In[ ]:=

```
(*Выполнил: Самута Даниил
  Группа 221703
  Вариант: 8*)
(*Задание 1*)

func1 [i_, j_] := Which[i > j, 1, i == j, i + 1, i < j, 2]
```
условный оператор с множественными ветвями

```
A = Array[func1, {7, 7}] (*Задаем массив по данным формулам*)
```
массив

Out[ ]=

```
{{2, 2, 2, 2, 2, 2, 2}, {1, 3, 2, 2, 2, 2, 2}, {1, 1, 4, 2, 2, 2, 2},
 {1, 1, 1, 5, 2, 2, 2}, {1, 1, 1, 1, 6, 2, 2}, {1, 1, 1, 1, 1, 7, 2}, {1, 1, 1, 1, 1, 1, 8}}
```

In[ ]:=

$$\texttt{func2[i\_] := } 16\,i - i^2$$

```
B = Array[func2, {7}] (*Задаем массив по данным формулам*)
```
массив

Out[ ]=

```
{15, 28, 39, 48, 55, 60, 63}
```

In[ ]:=

```
InversedA = Inverse[A] (*Находим обратную матрицу для A*)
```
обратная матрица

Out[ ]=

$$\left\{\left\{\frac{13}{14}, -\frac{1}{2}, -\frac{1}{6}, -\frac{1}{12}, -\frac{1}{20}, -\frac{1}{30}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, \frac{1}{2}, -\frac{1}{6}, -\frac{1}{12}, -\frac{1}{20}, -\frac{1}{30}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, 0, \frac{1}{3}, -\frac{1}{12}, -\frac{1}{20}, -\frac{1}{30}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, 0, 0, \frac{1}{4}, -\frac{1}{20}, -\frac{1}{30}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, 0, 0, 0, \frac{1}{5}, -\frac{1}{30}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, 0, 0, 0, 0, \frac{1}{6}, -\frac{1}{42}\right\}, \left\{-\frac{1}{14}, 0, 0, 0, 0, 0, \frac{1}{7}\right\}\right\}$$

In[ ]:=

$$\texttt{t1 = Table}\left[\sum_{j=1}^{7} \texttt{Abs[A[[i, j]]], \{i, 1, 7\}}\right]$$
таблица абсолютное значение

```
(*Вектор,координатами которого являются суммы
  взятых по модулю элементов каждой строки матрицы A.*)
```

Out[ ]=

```
{14, 14, 14, 14, 14, 14, 14}
```

In[ ]:=

$$\texttt{t2 = Table}\left[\sum_{j=1}^{7} \texttt{Abs[InversedA[[i, j]]], \{i, 1, 7\}}\right]$$
таблица абсолютное значение

```
(*Вектор,координатами которого являются суммы
  взятых по модулю элементов каждой строки матрицы обратной для A.*)
```

Out[ ]=

$$\left\{\frac{25}{14}, \frac{13}{14}, \frac{25}{42}, \frac{3}{7}, \frac{23}{70}, \frac{11}{42}, \frac{3}{14}\right\}$$

In[∘]:=
```
norm1 = Max[t1] (*Находим норму-максимум для матрицы A*)
        максимум
```
Out[∘]=
```
14
```
In[∘]:=
```
norm2 = Max[t2] (*Находим норму-максимум для обратной матрицы для A*)
        максимум
```
Out[∘]=

