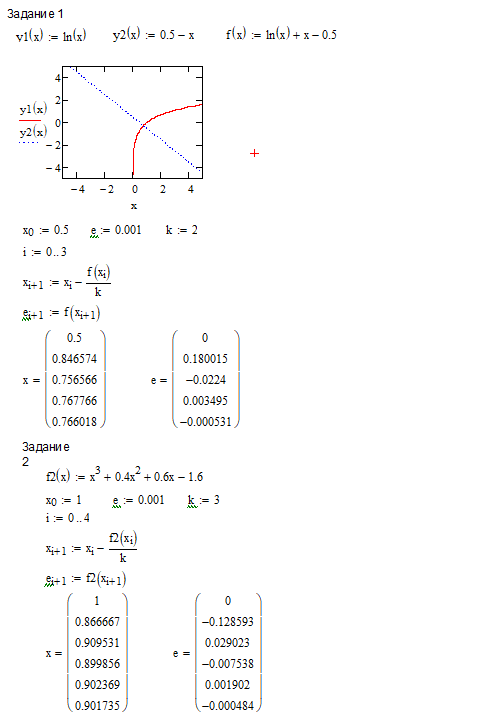


Рисунок 18 – работа 5 в Mathcad

## Работа 6

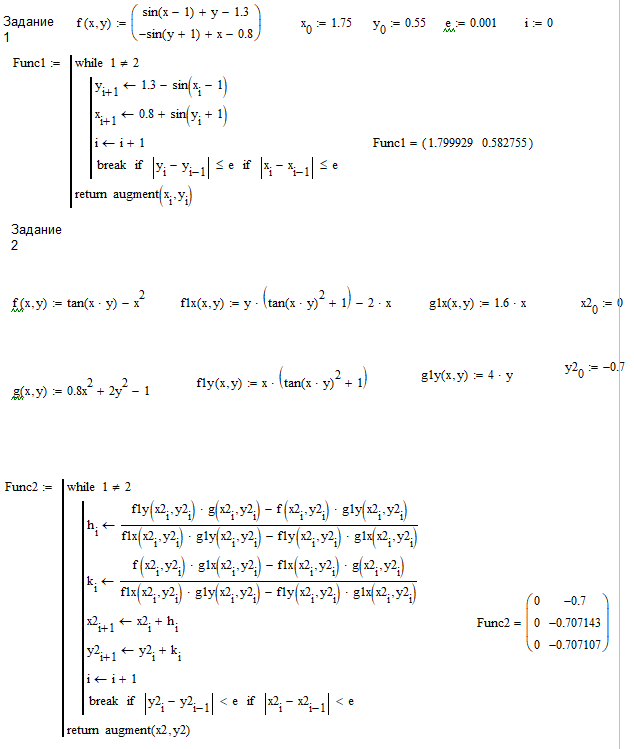


Рисунок 19 – работа 6 в Mathcad

# Реализация в С++

Среда разработки: Visual Studio 2019

#include <Windows.h>

#include <stdlib.h>

#include <math.h>

#include <cstdio>

#include <iostream>

#include <iomanip>

#include <cmath>

#include <limits>

using namespace std;

double fun11(double x) {

return exp(-2 \* x) - 2 \* x + 1;

}

double fun12(double x) {

return pow(x, 4) + 4 \* pow(x, 3) - 8 \* pow(x, 2) - 17;

}

double fun13(double x) {

return pow(0.5, x) - 1 - (x + 2);

}

double fun14(double x) {

return pow(x, 2) \* cos(2 \* x) + 1;

}

double method\_prob(double a, double b, double e, double f(double)) {

if (f(a) \* f(b) >= 0) {

cout << "В этом промежутке нет корня";

return 0;

}

double x = a;

do {

x = (a + b) / 2;

if (f(x) == 0.0)

break;

else if (f(x) \* f(a) < 0)

b = x;

else

a = x;

} while (fabs(f(x)) >= e);

return x;

}

double fun21(double x) {

return 3 \* x - cos(x) - 1;

}

double fun22(double x) {

return pow(x, 3) + 0.2 \* pow(x, 2) + 0.5 \* x - 1.2;

}

double fun31(double x) {

return 3 \* x - cos(x) - 1;

}

double dfun31(double x) {

return 3 + sin(x);

}

double fun32(double x) {

return pow(x, 3) + 0.2 \* pow(x, 2) + 0.5 \* x - 1.2;

}

double dfun32(double x) {

return 3 \* pow(x, 2) + 0.4 \* x + 0.5;

}

double method\_kasatelnih(double x0, double e, double f(double), double df(double)) {

double x = x0 - f(x0) / df(x0);

cout << x << endl;

do {

x0 = x;

x = x0 - f(x0) / df(x0);

cout << x << endl;

} while (fabs(f(x)) > e);

return x;

}

double fun41(double x) {

return 2 \* pow(x, 3) + 9 \* pow(x, 2) - 21;

}

double dfun41(double x) {

return 6 \* pow(x, 2) + 18 \* x;

}

double ddfun41(double x) {

return 12 \* x + 18;

}

double fun51(double x) {

return 2 - log(x);

}

double fun52(double x) {

return pow(x, 3) - 0.2 \* pow(x, 2) + 0.4 \* x - 1.4;

}

double method\_iteracii(double x, double e, double f(double), double k) {

do {

x = x - f(x) / k;

cout << x << endl;

} while (fabs(f(x)) > e);

return x;

}

// Функция f(x, y) = tg(xy) - x^2

double f62(double x, double y) {

return tan(x \* y) - pow(x, 2);

}

// Функция g(x, y) = 0.8x^2 + 2y^2 - 1

double g62(double x, double y) {

return 0.8 \* pow(x, 2) + 2 \* pow(y, 2) - 1;

}

// Производная f по x

double f62x(double x, double y) {

return y / (pow(cos(x \* y), 2)) - 2 \* x;

}

// Производная f по y

double f62y(double x, double y) {

return x / (pow(cos(x \* y), 2));

}

// Производная g по x

double g62x(double x, double y) {

return 1.6 \* x;

}

// Производная g по y

double g62y(double x, double y) {

return 4 \* y;

}

void method\_newton(double x0, double y0, double& x, double& y, double e) {

x = x0;

y = y0;

do {

x0 = x;

y0 = y;

double d = f62x(x0, y0) \* g62y(x0, y0) - f62y(x0, y0) \* g62x(x0, y0);

cout << "d " << d << endl;

double deltahn = f62y(x0, y0) \* g62(x0, y0) - f62(x0, y0) \* g62y(x0, y0);

cout << "deltahn " << deltahn << endl;

double deltakn = g62x(x0, y0) \* f62(x0, y0) - g62(x0, y0) \* f62x(x0, y0);

cout << "deltakn " << deltakn << endl;

x = x0 + deltahn / d;

y = y0 + deltakn / d;

cout << "x = " << x << "; y = " << y << " " << endl;

} while ((fabs(x - x0) > e) || (fabs(y - y0) > e));

}

double graph(double a, double b, double graphic(double)) {

float x;

