ОТЧЁТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №4
«Численное решение нелинейных уравнений»

Подготовил Сечейко Н.В., ст. гр. 421702

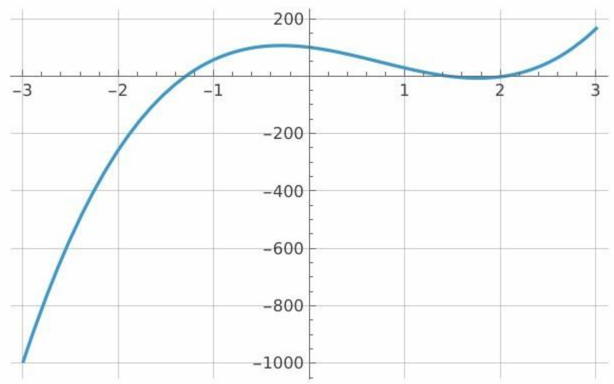
Задание 1. Отделить графически корни алгебраического уравнения $f(x)=0$. Найти один из них (нецелый) с точностью $eps=10^{-3}$ методом хорд. Указать потребовавшееся число итераций. Проиллюстрировать графически нахождение первых двух приближений.

Функция:
$f(x)=26x^3-57x^2-41x+102$

```
f[x_] := 26 x^3 - 57 x^2 - 41 x + 102
```

График (со всеми нулями):

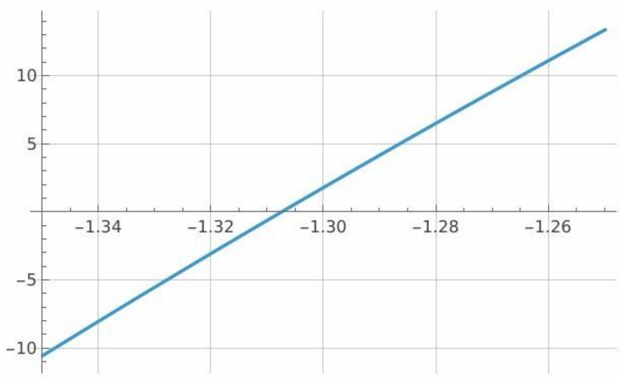
```
Plot[f[x], {x, -3, 3}, AxesOrigin → {0, 0}, PlotRange → All, GridLines → Automatic]
```
график функции   точка пересечения о···   отобража···   всё   линии коо···   автоматичес



а) Пересечения с OX вблизи (нули функции):

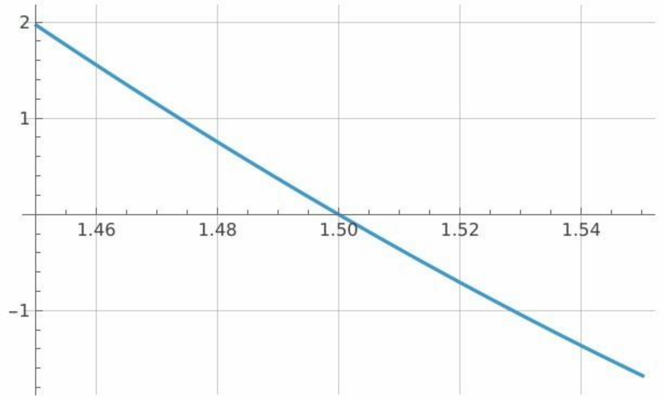
$x≈-1.31$:

```
Plot[f[x], {x, -1.35, -1.25}, AxesOrigin → {-1.35, 0}, PlotRange → All, GridLines → Automatic]
```
график функции   точка пересечения осей   отобража···   всё   линии коо···   автоматичес