$$\frac{25}{14}$$

In[∘]:=
```
condA = norm1 * norm2 (*Число обусловленности матрицы A в норме-максимум*)
```
Out[∘]=
```
25
```
In[∘]:=
```
solve = N[LinearSolve[A, B]] (*решение данного уравнения*)
        ·· решить линейные уравнения
```
Out[∘]=
```
{-16.8214, -3.82143, 1.67857, 4.67857, 6.42857, 7.42857, 7.92857}
```
In[∘]:=
```
dB1 = B * 0
dB1[[7]] = 0.0001 (*Вектор для изменения правой части вектора B*)
```
Out[∘]=
```
{0, 0, 0, 0, 0, 0, 0}
```
Out[∘]=
```
0.0001
```
In[∘]:=
```
dB2 = B * 0
dB2[[7]] = 0.001 (*Вектор для изменения правой части вектора B*)
```
Out[∘]=
```
{0, 0, 0, 0, 0, 0, 0}
```
Out[∘]=
```
0.001
```
In[∘]:=
```
dB3 = B * 0
dB3[[7]] = 0.01 (*Вектор для изменения правой части вектора B*)
```
Out[∘]=
```
{0, 0, 0, 0, 0, 0, 0}
```
Out[∘]=
```
0.01
```
In[∘]:=
```
solve1 = N[LinearSolve[A, B + dB1]] (*Решение 1-ой возмущенной системы*)
         ·· решить линейные уравнения
```
Out[∘]=
```
{-16.8214, -3.82143, 1.67857, 4.67857, 6.42857, 7.42857, 7.92859}
```
In[∘]:=
```
solve2 = N[LinearSolve[A, B + dB2]] (*Решение 2-ой возмущенной системы*)
         ·· решить линейные уравнения
```
Out[∘]=
```
{-16.8215, -3.82145, 1.67855, 4.67855, 6.42855, 7.42855, 7.92871}
```

In[ ]:=
```
solve3 = N[LinearSolve[A, B + dB3]] (*Решение 3-ей возмущенной системы*)
```
решить линейные уравнения

Out[ ]=
{-16.8217, -3.82167, 1.67833, 4.67833, 6.42833, 7.42833, 7.93}

In[ ]:=
$$pr1 = condA * \frac{Norm[dB1, 1]}{Norm[B + dB1, 1]}$$ (*Прогнозируемая предельная относительная погрешность решения 1-ой возмущенной системы *)

Out[ ]=
8.11688×10^{-6}

In[ ]:=
$$pr2 = condA * \frac{Norm[dB2, 1]}{Norm[B + dB2, 1]}$$ (*Прогнозируемая предельная относительная погрешность решения 2-ой возмущенной системы *)

Out[ ]=
0.0000811686

In[ ]:=
$$pr3 = condA * \frac{Norm[dB3, 1]}{Norm[B + dB3, 1]}$$ (*Прогнозируемая предельная относительная погрешность решения 3-ей возмущенной системы *)

Out[ ]=
0.000811662

In[ ]:=
```
deltaSolve1 = solve1 - solve (*вектор ошибки 1-ой возмущенной системы*)
```

Out[ ]=
$\{-2.38095 \times 10^{-6}, -2.38095 \times 10^{-6}, -2.38095 \times 10^{-6}, -2.38095 \times 10^{-6}, -2.38095 \times 10^{-6}, -2.38095 \times 10^{-6}, 0.0000142857\}$

In[ ]:=
```
deltaSolve2 = solve2 - solve (*вектор ошибки 2-ой возмущенной системы*)
```

Out[ ]=
{-0.0000238095, -0.0000238095, -0.0000238095,
-0.0000238095, -0.0000238095, -0.0000238095, 0.000142857}

In[ ]:=
```
deltaSolve3 = solve3 - solve (*вектор ошибки 3-ей возмущенной системы*)
```

Out[ ]=
{-0.000238095, -0.000238095, -0.000238095,
-0.000238095, -0.000238095, -0.000238095, 0.00142857}

In[ ]:=
```
nd1 = Norm[deltaSolve1, 1] (*Абсолютная погрешность решения 1-ой возмущенной системы*)
```
норма

Out[ ]=
0.0000285714

In[ ]:=
```
nd2 = Norm[deltaSolve2, 1] (*Абсолютная погрешность решения 2-ой возмущенной системы*)
```
норма

