3. Массивы и матрицы в Scilab. Решение задач линейной алгебры

Для работы с множеством данных удобно использовать массивы. Например, можно создать массив для хранения числовых или символьных данных. В этом случае, вместо создания переменной, для хранения каждого данного, достаточно создать один массив, где каждому элементу будет присвоен порядковый номер.

Таким образом, *массив* – множественный тип данных, состоящий из фиксированного числа элементов. Как и любой другой переменной, массиву должно быть присвоено *имя*.

Переменную, представляющую собой просто список данных, называют *одномерным* массивом или *вектором*. Для доступа к данным, хранящимся в определенном элементе массива, необходимо указать имя массива и порядковый номер этого элемента, называемый *индексом*.

Если возникает необходимость хранения данных в виде таблиц, в формате строк и столбцов, то необходимо использовать *двумерные* массивы (*матрицы*). Для доступа к данным, хранящимся в таком массиве, необходимо указать имя массива и *два индекса*, первый должен соответствовать номеру строки, а второй номеру столбца в которых хранится необходимый элемент.

Значение нижней границы индексации в `Scilab` равно единице. Индексы могут быть только целыми положительными числами.

3.1. Ввод и формирование массивов и матриц

Самый простой способ *задать одномерный массив* в `Scilab` имеет вид

`[name]=Xn:dX:Xk`[1]

где `name` – имя переменной, в которую будет записан сформированный массив, `Xn` – значение первого элемента массива, `Xk` – значение последнего элемента массива, `dX` – шаг, с помощью которого формируется каждый следующий элемент массива, то есть значение второго элемента составит `Xn+dX`, третьего `Xn+ dX+dX` и так далее до `Xk`.

Возможен и такой способ *определения одномерного массива*:

`[name]=Xn:Xk`

Если параметр `dX` в конструкции отсутствует, это означает, что по умолчанию он принимает значение равное единице, то есть каждый следующий элемент массива равен значению предыдущего плюс один.

Примеры определения одномерных массивов таким способом приведены в листинге 3.1.

Переменную заданную как массив можно использовать в арифметических выражениях и в качестве аргумента математических функций. Результатом работы таких операторов

[1] Здесь и далее квадратные скобки (если не сказано иначе) означают, что выражение, указанное в них может отсутствовать.

являются массивы (листинг 3.1.).

```
--> Xn=-3.5;dX=1.5;Xk=4.5;
--> X=Xn:dX:Xk
X =
-3.5000 -2.0000 -0.5000 1.0000 2.5000 4.0000
--> Y=sin(X/2)
Y =
-0.9840   -0.8415   -0.2474   0.4794   0.9490   0.9093
--> A=0:5
A =
0 1 2 3 4 5
--> 0:5
ans =
0  1    2    3    4    5
--> ans/2+pi
ans =
3.1416   3.6416   4.1416   4.6416   5.1416   5.6416
```

Листинг 3.1

Еще один способ задания векторов и матриц в Scilab это их *поэлементный ввода*.

Так для *определения вектор–строки* следует ввести имя массива, а затем после знака присваивания, в квадратных скобках через пробел или запятую перечислить элементы массива:

[name]=x_1 x_2 … x_n или [name]=x_1, x_2, …, x_n

Пример ввода вектора–строки приведен в листинге 3.2.

```
--> V=[1 2 3 4 5]
V =
1 2 3 4 5
--> W=[1.1,2.3,-0.1,5.88]
W =
1.1000 2.3000 -0.1000 5.8800
```

Листинг 3.2

Элементы *вектора–столбца* вводятся через точку с запятой: [name]=x_1; x_2; …; x_n

Листинг 3.3 содержит пример ввода вектора–столбца.

```
--> X=[1;2;3]
X =
1
2
3
```

Листинг 3.3

Обратиться к элементу вектора можно, указав имя массива и порядковый номер элемента в круглых скобках: name(индекс). Листинг 3.4 содержит пример обращения к элементам массива.

```
--> W=[1.1,2.3,-0.1,5.88];
--> W(1)+2*W(3)
ans =
    0.9000
```

Листинг 3.4

Ввод элементов *матрицы* так же осуществляется в квадратных скобках, при этом элементы строки отделяются друг от друга пробелом или запятой, а строки разделяются между собой точкой с запятой:

[name]=[x_{11}, x_{12}, …, x_{1n}; x_{21}, x_{22}, …, x_{2n};…; x_{m1}, x_{m2}, …, x_{mn};

Обратиться к *элементу матрицы* можно, указав после имени матрицы, в круглых скобках, через запятую, номер строки и номер столбца на пересечении которых элемент расположен: `name(индекс1, индекс2).`