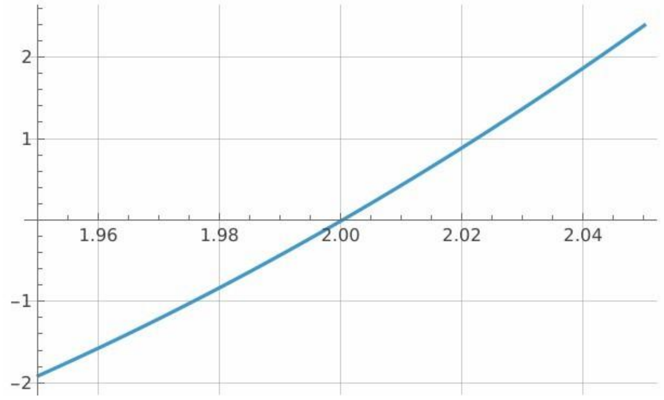
x=1.5:

```
Plot[f[x], {x, 1.45, 1.55}, AxesOrigin → {1.45, 0}, PlotRange → All, GridLines → Automatic]
график функции            точка пересечения осей    отобража⋯  всё  линии коо⋯  автоматичес
```



x=2:

```
Plot[f[x], {x, 1.95, 2.05}, AxesOrigin → {1.95, 0}, PlotRange → All, GridLines → Automatic]
график функции            точка пересечения осей    отобража⋯  всё  линии коо⋯  автоматичес
```



б) Нахождение нецелого корня методом хорд.

Формула метода хорд:

$$x_{next} = x_k - f(x_k)\,\frac{x_k - x_{prev}}{f(x_k) - f(x_{prev})}$$

Функция для метода хорд (возвращает найденный корень и количество итераций, которые потребовалось выполнить для получения корня с нужной точностью):

```
chord[f_, x0_, x1_, eps_] := Module[
                             программный модуль
  {xPrev = x0, xCurr = x1, xNext, iter = 0},
    While[True,
    цик⋯ истина
      iter++;
      xNext = xCurr - f[xCurr] (xCurr - xPrev) / (f[xCurr] - f[xPrev]);
      If[Abs[xNext - xCurr] < eps, Return[{xNext, iter}]];
      ⋯ абсолютное значение      вернуть управление
      xPrev = xCurr;
      xCurr = xNext;
    ]
]
```

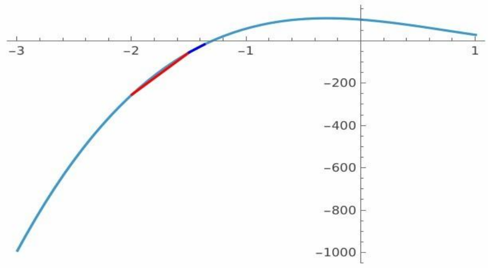
Начальный отрезок приближения: *[-2; -1.5]*. Полученное значение: *-1.30769* за 4 итерации.

```
{root, iterations} = chord[f, -2, -1.5, 1.*10^-3]
{-1.30769, 4}
```

в) Визуализация первых двух приближений (первое приближение показано красным цветом, второе - синим):

```
x0 = -2;
x1 = -1.5;
x2 = x1 - f[x1] (x1 - x0) / (f[x1] - f[x0]);

Show[
     Plot[f[x], {x, -3, 1}, PlotRange → All, AxesOrigin → {0, 0}],
     Graphics[{
            Red, Thick, Line[{{x0, f[x0]}, {x1, f[x1]}}],
            Blue, Thick, Line[{{x1, f[x1]}, {x2, f[x2]}}]
       }]
]
```



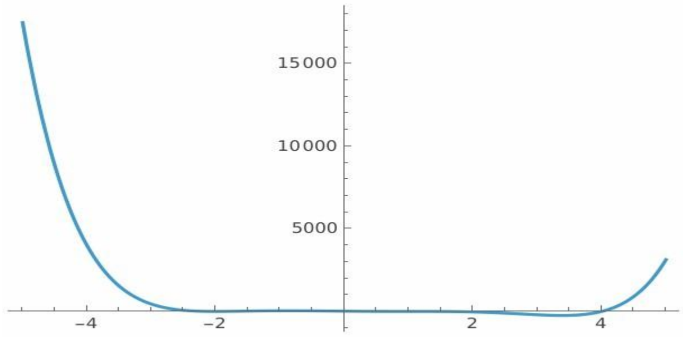
Задание 2. Отделить графически и найти с помощью функций **Solve**, **NSolve**, **Roots**, **FindRoot** корни алгебраического уравнения *f(x)=0*. Разложить многочлен *f(x)* на множители с помощью функции **Factor**.