Out[ ]=
0.000285714

In[ ]:=
```
nd3 = Norm[deltaSolve3, 1] (*Абсолютная погрешность решения 3-ей возмущенной системы*)
```
норма

Out[ ]=
0.00285714

In[ ]:= nrd1 = $\frac{\text{nd1}}{\text{Norm[solve1, 1]}}$ (*Относительная погрешность решения 1-ой возмущенной системы*)

Out[ ]=

5.85651×10^{-7}

In[ ]:= nrd2 = $\frac{\text{nd2}}{\text{Norm[solve2, 1]}}$ (*Относительная погрешность решения 2-ой возмущенной системы*)

Out[ ]=

5.8565×10^{-6}

In[ ]:= nrd3 = $\frac{\text{nd3}}{\text{Norm[solve3, 1]}}$ (*Относительная погрешность решения 3-ей возмущенной системы*)

Out[ ]=

0.000058564

In[ ]:= func1 [i_, j_] := $\frac{1}{i + j - 1}$

A = Array[func1, {7, 7}] (*Задаем массив по данным формулам*)
массив

Out[ ]=

$\left\{\left\{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{7}\right\}, \left\{\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{7}, \frac{1}{8}\right\}, \left\{\frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{7}, \frac{1}{8}, \frac{1}{9}\right\}, \left\{\frac{1}{4}, \frac{1}{5}, \frac{1}{6}, \frac{1}{7}, \frac{1}{8}, \frac{1}{9}, \frac{1}{10}\right\}, \left\{\frac{1}{5}, \frac{1}{6}, \frac{1}{7}, \frac{1}{8}, \frac{1}{9}, \frac{1}{10}, \frac{1}{11}\right\}, \left\{\frac{1}{6}, \frac{1}{7}, \frac{1}{8}, \frac{1}{9}, \frac{1}{10}, \frac{1}{11}, \frac{1}{12}\right\}, \left\{\frac{1}{7}, \frac{1}{8}, \frac{1}{9}, \frac{1}{10}, \frac{1}{11}, \frac{1}{12}, \frac{1}{13}\right\}\right\}$

In[ ]:= func2[i_] := 3 i - 16

B = Array[func2, {7}] (*Задаем массив по данным формулам*)
массив

Out[ ]=

{-13, -10, -7, -4, -1, 2, 5}

In[ ]:= InversedA = Inverse[A] (*Находим обратную матрицу для A*)
обратная матрица

Out[ ]=

{{49, -1176, 8820, -29 400, 48 510, -38 808, 12 012},
{-1176, 37 632, -317 520, 1 128 960, -1 940 400, 1 596 672, -504 504},
{8820, -317 520, 2 857 680, -10 584 000, 18 711 000, -15 717 240, 5 045 040},
{-29 400, 1 128 960, -10 584 000, 40 320 000, -72 765 000, 62 092 800, -20 180 160},
{48 510, -1 940 400, 18 711 000, -72 765 000, 133 402 500, -115 259 760, 37 837 800},
{-38 808, 1 596 672, -15 717 240, 62 092 800, -115 259 760, 100 590 336, -33 297 264},
{12 012, -504 504, 5 045 040, -20 180 160, 37 837 800, -33 297 264, 11 099 088}}

In[ ]:=
```
t1 = Table[Sum[Abs[A[[i, j]]], {j, 1, 7}], {i, 1, 7}]
```
таблица абсолютное значение

(*Вектор,координатами которого являются суммы взятых по модулю элементов каждой строки матрицы A.*)

Out[ ]=

$$\left\{\frac{363}{140}, \frac{481}{280}, \frac{3349}{2520}, \frac{2761}{2520}, \frac{25961}{27720}, \frac{22727}{27720}, \frac{263111}{360360}\right\}$$

In[ ]:=
```
t2 = Table[Sum[Abs[InversedA[[i, j]]], {j, 1, 7}], {i, 1, 7}]
```
таблица абсолютное значение