// Получаем контекст устройства консоли

HDC hDC = GetDC(GetConsoleWindow());

// Создаем перо для рисования линий

HPEN Pen = CreatePen(PS\_SOLID, 2, RGB(255, 255, 255));

// Выбираем созданное перо

SelectObject(hDC, Pen);

// Рисуем оси координат

MoveToEx(hDC, 0, a, NULL);

LineTo(hDC, 200, a);

MoveToEx(hDC, 100, b, NULL);

LineTo(hDC, 100, b + 200);

for (x = -8.0f; x <= 8.0f; x += 0.01f) // O(100,85) - центр

{

// Рисуем график

MoveToEx(hDC, 10 \* x + 100, -10 \* graphic(x) + a, NULL); // 10 - масштаб

LineTo(hDC, 10 \* x + 100, -10 \* graphic(x) + a);

}

// Удаляем перо

DeleteObject(Pen);

ReleaseDC(GetConsoleWindow(), hDC);

getchar(); // Ожидание нажатия клавиши

return 0;

}

int main() {

setlocale(LC\_ALL, "RUS");

cout << fixed << setprecision(6);

int task;

cout << "Выберите номер работы от 1 до 6: ";

cin >> task;

if (task < 1 || task > 6) {

cout << "Такой работы нет. Попробуйте еще раз" << endl;

return 0;

}

int chase;

switch (task) {

case 1:

cout << "Работа 1 состоит из 4 заданий" << endl;

cout << "Выберите задание, которое нужно решить:" << endl;

cout << "1. Отделить корни уравнения графически и уточнить один из них методом хорд с точностью 0,001: e^(-2x) - 2x + 1 = 0" << endl;

cout << "2. Отделить корни уравнения аналитически и уточнить один из них методом хорд с точностью 0,001: x^4 + 4x^3 - 8x^2 - 17 = 0" << endl;

cout << "3. Отделить корни уравнения графически: 0.5^x - 1 = (x + 2)" << endl;

cout << "4. Отделить корни уравнения графически и уточнить один из них методом проб с точностью до 0,01: x^2 \* cos(2x) + 1 = 0" << endl;

cout << "Задание: ";

cin >> chase;

if (chase == 1 or chase == 3) {

switch (chase) {

case 1:

cout << "Построение графика исходной функции" << endl;

graph(400, 300, fun11);

getchar();

break;

case 3:

cout << "Построение графика исходной функции" << endl;

graph(400, 300, fun13);

getchar();

break;

}

}

else if (chase == 2 or chase == 4) {

double e, a, b, x1;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

cout << "Вы ввели точность: " << e << endl;

cout << "Введите левую границу интервала: ";

while (!(cin >> a)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Левая граница интервала равна: " << a << endl;

cout << "Введите правую границу интервала: ";

while (!(cin >> b)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Правая граница интервала равна: " << b << endl;

cout << "Уточняем корень методом проб: " << endl;

switch (chase) {

case 2:

x1 = method\_prob(a, b, e, fun12);

cout << "Ответ: x = " << x1 << endl;

cout << "Построение графика исходной функции" << endl;

graph(500, 400, fun12);

getchar();

break;

case 4:

x1 = method\_prob(a, b, e, fun14);

cout << "Ответ: x = " << x1 << endl;

cout << "Построение графика исходной функции" << endl;

graph(500, 400, fun14);

getchar();

break;

}

}

else {

cout << "Такого задания в этой работе нет, попробуйте еще раз." << endl;

}

return 0;

break;

case 2:

cout << "Работа 2 состоит из 2 заданий" << endl;

cout << "Выберите задание, которое нужно решить:" << endl;

cout << "1. Отделить корни уравнения графически и уточнить один из них методом хорд с точностью 0,001: 3x - cos(x) - 1 = 0" << endl;

cout << "2. Отделить корни уравнения аналитически и уточнить один из них методом хорд с точностью 0,001: x^3 + 0.2x^2 + 0.5x - 1.2 = 0" << endl;

cout << "Задание: ";

cin >> chase;

if (chase == 1 or chase == 2) {

double e, a, b, x;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

}

else {

cout << "Такого задания в этой работе нет, попробуйте еще раз." << endl;

}

return 0;

break;

case 3:

cout << "Работа 3 состоит из 2 заданий" << endl;

cout << "Выберите задание, которое нужно решить:" << endl;

cout << "1.Отделить корни уравнения графически и уточнить один из них методом касательных с точностью 0,001: 3x - cos(x) - 1 = 0 " << endl;

cout << "2.Отделить корни уравнения аналитически и уточнить один из них методом касательных с точностью 0,001: x^3 + 0.2x^2 + 0.5x - 1.2 = 0 " << endl;

cout << "Задание: ";

cin >> chase;

if (chase == 1 or chase == 2) {

double e, x0, x, dffix;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

cout << "Вы ввели точность: " << e << endl;

cout << "Введите начальное приближение: ";

while (!(cin >> x0)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Начальное приближение равно: " << x0 << endl;

cout << "Уточняем корень методом касательных: " << endl;

switch (chase) {

case 1:

dffix = x0;

x = x0 - fun31(x0) / dfun31(dffix);

cout << x << endl;

do {

x0 = x;

x = x0 - fun31(x0) / dfun31(dffix);

cout << x << endl;

} while (fabs(fun31(x)) > e);

cout << "Ответ: x = " << x << endl;

cout << "Построение графика исходной функции" << endl;

graph(500, 400, fun31);

getchar();

break;

case 2:

x = method\_kasatelnih(x0, e, fun32, dfun32);

cout << "Ответ: x = " << x << endl;

cout << "Построение графика исходной функции" << endl;

graph(450, 350, fun32);

getchar();

break;

}

}

else {

cout << "Такого задания в этой работе нет, попробуйте еще раз." << endl;

}

return 0;

break;

case 4:

cout << "Работа 4 состоит из 1 задания" << endl;

cout << "Задание: Комбинированным методом хорд и касательных решить уравнение третьей степени, вычислив корни с точностью до 0,001: 2x^3 + 9x^2 - 21 = 0 " << endl;

double e, xn, x1n;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

cout << "Вы ввели точность: " << e << endl;

for (int i = 1; i <= 3; i++)

{

cout << "Введите левую границу для интервала " << i << ": ";

while (!(cin >> xn)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Левая граница для интервала " << i << " равна " << xn << endl;

cout << "Введите правую границу для интервала " << i << ": ";

while (!(cin >> x1n)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Правая граница для интервала " << i << " равна " << x1n << endl;

cout << "Уточняем корень комбинированным методом хорд и касательных: " << endl;

do {

if (fun41(xn) \* ddfun41(xn) > 0) {

xn = xn - fun41(xn) / dfun41(xn);

x1n = x1n - fun41(xn) \* (x1n - xn) / (dfun41(x1n) - fun41(xn));

}

else {

xn = xn - fun41(xn) \* (x1n - xn) / (dfun41(x1n) - fun41(xn));

x1n = x1n - fun41(x1n) / dfun41(x1n);

}

} while (fabs(fun41(xn)) > e);

cout << "Ответ: x" << i << " = " << xn << endl;