Функция *f(x)*:
$f(x)=x^6-3x^5-9x^4+19x^3+12x^2-36x+16$

```
f[x_] := x^6 - 3 x^5 - 9 x^4 + 19 x^3 + 12 x^2 - 36 x + 16
```

График функции *f(x)* (со всеми нулями):

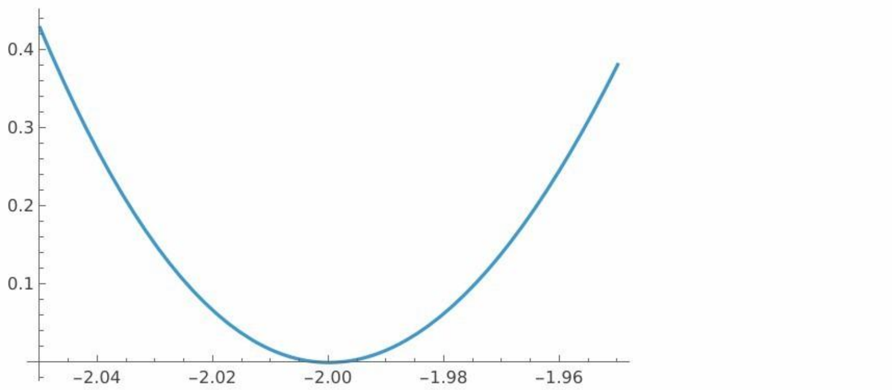
```
Plot[f[x], {x, -5, 5}, PlotRange → All, AxesOrigin → {0, 0}]
```



а) Нули функции *f(x)*.

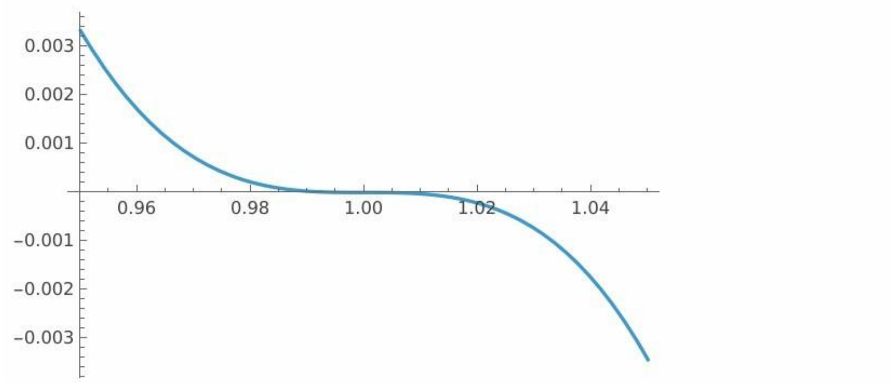
x=-2 (корень кратности 2):

```
Plot[f[x], {x, -2.05, -1.95}, PlotRange → All, AxesOrigin → {-2.05, 0}]
график функции                  отобража…   всё   точка пересечения осей
```



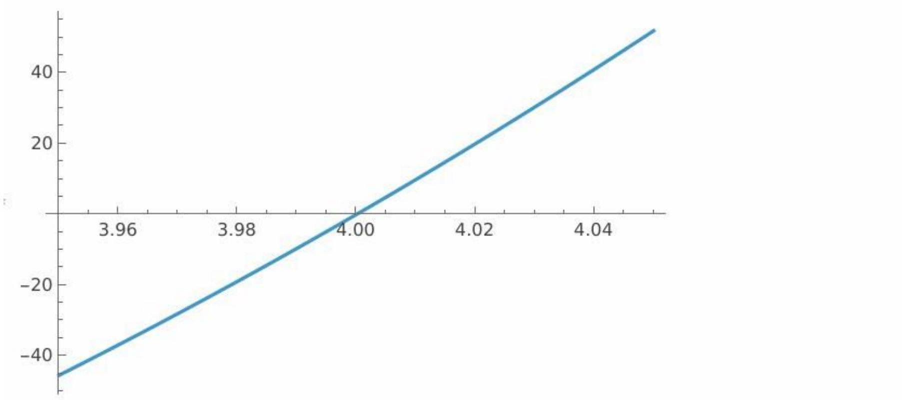
x=1 (корень кратности 3):

```
Plot[f[x], {x, 0.95, 1.05}, PlotRange → All, AxesOrigin → {0.95, 0}]
график функции                  отобража…   всё   точка пересечения осей
```



x=4:

```
Plot[f[x], {x, 3.95, 4.05}, PlotRange → All, AxesOrigin → {3.95, 0}]
график функции                  отобража…   всё   точка пересечения осей
```