(*Вектор,координатами которого являются суммы взятых по модулю элементов каждой строки матрицы обратной для A.*)

Out[ ]=

{138 775, 5 526 864, 53 241 300, 207 100 320, 379 964 970, 328 592 880, 107 975 868}

In[ ]:=
```
norm1 = Max[t1] (*Находим норму-максимум для матрицы A*)
```
максимум

Out[ ]=

$$\frac{363}{140}$$

In[ ]:=
```
norm2 = Max[t2] (*Находим норму-максимум для обратной матрицы для A*)
```
максимум

Out[ ]=

379 964 970

In[ ]:=
```
condA = norm1 * norm2 (*Число обусловленности матрицы A в норме-максимум*)
```

Out[ ]=

$$\frac{1970389773}{2}$$

In[ ]:=
```
solve = N[LinearSolve[A, B]] (*решение данного уравнения*)
```
решить линейные уравнения

Out[ ]=

$\{917., -43008., 472500., -2.0496 \times 10^6, 4.12335 \times 10^6, -3.85862 \times 10^6, 1.35736 \times 10^6\}$

In[ ]:=
```
dB1 = B * 0
dB1[[7]] = 0.0001 (*Вектор для изменения правой части вектора B*)
```

Out[ ]=

{0, 0, 0, 0, 0, 0, 0}

Out[ ]=

0.0001

In[ ]:=
```
dB2 = B * 0
dB2[[7]] = 0.001 (*Вектор для изменения правой части вектора B*)
```

Out[ ]=

{0, 0, 0, 0, 0, 0, 0}

Out[ ]=

0.001

In[ ]:=
```
dB3 = B * 0
dB3[[7]] = 0.01 (*Вектор для изменения правой части вектора B*)
```
Out[ ]=
```
{0, 0, 0, 0, 0, 0, 0}
```
Out[ ]=
```
0.01
```
In[ ]:=
```
solve1 = N[LinearSolve[A, B + dB1]] (*Решение 1-ой возмущенной системы*)
```
решить линейные уравнения

Out[ ]=

$\{918.201, -43058.5, 473005., -2.05162 \times 10^6, 4.12713 \times 10^6, -3.86195 \times 10^6, 1.35847 \times 10^6\}$

In[ ]:=
```
solve2 = N[LinearSolve[A, B + dB2]] (*Решение 2-ой возмущенной системы*)
```
решить линейные уравнения

Out[ ]=

$\{929.012, -43512.5, 477545., -2.06978 \times 10^6, 4.16119 \times 10^6, -3.89192 \times 10^6, 1.36846 \times 10^6\}$

In[ ]:=
```
solve3 = N[LinearSolve[A, B + dB3]] (*Решение 3-ей возмущенной системы*)
```
решить линейные уравнения

Out[ ]=

$\{1037.12, -48053., 522950., -2.2514 \times 10^6, 4.50173 \times 10^6, -4.1916 \times 10^6, 1.46835 \times 10^6\}$

In[ ]:=

$\text{pr1} = \text{condA} * \frac{\text{Norm[dB1, 1]}}{\text{Norm[B + dB1, 1]}}$ (*Прогнозируемая предельная относительная погрешность решения 1-ой возмущенной системы *)

Out[ ]=
```
2345.7
```
In[ ]:=

$\text{pr2} = \text{condA} * \frac{\text{Norm[dB2, 1]}}{\text{Norm[B + dB2, 1]}}$ (*Прогнозируемая предельная относительная погрешность решения 2-ой возмущенной системы *)

Out[ ]=
```
23456.5
```
In[ ]:=

$\text{pr3} = \text{condA} * \frac{\text{Norm[dB3, 1]}}{\text{Norm[B + dB3, 1]}}$ (*Прогнозируемая предельная относительная погрешность решения 3-ей возмущенной системы *)