}

cout << "Построение графика исходной функции" << endl;

graph(700, 600, fun41);

getchar();

return 0;

break;

case 5:

cout << "Работа 5 состоит из 2 заданий" << endl;

cout << "Выберите задание, которое нужно решить:" << endl;

cout << "1.Отделить корни уравнения графически и уточнить один из них методом итераций с точностью 0,001: 2 - ln(x) = x" << endl;

cout << "2.Отделить корни уравнения аналитически и уточнить один из них методом итераций с точностью 0,001: x^3 - 0.2\*x^2 + 0.4\*x - 1.4 = 0" << endl;

cout << "Задание: ";

cin >> chase;

if (chase == 1 or chase == 2) {

double e, x0, x, dffix;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

cout << "Вы ввели точность: " << e << endl;

cout << "Введите начальное приближение: ";

while (!(cin >> x0)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Начальное приближение равно: " << x0 << endl;

cout << "Уточняем корень методом итераций: " << endl;

switch (chase) {

case 1:

x = method\_iteracii(x0, e, fun51, 2);

cout << "Ответ: x = " << x << endl;

cout << "Построение графика исходной функции" << endl;

graph(500, 400, fun51);

getchar();

break;

case 2:

x = method\_iteracii(x0, e, fun52, 3);

cout << "Ответ: x = " << x << endl;

cout << "Построение графика исходной функции" << endl;

graph(500, 400, fun52);

getchar();

break;

}

}

else {

cout << "Такого задания в этой работе нет, попробуйте еще раз." << endl;

}

return 0;

break;

case 6:

cout << "Работа 6 состоит из 2 заданий" << endl;

cout << "Выберите задание, которое нужно решить:" << endl;

cout << "1. Используя метод итераций, решить систему нелинейных уравнений с точностью до 0,001: 1) sin(x-1) = 1.3 - y; 2) x - sin(y+1) = 0.8" << endl;

cout << "2. Используя метод Ньютона, решить систему нелинейных уравнений с точностью 0,001: 1) tg(xy) = x^2; 2) 0.8\*x^2 + 2\*y^2 = 1" << endl;

cout << "Задание: ";

cin >> chase;

if (chase == 1 or chase == 2) {

double e, x0, y0, x, y;

cout << "Введите нужную точность: ";

while (!(cin >> e) || e >= 1 || e <= 0) {

if (cin.fail()) { // проверяем на ошибку ввода

cin.clear();

}

cout << "Ошибка! Пожалуйста, введите положительное число, меньше 1: ";

}

cout << "Вы ввели точность: " << e << endl;

cout << "Введите начальное приближение для x: ";

while (!(cin >> x0)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Начальное приближение равно: " << x0 << endl;

cout << "Введите начальное приближение для y: ";

while (!(cin >> y0)) {

cout << "Ошибка! Пожалуйста, введите число: ";

cin.clear();

}

cout << "Начальное приближение равно: " << y0 << endl;

switch (chase) {

case 1:

cout << "Уточняем корни методом итераций: " << endl;

x = x0; y = y0;

do {

x0 = x;

y0 = y;

x = 0.8 + sin(y0 + 1);

y = 1.3 - sin(x0 - 1);

cout << "x = " << x << "; y = " << y << " " << endl;

} while (fabs(x - x0) > e || fabs(y - y0) > e);

cout << "Достигнута заданная точность" << endl;

cout << "Ответ: x = " << x << "; y = " << y << " " << endl;

break;

case 2:

cout << "Уточняем корни методом Ньютона: " << endl;

method\_newton(x0, y0, x, y, e);

cout << "Ответ: x = " << x << "; y = " << y << " " << endl;

break;

}

}

else {

cout << "Такого задания в этой работе нет, попробуйте еще раз." << endl;

}

return 0;

break;

default:

cout << "Что-то пошло не так. Попробуйте еще раз." << endl;

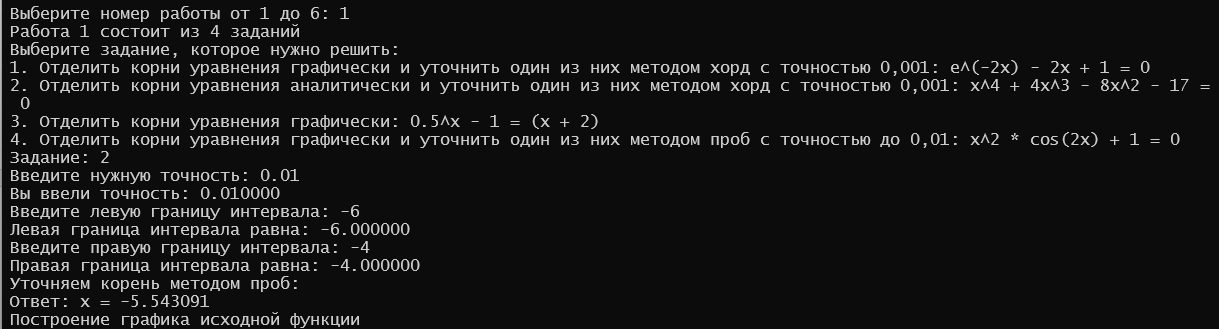
break;

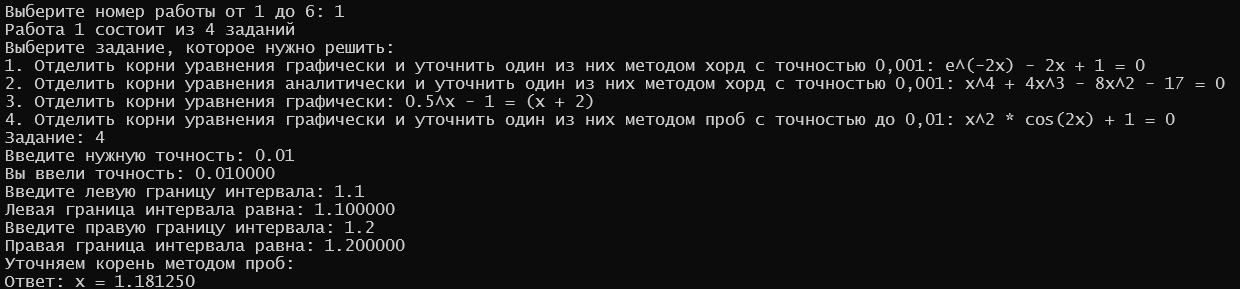
}

return 0;