б.1,2,3) Решение при помощи функций **Solve**, **NSolve**, **Roots**:

```
Solve[f[x] == 0, x]
решить уравнения

{{x → -2}, {x → -2}, {x → 1}, {x → 1}, {x → 1}, {x → 4}}

NSolve[f[x] == 0, x]
численное решение уравнений

{{x → -2.}, {x → -2.}, {x → 1.}, {x → 1.}, {x → 1.}, {x → 4.}}

Roots[f[x] == 0, x]
корни многочлена

x == -2 || x == -2 || x == 1 || x == 1 || x == 1 || x == 4
```

б.4) Решение при помощи функции **FindRoot** (от начального приближения):

```
FindRoot[f[x] == 0, {x, -2.2}]
найти корень

{x → -2.}

FindRoot[f[x] == 0, {x, -1.8}]
найти корень

{x → -2.}

FindRoot[f[x] == 0, {x, 0.9}]
найти корень

{x → 0.999996}

FindRoot[f[x] == 0, {x, 1.05}]
найти корень

{x → 1.}

FindRoot[f[x] == 0, {x, 1.}]
найти корень

{x → 1.}

FindRoot[f[x] == 0, {x, 3.9}]
найти корень

{x → 4.}
```

в) Разложение на множители при помощи **Factor**:

```
Factor[f[x]]
факторизовать
```

$$(-4+x)\,(-1+x)^3\,(2+x)^2$$

Задание 3. Отделить графически корни трансцендентного уравнения $f(x)=0$. Найдите один из них с точностью $eps=10^{-3}$ а) методом Ньютона; б) методом секущих. Укажите потребовавшееся число итераций.

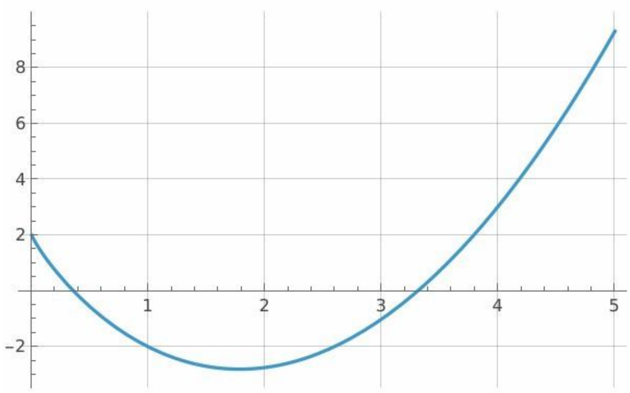
$f(x)=x\log_3 x+x^2-5x+2$

```
f[x_] := x * Log[3, x] + x^2 - 5 x + 2
                натуральный логарифм
```

График функции $f(x)$ (с нулями):

```
Plot[f[x], {x, 0, 5}, PlotRange → All, AxesOrigin → {0, 0}, GridLines → Automatic]
график функции          отобража···  всё  точка пересечения о···  линии коо···  автоматичес
```

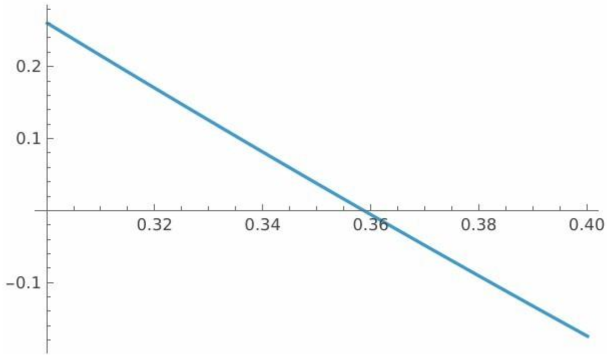
0.0380438



Нули функции:

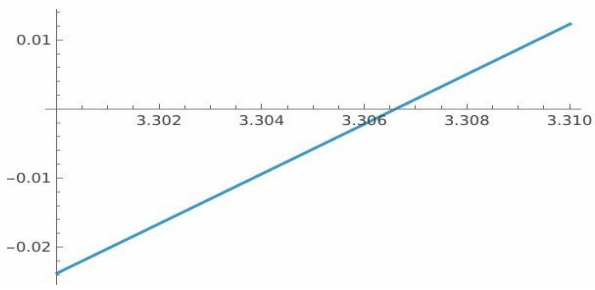
$x\approx 0.36$:

```
Plot[f[x], {x, 0.3, 0.4}, PlotRange → All, AxesOrigin → {0.3, 0}]
график функции             отобража···  всё  точка пересечения осей
```



$x\approx 3.3$:

```
Plot[f[x], {x, 3.3, 3.31}, PlotRange → All, AxesOrigin → {3.3, 0}]
график функции             отобража···  всё  точка пересечения осей
```



а) Нахождение маленького корня (≈0.36) при помощи метода Ньютона.

Формула метода Ньютона:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Приближение корня (в коде *df* — производная функции *f(x)*):

```
df[x_] := (Log[x] + 1) / Log[3] + 2 x - 5
         |натуральны··· |натуральный логар

x = 0.35;
eps = 10 ^ -3;
iter = 0;

While[True,
|цик···  |истина
      iter ++;
      xNew = x - f[x] / df[x];
      If[Abs[xNew - x] < eps, Break[]];
      |···  |абсолютное значение  |прервать ци
      x = xNew;
]
```

Ответ: *0.358796* за 2 итерации:

```
{xNew, iter}
{0.358796, 2}
```

б) Нахождение большого корня (≈3.3) при помощи метода секущих. Отрезок: *[3.3; 3.4]*.

```
x0 = 3.3; x1 = 3.4;
eps = 10 ^ -3;
iter = 0;

While[True,
|цик···  |истина
      iter ++;
      f0 = f[x0];
      f1 = f[x1];
      x2 = x1 - f1 * (x1 - x0) / (f1 - f0);
      If[Abs[x2 - x1] < eps, Break[]];
      |···  |абсолютное значе···  |прервать ци
      x0 = x1;
      x1 = x2;
]]
```

Ответ: *3.30657* за 2 итерации:

```
{x2, iter}
{3.30657, 2}
```

Задание 4. Привести уравнение из задания 3 к виду, пригодному для итераций. Найти его корни методом простых итераций с точностью $eps=10^{-3}$. Указать число потребовавшихся итераций.

Для разных корней были использованы разные преобразования функции. Для маленького корня вынесем x за скобки, перенесём 2 в правую часть уравнения $f(x)=0$ и разделим обе части на $(x+\log_3 x-5)$, после чего получим:

$x=\varphi_1(x)=-2/(x+\log_3 x-5)$

```
phi1[x_] := -2/(x + Log[x]/Log[3] - 5)
                    натур…  натуральный логарифм
```

Для большого корня разделим уравнение на x:

$x=\varphi_2(x)=5-\log_3 x-2/x$

```
phi2[x_] := 5 - Log[x]/Log[3] - 2/x
                натур…  натуральный логарифм
```

Для метода простых итераций была определена функция **FixedPointIterate**.

```
FixedPointIterate[phi_, x0_] := Module[
                                программный модуль
   {x = x0, xn, i = 0},
      While[True,
      цик…  истина
            i++;
            xn = phi[x];
            If[Abs[xn - x] < eps, Return[{xn, i}]];
            …  абсолютное знач…    вернуть управление
            x = xn;
         ];
   ];
```

Нахождение малого корня (начальное приближение *0.35*). Ответ: *0.358692*, итерации: 3:

```
FixedPointIterate[phi1, 0.35]

{0.358692, 3}
```

Нахождение большого корня (начальное приближение *3.3*). Ответ: *3.30652*, итерации: 2:

```
FixedPointIterate[phi2, 3.3]

{3.30652, 2}
```