Out[ ]=
```
234514.
```
In[ ]:=
```
deltaSolve1 = solve1 - solve (*вектор ошибки 1-ой возмущенной системы*)
```
Out[ ]=
```
{1.2012, -50.4504, 504.504, -2018.02, 3783.78, -3329.73, 1109.91}
```
In[ ]:=
```
deltaSolve2 = solve2 - solve (*вектор ошибки 2-ой возмущенной системы*)
```
Out[ ]=
```
{12.012, -504.504, 5045.04, -20180.2, 37837.8, -33297.3, 11099.1}
```
In[ ]:=
```
deltaSolve3 = solve3 - solve (*вектор ошибки 3-ей возмущенной системы*)
```
Out[ ]=
```
{120.12, -5045.04, 50450.4, -201802., 378378., -332973., 110991.}
```

```
In[ ]:= nd1 = Norm[deltaSolve1, 1] (*Абсолютная погрешность решения 1-ой возмущенной системы*)
              норма
Out[ ]=
        10 797.6

In[ ]:= nd2 = Norm[deltaSolve2, 1] (*Абсолютная погрешность решения 2-ой возмущенной системы*)
              норма
Out[ ]=
        107 976.
```

```
In[ ]:= nd3 = Norm[deltaSolve3, 1] (*Абсолютная погрешность решения 3-ей возмущенной системы*)
              норма
Out[ ]=
```

1.07976×10^{6}

In[ ]:= $\text{nrd1} = \dfrac{\text{nd1}}{\text{Norm[solve1, 1]}}$ (*Относительная погрешность решения 1-ой возмущенной системы*)

Out[ ]=

0.000906131

In[ ]:= $\text{nrd2} = \dfrac{\text{nd2}}{\text{Norm[solve2, 1]}}$ (*Относительная погрешность решения 2-ой возмущенной системы*)

Out[ ]=

0.008988

In[ ]:= $\text{nrd3} = \dfrac{\text{nd3}}{\text{Norm[solve3, 1]}}$ (*Относительная погрешность решения 3-ей возмущенной системы*)

Out[ ]=

0.0831536

In[ ]:=

```
In[ ]:= (*Вывод: из полученных результаов можно выявить следующую зависимость:
         чем больше число обусловленности,
         тем больше относительная погрешность*)
```

In[ ]:= (*Задание 2*)

In[ ]:=

$$A = \begin{pmatrix} 7 & 2 & 0 & 0 & 0 \\ 1 & 12 & 3 & 0 & 0 \\ 0 & 2 & -9 & -4 & 0 \\ 0 & 0 & 1 & 13 & 2 \\ 0 & 0 & 0 & 3 & 15 \end{pmatrix}$$

```
B = {-14, 7, -23, -8, 12}
Out[ ]=
{{7, 2, 0, 0, 0}, {1, 12, 3, 0, 0}, {0, 2, -9, -4, 0}, {0, 0, 1, 13, 2}, {0, 0, 0, 3, 15}}
Out[ ]=
{-14, 7, -23, -8, 12}
In[ ]:=
```