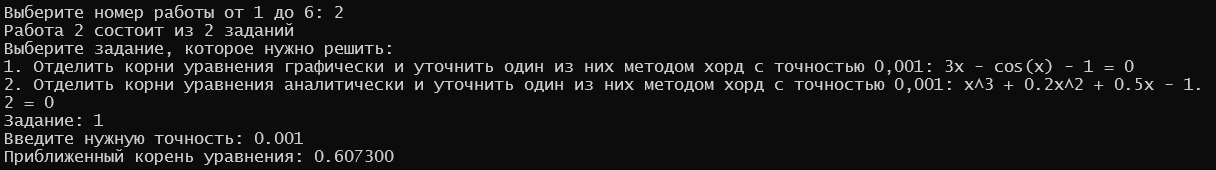
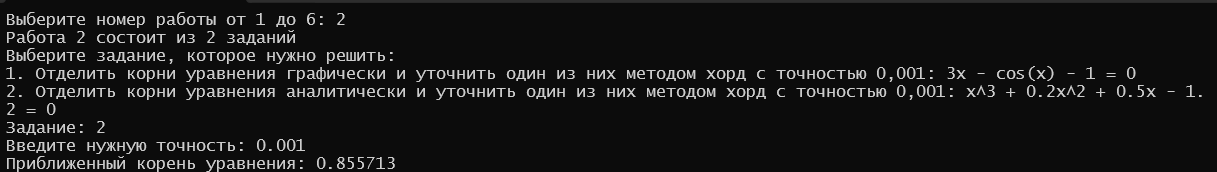
}

## Работа 1

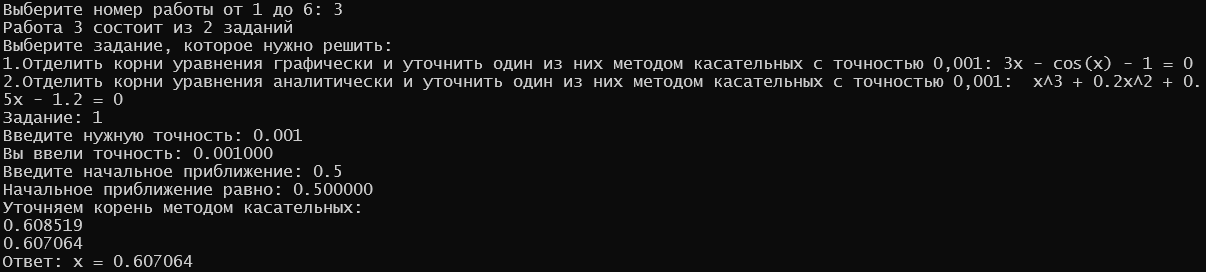
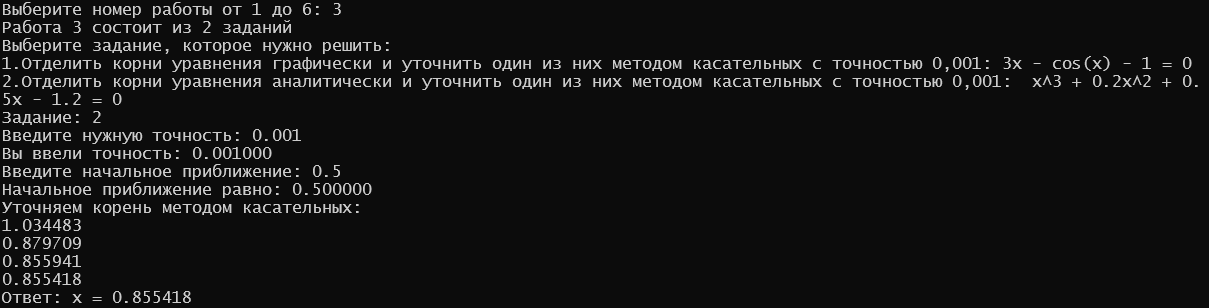
  
  
Рисунок 21 – работа 1 часть 1 на c++

  
Рисунок 20 – работа 1 часть 2 на c++

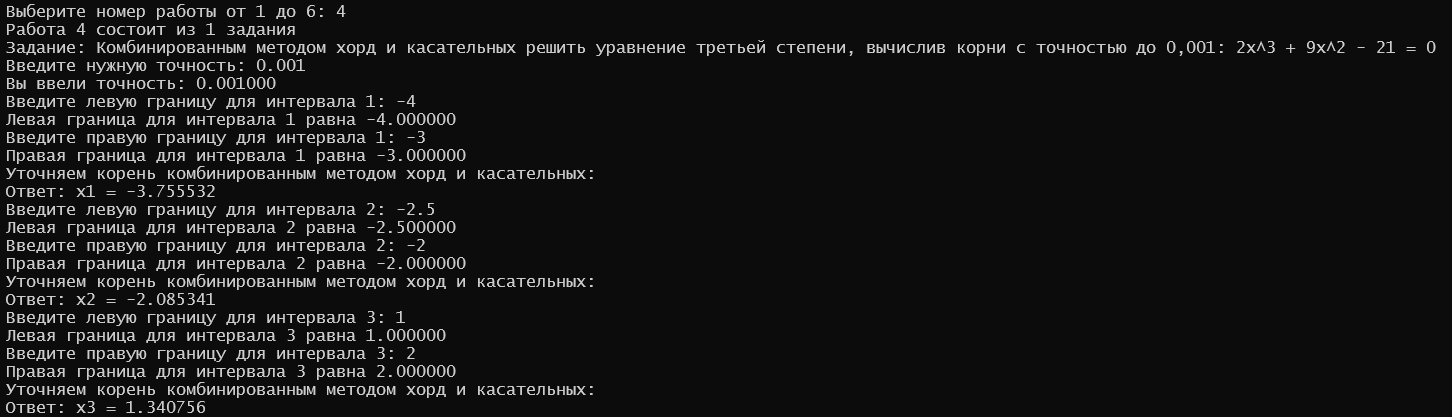
## Работа 2

  
Рисунок 22 – работа 2 часть 1 на c++  
  
  
Рисунок 23 – работа 2 часть 2 на c++

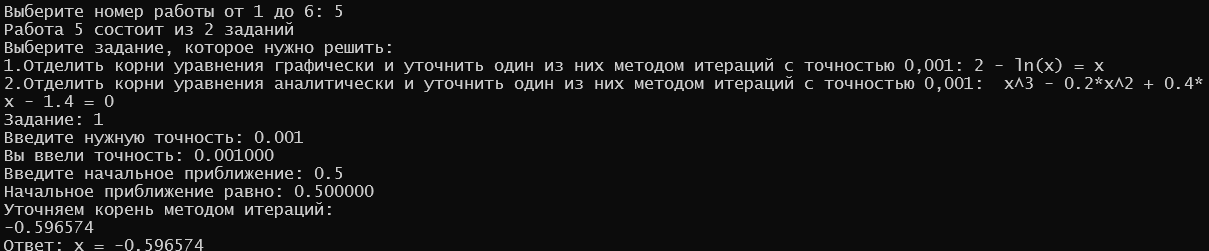
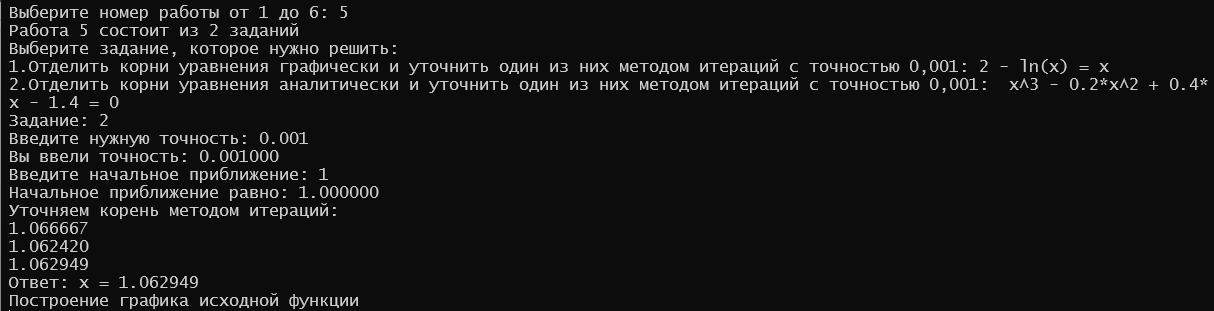
## Работа 3