Примеры задания матриц и обращение к их элементам в листинге 3.5.

```
--> A=[1 2 3;4 5 6;7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
--> A(1,2)^A(2,2)/A(3,3)
ans =
    3.5556
```

Листинг 3.5

Кроме того, матрицы и векторы можно *формировать*, составляя их из ранее заданных матриц и векторов (листинг 3.6).

```
--> v1=[1 2 3]; v2=[4 5 6]; v3=[7 8 9];
--> //Горизонтальная конкатенация векторов-строк.
--> V=[v1 v2 v3]
V =
1  2    3    4    5    6    7    8    9
--> //Вертикальная конкатенация векторов-строк, результат матрица
--> V=[v1; v2; v3]
V =
1  2    3
4  5    6
7  8    9
--> //Горизонтальная конкатенация матриц
--> M=[V V V]
M =
1  2    3    1    2    3    1    2    3
4  5    6    4    5    6    4    5    6
7  8    9    7    8    9    7    8    9
--> //Вертикальная конкатенация матриц
--> M=[V;V]
M =
1  2    3
4  5    6
7  8    9
1  2    3
4  5    6
7  8    9
```

Листинг 3.6

Важную роль при работе с матрицами играет знак двоеточия «:». Указывая его вместо индекса при обращении к массиву, можно иметь доступ к группам его элементов (листинг

3.7).

```
--> A=[5 7 6 5; 7 10 8 7;6 8 10 9;5 7 9 10]
A = //Пусть задана матрица A
5  7    6    5
7  10   8    7
6  8    10   9
5  7    9    10
--> //Выделить из матрицы A второй столбец
--> A(:,2)
ans =
7
10
8
7
--> //Выделить из матрицы A третью строку
--> A(3,:)
ans =
6  8    10   9
--> //Выделить из матрицы A подматрицу M
--> M=A(3:4,2:3)
M =
8  10
7  9
--> //Удалить из матрицы A второй столбец
--> A(:,2)=[]
A =
5  8    10
7  7    9
6  10   9
5  9    10
--> //Удалить из матрицы A третью строку
--> A(3,:)=[]
A =
5  8    10
7  7    9
5  9    10
--> //Представить матрицу M в виде вектора-столбца
--> v=M(:)
v =
8
7
10
9
--> //Выделить из вектора v элементы со второго по четвертый
--> b=v(2:4)
b =
7
10
9
--> //Удалить из массива b второй элемент
--> b(2)=[];
```

Листинг 3.7

3.2. Действия над матрицами

Для работы с матрицами и векторами в Scilab предусмотрены следующие операции:

- + – сложение;
- – – вычитание[2];
- ’ – транспонирование;
- * – матричное умножение[3];
- * – умножение на число;
- ^ – возведение в степень[4];
- \ – левое деление, $(\boldsymbol{A}\backslash\boldsymbol{B}) \Rightarrow (\boldsymbol{A}^{-1}\cdot\boldsymbol{B})$, операция может быть применима для решения уравнения матричного уравнения вида $\boldsymbol{A}\cdot\boldsymbol{X}=\boldsymbol{B}$;
- / – правое деление, $(\boldsymbol{B}/\boldsymbol{A}) \Rightarrow (\boldsymbol{B}\cdot\boldsymbol{A}^{-1})$, используют для *решения матричных уравнений* вида $\boldsymbol{X}\cdot\boldsymbol{A}=\boldsymbol{B}$;
- .* – поэлементное умножение матриц;
- **.^** – поэлементное возведение в степень;
- **.** – поэлементное левое деление;
- **./** – поэлементное правое деление;

Листинг 3.8. содержит пример действий над матрицами.

```
-->A=[1 2 0;-1 3 1;4 -2 5];
-->B=[-1 0 1;2 1 1;3 -1 -1];
-->//Вычислить (A^T+B)^2 - 2A(0.5B^T-A)
-->(A'+B)^2-2*A*(1/2*B'-A)
 ans  =
    10.    8.     24.
    11.    20.    35.
    63.  - 30.    68.
```

Листинг 3.8

Кроме того, если к некоторому заданному вектору или матрице применить *математическую функцию*, то результатом будет новый вектор или матрица той же размерности, но элементы будут преобразованы в соответствии с заданной функцией (листинг 3.9).

---

[2] Эти операции определены для матриц одной размерности или векторов одного типа, то есть суммировать (вычитать) можно либо векторы–столбцы, либо векторы–строки одинаковой длины.

[3] Операция умножения вектора на вектор определена только для векторов одинакового размера, причем один из них должен быть вектором–столбцом, а второй вектором–строкой. Матричное умножение допустимо, если количество строк в одной матрице совпадает с количеством столбцов в другой.