In[∘]:=
```
(*Матрица решений*)
```

In[∘]:=
```
X = B * 0
```

Out[∘]=
```
{0, 0, 0, 0, 0}
```

In[∘]:=
```
(*Прогоночные коэффициенты*)
```

In[∘]:=
```
L = B * 0
M = B * 0
```

Out[∘]=
```
{0, 0, 0, 0, 0}
```

Out[∘]=
```
{0, 0, 0, 0, 0}
```

In[∘]:=
```
{0, 0, 0, 0, 0}
(*Для первой 0-й строки*)
```

Out[∘]=
```
{0, 0, 0, 0, 0}
```

In[∘]:=
```
L⟦1⟧ = N[A⟦1, 2⟧ / (-A⟦1, 1⟧)]
       численное приближение
```

Out[∘]=
```
-0.285714
```

In[∘]:=
```
M⟦1⟧ = N[(-B⟦1⟧) / (-A⟦1, 1⟧)]
       численное приближение
```

Out[∘]=
```
-2.
```

In[∘]:=
```
M
L
```

Out[∘]=
```
{-2., 0, 0, 0, 0}
```

Out[∘]=
```
{-0.285714, 0, 0, 0, 0}
```

In[∘]:=

In[∘]:=
```
(*Заполняем за исключением 1-ой и последней строки*)
```

In[∘]:=
```
For[i = 2, i ≤ 4, i++,
цикл ДЛЯ
 L⟦i⟧ = N[A⟦i, i + 1⟧ / (-A⟦i, i⟧ - A⟦i, i - 1⟧ * L⟦i - 1⟧)];
        численное приближение
 M⟦i⟧ = N[(A⟦i, i - 1⟧ * M⟦i - 1⟧ - B⟦i⟧) / (-A⟦i, i⟧ - A⟦i, i - 1⟧ * L⟦i - 1⟧)]
        численное приближение
]
```

In[∘]:=

In[∘]:=
```
L
```

Out[∘]=
```
{-0.285714, -0.256098, -0.420513, -0.158989, 0}
```

In[ ]:=
```
M
```
Out[ ]=
```
{-2., 0.768293, 2.57949, -0.841011, 0}
```
In[ ]:=

In[ ]:=
```
(*Заполняем последнюю строку*)
```
In[ ]:=
```
L[[5]] = 0
M[[5]] = N[(A[[5, 4]] * M[[4]] - B[[5]]) / (-A[[5, 5]] - A[[5, 4]] * L[[4]])]
         численное приближение
```
Out[ ]=
```
0
```
Out[ ]=
```
1.
```
In[ ]:=
```
L
M
```
Out[ ]=
```
{-0.285714, -0.256098, -0.420513, -0.158989, 0}
```
Out[ ]=
```
{-2., 0.768293, 2.57949, -0.841011, 1.}
```
In[ ]:=
```
(*Обратная прогонка*)
```
In[ ]:=
```
X[[5]] = M[[5]]
```
Out[ ]=
```
1.
```
In[ ]:=
```
For[i = 5, i > 1, i--,
цикл ДЛЯ
 X[[i - 1]] = N[L[[i - 1]] * X[[i]] + M[[i - 1]]]
              численное приближение
]
```
In[ ]:=
```
(*Корни уравнения*)
```
In[ ]:=
```
X
```
Out[ ]=

$\{-2., 1.11022 \times 10^{-16}, 3., -1., 1.\}$

In[ ]:=

```
(*Задание 3*)
```
In[ ]:=
```
(*пункт а)*)
```
In[ ]:=
```
(*Метод Якоби*)
```

In[ ]:=
```
n = 10
f1[i_, j_] := Which[i ≠ j, 1, i == j, 2 * n]
              условный оператор с множественными ветвями
f2[i_] := (2 * n - 1) * i + n * (n + 1) / 2 + (3 * n - 1) * 7
A = Array[f1, {n, n}]
    массив
B = Array[f2, n] (*Задаем матрицу A и вектор решений B*)
    массив
```

Out[ ]=
```
10
```

Out[ ]=
```
{{20, 1, 1, 1, 1, 1, 1, 1, 1, 1}, {1, 20, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 20, 1, 1, 1, 1, 1, 1, 1}, {1, 1, 1, 20, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 20, 1, 1, 1, 1, 1}, {1, 1, 1, 1, 1, 20, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 20, 1, 1, 1}, {1, 1, 1, 1, 1, 1, 1, 20, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 20, 1}, {1, 1, 1, 1, 1, 1, 1, 1, 1, 20}}
```

Out[ ]=
```
{277, 296, 315, 334, 353, 372, 391, 410, 429, 448}
```

In[ ]:=
```
x = ConstantArray[1, n]
    постоянный массив
```

Out[ ]=
```
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

In[ ]:=
```
x = {1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