  
Рисунок 22 – работа 3 часть 1 на c++  
  
  
Рисунок 23 – работа 3 часть 2 на c++

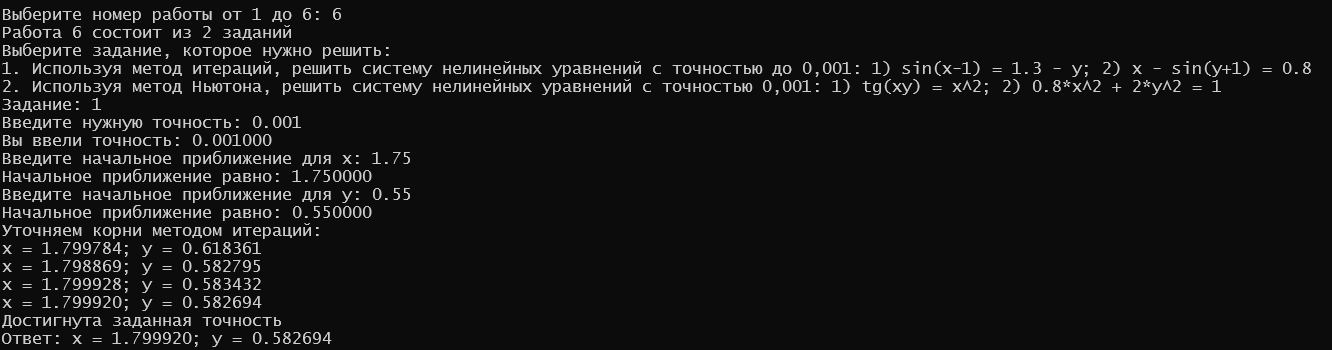
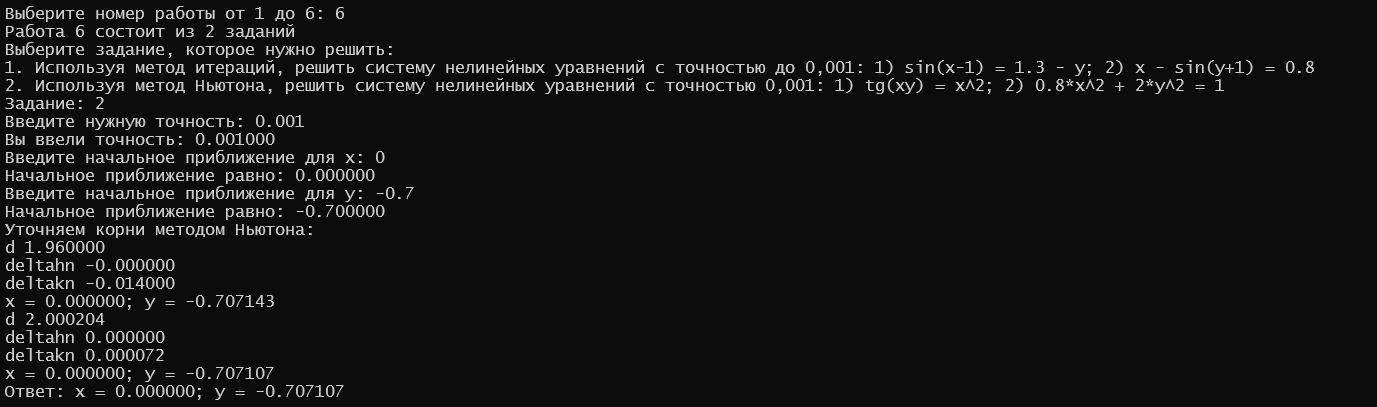
## Работа 4

  
Рисунок 24 – работа 4 на c++

## Работа 5

  
Рисунок 25 – работа 5 задание 1 на c++  
  
Рисунок 26 – работа 5 задание 2 на c++

## Работа 6

  
Рисунок 27 – работа 6 задание 1 на c++  
  
  
Рисунок 28 – работа 6 задание 2 на c++

# Реализация в Python 3.10

import math

# Заданные функции для работы 1

def fun11(x):

return pow(x, 4) + 4 \* pow(x, 3) - 8 \* pow(x, 2) - 17

def fun12(x):

return pow(x, 2) \* math.cos(2 \* x) + 1

# Заданные функции для работы 2

def fun21(x):

return 3 \* x - math.cos(x) - 1

def fun22(x):

return pow(x, 3) + 0.2 \* pow(x, 2) + 0.5 \* x - 1.2

# Заданные функции для работы 3

def fun31(x):

return 3 \* x - math.cos(x) - 1

def dfun31(x):

return 3 + math.sin(x)

def fun32(x):

return pow(x, 3) + 0.2 \* pow(x, 2) + 0.5 \* x - 1.2

def dfun32(x):

return 3 \* pow(x, 2) + 0.4 \* x + 0.5

# Заданные функции для работы 4

def fun41(x):

return 2 \* pow(x, 3) + 9 \* pow(x, 2) - 21

def dfun41(x):

return 6 \* pow(x, 2) + 18 \* x

def ddfun41(x):

return 12 \* x + 18

def f62(x, y):

return math.tan(x \* y) - x\*\*2

def g62(x, y):

return 0.8 \* x\*\*2 + 2 \* y\*\*2 - 1

def f62x(x, y):

return y / (math.cos(x \* y)\*\*2) - 2 \* x

def f62y(x, y):

return x / (math.cos(x \* y)\*\*2)

def g62x(x, y):

return 1.6 \* x

def g62y(x, y):

return 4 \* y

def method\_newton(x0, y0, e):

x = x0

y = y0

while True:

x0 = x

y0 = y

d = f62x(x0, y0) \* g62y(x0, y0) - f62y(x0, y0) \* g62x(x0, y0)

deltahn = f62y(x0, y0) \* g62(x0, y0) - f62(x0, y0) \* g62y(x0, y0)

deltakn = g62x(x0, y0) \* f62(x0, y0) - g62(x0, y0) \* f62x(x0, y0)

x = x0 + deltahn / d

y = y0 + deltakn / d

if abs(x - x0) <= e and abs(y - y0) <= e:

break

return x, y

# Метод половинного деления (метод проб)

def method\_prob(a, b, e, f):

if f(a) \* f(b) >= 0:

print("В этом промежутке нет корня")

return 0

x = a

while True:

x = (a + b) / 2

if f(x) == 0.0:

break

elif f(x) \* f(a) < 0:

b = x

else:

a = x

if abs(f(x)) < e:

break

return x

# Метод хорд

def method\_chords(a, b, e, f):

while abs(b - a) > e:

a = b - (b - a) \* f(b) / (f(b) - f(a))

b = a + (a - b) \* f(a) / (f(a) - f(b))

return b

# Метод касательных (метод Ньютона)

def method\_tangents(x0, e, f, df):

x = x0 - f(x0) / df(x0)

while abs(f(x)) > e:

x0 = x

x = x0 - f(x0) / df(x0)

return x

# Метод итераций

def method\_iteracii(x, e, phi, max\_iter=1000):

iter\_count = 0

while iter\_count < max\_iter:

next\_x = phi(x)

if abs(next\_x - x) < e:

return next\_x

x = next\_x

iter\_count += 1

raise ValueError("Метод итераций не сошелся после {} итераций".format(max\_iter))