Задание 5. Решить уравнение из задания 3 с помощью функций **Solve**, **NSolve**, **FindRoot**.

```
Solve[f[x] == 0, x]
решить уравнения

••• Solve: This system cannot be solved with the methods available to Solve. Try Reduce or FindInstance instead.
```

$$\text{Solve}\left[2 - 5\,x + x^2 + \frac{x\,\text{Log}[x]}{\text{Log}[3]} == 0,\ x\right]$$

```
NSolve[f[x] == 0 && x > 0, x]
численное решение уравнений

{{x → 0.358796}, {x → 3.30658}}

FindRoot[f[x] == 0, {x, 0.35}]
найти корень

{x → 0.358796}

FindRoot[f[x] == 0, {x, 3.3}]
найти корень

{x → 3.30658}
```

Поскольку уравнение *f(x)=0* является трансцендентным, функции **Solve** не удаётся решить данное уравнение (**Solve** ищет аналитическое решение, однако для уравнения *f(x)=0* его нет).

Для **NSolve** дополнительно необходимо указать условие *x>0* для того, чтобы **NSolve** не начала искать корни там, где функция *f(x)* не определена или комплексна (неположительные значения).

NSolve и **FindRoot** находят корни исправно, в отличие от **Solve**.

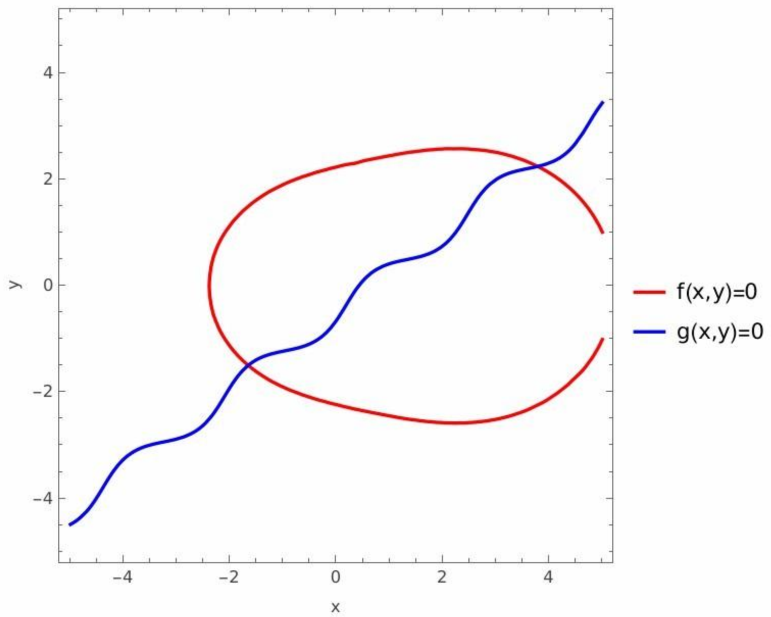
Задание 6. Дана система двух нелинейных уравнений *f(x,y)=0*, *g(x,y)=0*. Изобразить на одном чертеже кривые *f(x,y)=0*, *g(x,y)=0*, и решить данную систему.

*f(x,y)=(x^2+y^2-2*x)^2-8(x^2-y^2)-65*
*g(x,y)=sin(2*x+y)-4*y+3*x-2*

```
f[x_, y_] := (x^2 + y^2 - 2 x)^2 - 8 (x^2 - y^2) - 65
g[x_, y_] := Sin[2 x + y] - 4 y + 3 x - 2
```

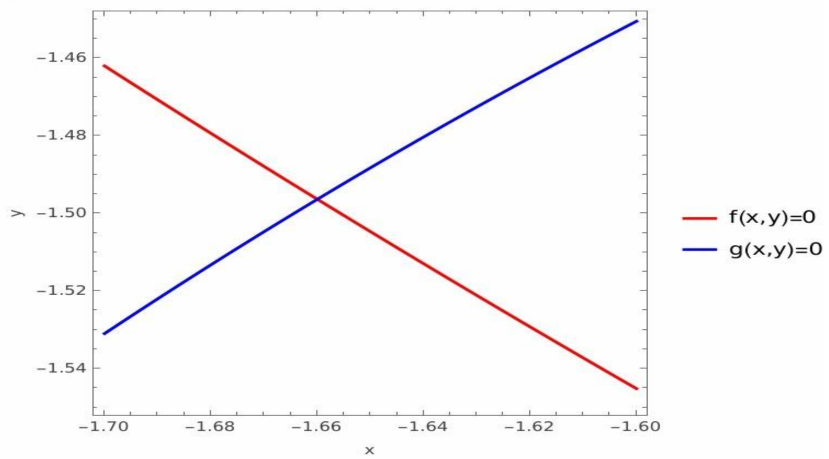
График решения системы (следующая страница):

```
ContourPlot[
контурный график
    {f[x, y] == 0, g[x, y] == 0},
    {x, -5, 5}, {y, -5, 5},
    ContourStyle → {Red, Blue},
    контурный стиль  кр⋯  синий
    PlotLegends → {"f(x,y)=0", "g(x,y)=0"},
    легенды графика
    FrameLabel → {"x", "y"}
    пометка для обрамления
]
```