Out[ ]=
```
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

In[ ]:=
```
numberOfIter = 0
```

Out[ ]=
```
0
```

In[ ]:=
```
maxIter = 100
eps = 0.001(*Задаем начальное значение кол-ва итераций,
 максимальное число итераций, приближение*)
```

Out[ ]=
```
100
```

Out[ ]=
```
0.001
```

In[ ]:=
```
While[numberOfIter < maxIter, (*Находим корни уравнеия по методу Якоби*)
цикл–пока
 xPrev = x;
 For[k = 1, k ≤ n, k++,
 цикл ДЛЯ
  S = 0;
  For[j = 1, j ≤ n, j++,
  цикл ДЛЯ

    If[
    условный оператор
     j ≠ k, S = N[S + A〚k, j〛 * x〚j〛]];
                 численное приближение
  ];
  x〚k〛 = N[B〚k〛 / A〚k, k〛 - S / A〚k, k〛];
          численное приближение
 ];
 If[Sqrt[Norm[x - xPrev] / Norm[x]] < 0.001, Break[];];
 … ква… норма            норма              прервать цикл
 numberOfIter++]
```

In[ ]:=
```
numberOfIter (*Смотрим, какое число итераций*)
```

Out[ ]=

7

In[ ]:=
```
Rationalize[x] (*Выводим корни уравнения*)
найти рациональное приближение
```

Out[ ]=

{8., 9., 10., 11., 12., 13., 14., 15., 16., 17.}

In[ ]:=
```
(*Метод Зейделя*)
```

In[ ]:=
```
x = B
g = 0
y = B
s = 0
numberOfIter = 0(*Задаем начальные значения*)
```

Out[ ]=

{277, 296, 315, 334, 353, 372, 391, 410, 429, 448}

Out[ ]=

0

Out[ ]=

{277, 296, 315, 334, 353, 372, 391, 410, 429, 448}

Out[ ]=

0

Out[ ]=

0

In[ ]:=
```
While[numberOfIter < maxIter, (*Находм корни используя метод Зейделя*)
цикл–пока
 s = 0;
 xPrev = x;
 For[i = 1, i ≤ n, i++,
 цикл ДЛЯ
  y⟦i⟧ = (B⟦i⟧ / A⟦i, i⟧);
  For[j = 1, j ≤ n, j++,
  цикл ДЛЯ
       If[
       условный оператор
     j ≠ i, y⟦i⟧ = y⟦i⟧ - ((A⟦i, j⟧ / A⟦i, i⟧) * x⟦j⟧);
     x⟦i⟧ = y⟦i⟧;
    ];

  ];
 ];
 If[Norm[x - xPrev] / Norm[x] < eps, Break[];];
 ⋯  норма              норма          прервать цикл
 numberOfIter++]
```

In[ ]:=
```
N[x] (*Выводим корни уравнения*)
численное приближение
```

Out[ ]=
```
{8.00119, 9.00126, 10.001, 11.0007, 12.0003, 13., 13.9998, 14.9998, 15.9998, 16.9998}
```

In[ ]:=
```
numberOfIter (*Смотрим, какое число итераций*)
```

Out[ ]=
```
5
```

In[ ]:=
```
(*пункт б)*)
(*Метод Якоби*)
```

In[∘]:=
```
n = 20(*Задаем матрицу A и вектор решений B*)
f1[i_, j_] := Which[i ≠ j, 1, i == j, 2*n]
              условный оператор с множественными ветвя
f2[i_] := (2*n - 1)*i + n*(n+1)/2 + (3*n - 1)*7
A = Array[f1, {n, n}]
    массив
B = Array[f2, n]
    массив
```

Out[∘]=
```
20
```

Out[∘]=
```
{{40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40, 1},
 {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 40}}
```

Out[∘]=
```
{662, 701, 740, 779, 818, 857, 896, 935, 974, 1013,
 1052, 1091, 1130, 1169, 1208, 1247, 1286, 1325, 1364, 1403}
```

In[∘]:=
```
x = ConstantArray[1, n]
    постоянный массив
```

Out[∘]=
```
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
```

In[∘]:=
```
numberOfIter = 0(*Задаем начальное значение кол-ва итераций,
максимальное число итераций, приближение*)
maxIter = 100
eps = 0.001
```

Out[∘]=
```
0
```

Out[∘]=
```
100
```

Out[∘]=
```
0.001
```

In[ ]:=
```
While[numberOfIter < maxIter, (*Находим корни уравнеия по методу Якоби*)
цикл-пока
 xPrev = x;
 For[k = 1, k ≤ n, k++,
 цикл ДЛЯ
  S = 0;
  For[j = 1, j ≤ n, j++,
  цикл ДЛЯ