# Подходящие функции phi(x) для каждой из функций f(x)

def phi51(x):

return math.log(x) + 2

def phi52(x):

return pow((0.2 \* pow(x, 2) - 0.4 \* x + 1.4), 1/3)

def main():

while True:

print("Выберите номер работы от 1 до 6 (для выхода введите любое другое число): ")

task = int(input())

if task < 1 or task > 6:

print("Выход из программы.")

break

if task == 1:

print("Работа 1 состоит из 2 заданий")

print("Выберите задание, которое нужно решить:")

print("1.Отделить корни уравнения аналитически и уточнить один из них методом проб с точностью до 0,01: x^4 + 4x^3 - 8x^2 - 17 = 0")

print("2.Отделить корни уравнения аналитически и уточнить один из них методом проб с точностью до 0,01: x^2 \* cos(2x) = -1")

chase = int(input("Задание: "))

if chase == 1:

e = float(input("Введите нужную точность: "))

a = float(input("Введите левую границу интервала: "))

b = float(input("Введите правую границу интервала: "))

print("Уточняем корень методом проб:")

x1 = method\_prob(a, b, e, fun11)

print("Ответ: x =", x1)

elif chase == 2:

e = float(input("Введите нужную точность: "))

a = float(input("Введите левую границу интервала: "))

b = float(input("Введите правую границу интервала: "))

print("Уточняем корень методом проб:")

x1 = method\_prob(a, b, e, fun12)

print("Ответ: x =", x1)

else:

print("Такого задания в этой работе нет, попробуйте еще раз.")

elif task == 2:

print("Работа 2 состоит из 2 заданий")

print("Выберите задание, которое нужно решить:")

print("1. Отделить корни уравнения аналитически и уточнить один из них методом хорд с точностью 0,001: 3x - cos(x) - 1 = 0")

print("2. Отделить корни уравнения аналитически и уточнить один из них методом хорд с точностью 0,001: x^3 + 0.2x^2 + 0.5x - 1.2 = 0")

chase = int(input("Задание: "))

if chase == 1:

e = 0.001

a = -10.0

b = 10.0

print("Уточняем корень методом хорд:")

x1 = method\_chords(a, b, e, fun21)

print("Ответ: x =", x1)

elif chase == 2:

e = 0.001

a = -10.0

b = 10.0

print("Уточняем корень методом хорд:")

x1 = method\_chords(a, b, e, fun22)

print("Ответ: x =", x1)

else:

print("Такого задания в этой работе нет, попробуйте еще раз.")

elif task == 3:

print("Работа 3 состоит из 2 заданий")

print("Выберите задание, которое нужно решить:")

print("1.Отделить корни уравнения графически и уточнить один из них методом касательных с точностью 0,001: 3x - cos(x) - 1 = 0 ")

print("2.Отделить корни уравнения аналитически и уточнить один из них методом касательных с точностью 0,001: x^3 + 0.2x^2 + 0.5x - 1.2 = 0 ")

chase = int(input("Задание: "))

if chase == 1:

e = 0.001

x0 = float(input("Введите начальное приближение: "))

print("Уточняем корень методом касательных: ")

x1 = method\_tangents(x0, e, fun31, dfun31)

print("Ответ: x =", x1)

elif chase == 2:

e = 0.001

x0 = float(input("Введите начальное приближение: "))

print("Уточняем корень методом касательных: ")

x1 = method\_tangents(x0, e, fun32, dfun32)

print("Ответ: x =", x1)

else:

print("Такого задания в этой работе нет, попробуйте еще раз.")

elif task == 4:

print("Работа 4 состоит из 1 задания")

print("Задание: Комбинированным методом хорд и касательных решить уравнение третьей степени, вычислив корни с точностью до 0,001: 2x^3 + 9x^2 - 21 = 0 ")

e = float(input("Введите нужную точность: "))

for i in range(1, 4):

a = float(input(f"Введите левую границу для интервала {i}: "))

b = float(input(f"Введите правую границу для интервала {i}: "))

print(f"Уточняем корень для интервала {i} комбинированным методом хорд и касательных:")

xn = a

x1n = b

while abs(fun41(xn)) > e:

if fun41(xn) \* ddfun41(xn) > 0:

xn = xn - fun41(xn) / dfun41(xn)

x1n = x1n - fun41(xn) \* (x1n - xn) / (dfun41(x1n) - fun41(xn))

else:

xn = xn - fun41(xn) \* (x1n - xn) / (dfun41(x1n) - fun41(xn))

x1n = x1n - fun41(x1n) / dfun41(x1n)

print(f"Ответ: x{i} = {xn}")

break

elif task == 5:

print("Работа 5 состоит из 2 заданий")

print("Выберите задание, которое нужно решить:")

print("1. Отделить корни уравнения аналитически и уточнить один из них методом итераций с точностью 0,001: log(x) + 2 = 0")

print("2. Отделить корни уравнения аналитически и уточнить один из них методом итераций с точностью 0,001: (0.2 \* x^2 - 0.4 \* x + 1.4)^(1/3) = 0")

chase = int(input("Задание: "))

if chase == 1 or chase == 2:

e = 0.001

x0 = float(input("Введите начальное приближение: "))

k = float(input("Введите значение k: ")) # Добавлен ввод значения k

if chase == 1:

print("Уточняем корень методом итераций (x = log(x) + 2):")

result = method\_iteracii(x0, e, phi51)

print("Ответ: x =", result)

else:

print("Уточняем корень методом итераций (x = (0.2 \* x^2 - 0.4 \* x + 1.4)^(1/3)):")

result = method\_iteracii(x0, e, phi52)

print("Ответ: x =", result)

else:

print("Такого задания в этой работе нет, попробуйте еще раз.")