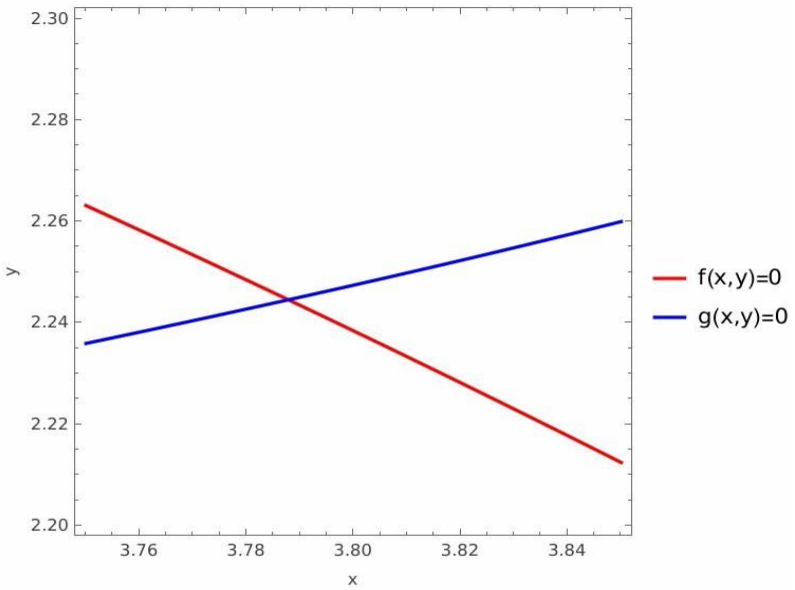
Пересечения *f(x, y)* и *g(x, y)*:
x≈-1.66, y≈-1.5:

```
ContourPlot[
контурный график
    {f[x, y] == 0, g[x, y] == 0},
    {x, -1.7, -1.6}, {y, -1.55, -1.45},
    ContourStyle → {Red, Blue},
    контурный стиль  кр⋯  синий
    PlotLegends → {"f(x,y)=0", "g(x,y)=0"},
    легенды графика
    FrameLabel → {"x", "y"}
    пометка для обрамления
]
```



x≈*3.79, y≈2.24:*

```
ContourPlot[
контурный график
    {f[x, y] == 0, g[x, y] == 0},
    {x, 3.75, 3.85}, {y, 2.2, 2.3},
    ContourStyle → {Red, Blue},
    контурный стиль  кр⋯  синий
    PlotLegends → {"f(x,y)=0", "g(x,y)=0"},
    легенды графика
    FrameLabel → {"x", "y"}
    пометка для обрамления
]
```



Решение системы при помощи функции **FindRoot**.

```
FindRoot[{f[x, y] == 0, g[x, y] == 0}, {{x, -1.7}, {y, -1.55}}]
найти корень
{x → -1.65999, y → -1.49634}

FindRoot[{f[x, y] == 0, g[x, y] == 0}, {{x, 3.75}, {y, 2.2}}]
найти корень
{x → 3.78782, y → 2.24456}
```

Вывод: Для решения нелинейных уравнений были рассмотрены такие методы, как: метод секущих (и метод хорд как его частный случай), метод Ньютона и метод простых итераций. Методы продемонстрировали примерно одинаковую эффективность на показанных в работе входных данных. Были рассмотрены функции **Solve**, **NSolve**, **Roots** и **FindRoot** для решения нелинейных уравнений. Было выявлено, что функция **Solve** непригодна для решения трансцендентных уравнений.