    If[
    условный оператор
     j ≠ k, S = S + A[[k, j]] * x[[j]]];
  ];
  x[[k]] = B[[k]] / A[[k, k]] - S / A[[k, k]];
 ];
 If[Sqrt[Norm[x - xPrev] / Norm[x]] ≤ 0.001, Break[];];
 ⋯ ква⋯ норма             норма                прервать цикл
 numberOfIter++]
```

In[ ]:=
```
numberOfIter (*Смотрим, какое число итераций*)
```

Out[ ]=

```
7
```

In[ ]:=
```
N[x] (*Выводим корни уравнения*)
численное приближение
```

Out[ ]=

```
{8., 9., 10., 11., 12., 13., 14., 15., 16.,
 17., 18., 19., 20., 21., 22., 23., 24., 25., 26., 27.}
```

In[ ]:=
```
(*Метод Зейделя*)
```

In[ ]:=
```
x = B
g = 0
y = B
s = 0
numberOfIter = 0(*Задаем начальные параметры*)
```

Out[ ]=

```
{662, 701, 740, 779, 818, 857, 896, 935, 974, 1013,
 1052, 1091, 1130, 1169, 1208, 1247, 1286, 1325, 1364, 1403}
```

Out[ ]=

```
0
```

Out[ ]=

```
{662, 701, 740, 779, 818, 857, 896, 935, 974, 1013,
 1052, 1091, 1130, 1169, 1208, 1247, 1286, 1325, 1364, 1403}
```

Out[ ]=

```
0
```

Out[ ]=

```
0
```

In[◦]:=
```
While[numberOfIter < maxIter, (*Находм корни используя метод Зейделя*)
цикл–пока
 s = 0;
 xPrev = x;
 For[i = 1, i ≤ n, i++,
 цикл ДЛЯ
  y[[i]] = (B[[i]] / A[[i, i]]);
  For[j = 1, j ≤ n, j++,
  цикл ДЛЯ
       If[
       условный оператор
     j ≠ i, y[[i]] = y[[i]] - ((A[[i, j]] / A[[i, i]]) * x[[j]]);
     x[[i]] = y[[i]];
    ];

  ];
 ];
 If[Norm[x - xPrev] / Norm[x] < eps, Break[];];
 ⋯  норма               норма           прервать цикл
 numberOfIter++]
```

In[◦]:=
```
N[x] (*Выводим корни уравнения*)
численное приближение
```

Out[◦]=

{7.99924, 8.99931, 9.9994, 10.9995, 11.9996, 12.9997, 13.9998, 14.9999, 16., 17.0001, 18.0001, 19.0001, 20.0001, 21.0002, 22.0001, 23.0001, 24.0001, 25.0001, 26.0001, 27.0001}

In[◦]:=
```
numberOfIter (*Смотрим, какое число итераций*)
```

Out[◦]=

6

```
(*Вывод: исходя из полученных значений, можно сделать вывод,
что число итераций у метода Зейделя меньше чем у метода Якоби*)
```