elif task == 6:

print("Работа 6 состоит из 2 заданий")

print("Выберите задание (1 или 2): ")

chase = int(input())

if chase == 1 or chase == 2:

e = float(input("Введите точность: "))

x0 = float(input("Введите начальное приближение для x: "))

y0 = float(input("Введите начальное приближение для y: "))

if chase == 1:

print("Уточняем корни методом итераций:")

x = x0

y = y0

while True:

x0 = x

y0 = y

x = 0.8 + math.sin(y0 + 1)

y = 1.3 - math.sin(x0 - 1)

print(f"x = {x}; y = {y}")

if abs(x - x0) <= e and abs(y - y0) <= e:

break

print("Достигнута заданная точность")

print(f"Ответ: x = {x}; y = {y}")

elif chase == 2:

print("Уточняем корни методом Ньютона:")

x, y = method\_newton(x0, y0, e)

print(f"Ответ: x = {x}; y = {y}")

else:

print("Такого задания в этой работе нет, попробуйте еще раз.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

## Работа 1

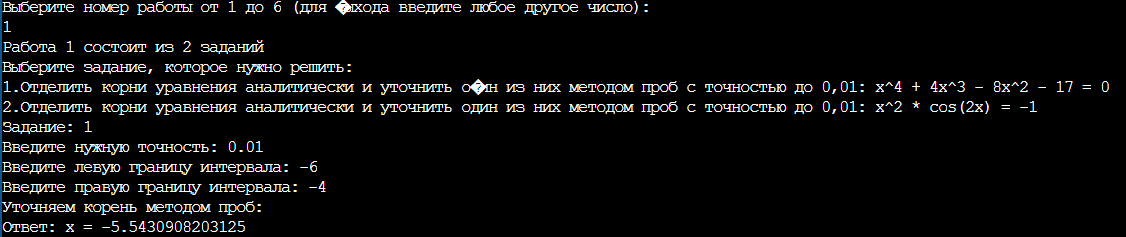
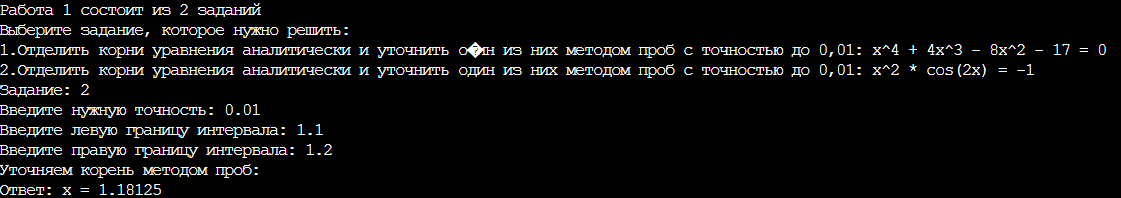
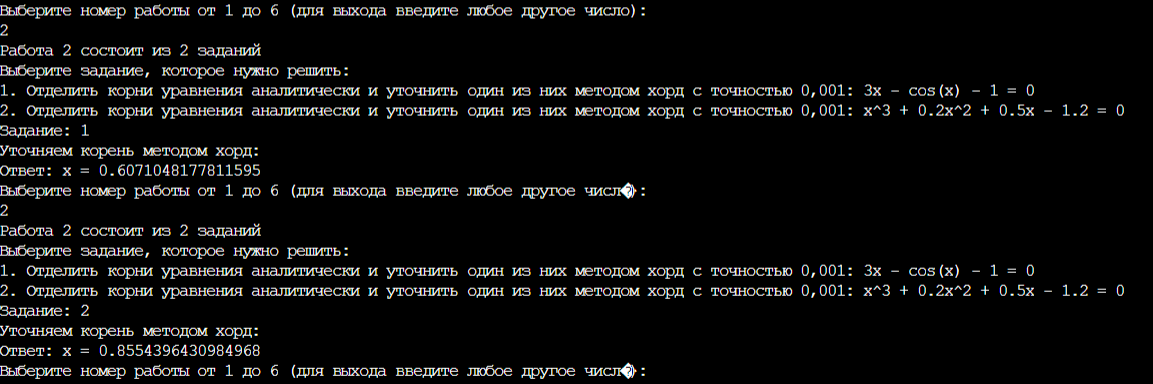
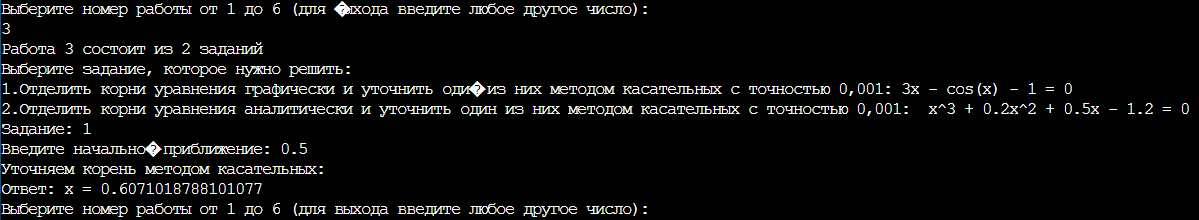
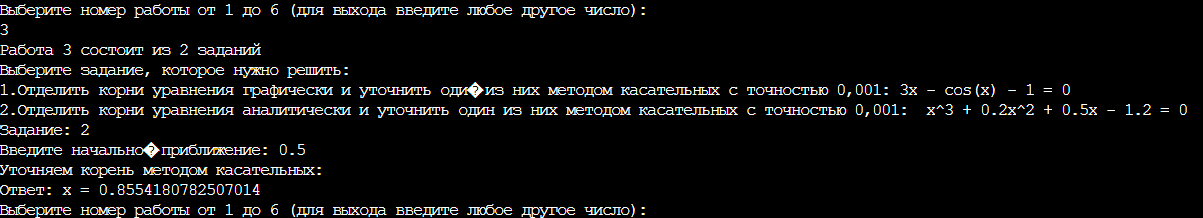


Рисунок 29 – работа 1 часть 1 на Python  
  
  
Рисунок 30 – работа 1 часть 2 на Python

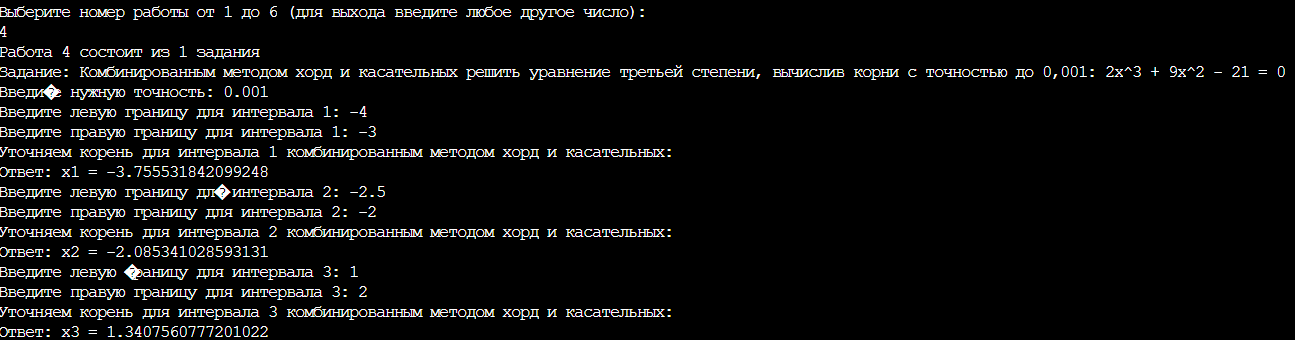
## Работа 2

  
Рисунок 31 – работа 2 на Python

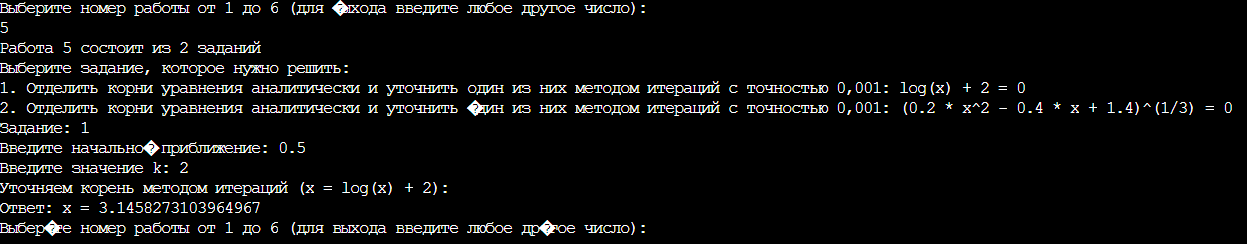
## Работа 3

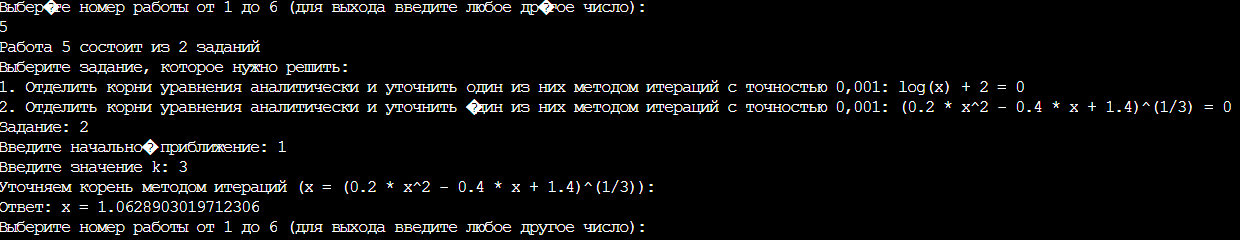
  
Рисунок 32 – работа 3 на часть 1 Python  
  
  
Рисунок 33 – работа 3 на часть2 Python

## Работа 4

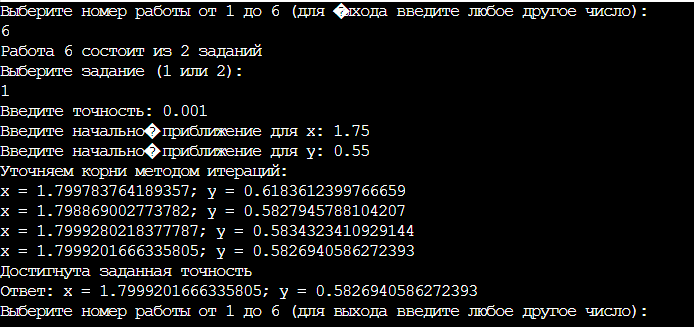
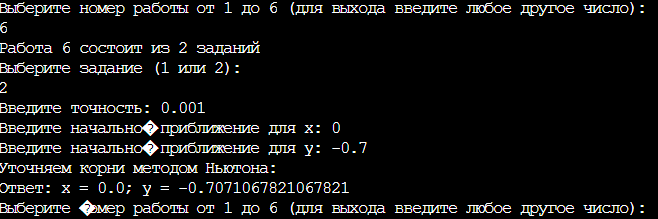
  
Рисунок 34 – работа 4 на Python

## Работа 5

  
Рисунок 35 – работа 5 на часть 1 Python

 Рисунок 36 – работа 5 на часть 2 Python

## Работа 6

  
Рисунок 37 – работа 6 на часть 2 Python  
  
Рисунок 38 – работа 6 на часть 2 Python

# Результаты и их анализ

## Работа 1

Таблица 8

Сводная таблица результатов работы 1

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходное уравнение | Полученные корни |
| Ручной счёт | Задание 2: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |
| Ручной счёт | Задание 4: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения нелинейных уравнений методом проб было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 8.

## Работа 2

Таблица 9

Сводная таблица результатов работы 2

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходное уравнение | Полученные корни |
| Ручной счёт | Задание 1: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |
| Ручной счёт | Задание 2: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения нелинейных уравнений методом хорд было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 9.

## Работа 3

Таблица 10

Сводная таблица результатов работы 3

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходное уравнение | Полученные корни |
| Ручной счёт | Задание 1: | 5 |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |
| Ручной счёт | Задание 2: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения нелинейных уравнений методом касательных было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 10.

## Работа 4

Таблица 11

Сводная таблица результатов работы 4

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходное уравнение | Полученные корни |
| Ручной счёт |  |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения нелинейных уравнений комбинированным методом хорд и касательных было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 11.

## Работа 5

Таблица 12

Сводная таблица результатов работы 5

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходное уравнение | Полученные корни |
| Ручной счёт | Задание 1: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |
| Ручной счёт | Задание 2: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения нелинейных уравнений методом итераций было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 12.

## Работа 6

Таблица 13

Сводная таблица результатов работы 6

|  |  |  |
| --- | --- | --- |
| Способ реализации | Исходная система: | Полученные корни |
| Ручной счёт | Задание 1: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |
| Ручной счёт | Задание 2: |  |
| Excel |
| Mathcad (ручной счет) |
| Mathcad (программирование) |
| С++ |
| Python |

Для решения систем нелинейных уравнений методом итераций и методом Ньютона было использовано 6 способов, конечные и промежуточные результаты вычислений совпали во всех способах. Результаты для каждого способа представлены в таблице 13.

# Вывод

В ходе лабораторной работы были решены нелинейные уравнения с помощью различных методов (Метод проб, метод хорд, метод касательных, метод итераций, метод Ньютона, комбинированный метод хорд и касательных). Каждый метод реализован 6 способами. (посчитан ручной счет, его контроль произведен в Excel и Mathcad, написаны программы на языках программирования C++ и Python, а также на прикладной программе Mathcad.) Все полученные ответы занесены в таблицу результатов и проанализированы.

Каждый из исследуемых методов имеет свои преимущества и недостатки. Для сравнения методов по времени и сложности выполнения, нужно учитывать не только их теоретические характеристики, но и конкретные условия задачи и особенности функции, решаемой методом. В зависимости от конкретной задачи, метод проб считается самым простым, хотя он может потребовать большего числа итераций. Он прост в реализации и не требует вычисления производных. Метод Ньютона считается самым быстрым, но он может потребовать больше вычислительных ресурсов из-за необходимости вычисления производных и выполнения сложных операций.

Версии использованного ПО:

* Microsoft Office Excel 2013
* Mathcad 15
* Python 3.10
* Visual Studio 2019

# Список литературы

1. Воробьева Г.Н., Данилова А.Н. - Практикум по вычислительной математике (1990г)
2. Основы вычислительной математики Демидович Б.П. Марон И.А. 1966 год