[4] Возвести матрицу в n-ю степень, значит умножить ее саму на себя n раз. При этом целочисленный показатель степени может быть как положительным, так и отрицательным. В первом случае выполняется алгоритм умножения матрицы на себя указанное число раз, во втором умножается на себя матрица *обратная к данной.*

```
--> x=[0.1 -2.2 3.14 0 -1];
--> sin(x)
ans =
0.0998  -0.8085   0.0016    0    -0.8415
```
Листинг 3.9

3.3. Специальные матричные функции

Для работы с матрицами и векторами в Scilab существуют специальные функции. Рассмотрим наиболее часто используемые из них:

- length(X) – определяет количество элементов матрицы X, если X – вектор, его длину (листинг 3.10);

```
--> V=[-1 0 3 -2 1 -1 1];//Вектор-строка
--> length(V)//Длина вектора
ans =
    7
```
Листинг 3.10

- prod(X) – вычисляет произведение элементов матрицы или вектора X (листинг 3.11);

```
--> V=[1,2,3];
--> prod(V)%Произведение элементов вектора
ans =
    6
```
Листинг 3.11

- sum(X) – вычисляет сумму элементов матрицы или вектора X, кроме того с помощью этой функции можно вычислить скалярное произведение векторов (листинг 3.12);

```
--> V=[-1 0 3 -2 1 -1 1];
--> sum(V) //Сумма элементов вектора
ans =
1
-->//Частный случай. Вычисление скалярного произведения
--> a=[1 2 3];b=[2 0 1];
--> sum(a.*b)
 ans  =
    5.
```
Листинг 3.12

- min(X) – находит минимальный элемент матрицы или вектора X (листинг 3.13);
- max(X) – находит максимальный элемент матрицы или вектора X (листинг 3.13);

```
--> V=[-1 0 3 -2 1 -1 1];
--> min(V) //Минимальный элемент
ans =
-2
--> max(V) //Максимальный элемент
ans =
3
```
Листинг 3.13

- `mean(X)` – определяет среднее арифметическое матрицы или вектора `X` (листинг 3.14);

```
--> V=[-1 0 3 -2 1 -1 1];
--> mean(V)%Среднее значение массива V
ans =
    0.1429
--> sum(V)/length(V)%То же что и mean(V)
ans =
    0.1429
```

Листинг 3.14

- `sort(X)` – выполняет упорядочивание массива `X`, если `X` – матрица, сортировка выполняется по столбцам (листинг 3.15);

```
-->b
 b  =
    2.    0.    1.
-->sort(b) //Сортировка по убыванию
 ans  =
    2.    1.    0.
-->-sort(-b) //Сортировка по возрастанию
 ans  =
    0.    1.    2.
-->A
 A  =
    1.    2.    0.
  - 1.    3.    1.
    4.  - 2.    5.
-->sort(A) //Сортировка матрицы
 ans  =
    5.    2.    0.
    4.    1.  - 1.
    3.    1.  - 2.
```

Листинг 3.15

- `eye(n, m)` – возвращает единичную матрицу соответствующей размерности (листинг 3.16);

```
-->eye(3,3)
 ans  =
    1.    0.    0.
    0.    1.    0.
    0.    0.    1.-->eye(5,1)
-->eye(5,1)
 ans  =
    1.
    0.
    0.
    0.
    0.
```

Листинг 3.16

- `ones(n, m)` – формирует матрицу, состоящую из единиц (листинг 3.17);

```
-->m=3; n=2;
```

```
-->X=ones(m,n)
 X  =
    1.    1.
    1.    1.
    1.    1.
```

Листинг 3.17

- `zeros(n, m)` – возвращает нулевую матрицу соответствующей размерности (листинг 3.18);

```
-->zeros(3,2)
 ans  =
    0.    0.
    0.    0.
    0.    0.
```

Листинг 3.18

- `diag(V [, k])` – возвращает квадратную матрицу с элементами `V` на главной диагонали или на `k`–й; функция `diag(A [, k])`, где `A` ранее определенная матрица, в качестве результата выдаст вектор столбец, содержащий элементы главной или `k`–ой диагонали матрицы `A` (листинг 3.19);

```
--> V=[1,2,3];
--> diag(V)//Диагональная матрица, V на главной диагонали
ans =
1   0    0
0   2    0
0   0    3
-->//Диагональная матрица, V на первой диагонали (выше главной)
--> diag(V,1)
ans =
0   1    0    0
0   0    2    0
0   0    0    3
0   0    0    0
-->//Диагональная матрица, V на первой диагонали (ниже главной)
--> diag(V,-1)
ans =
0   0    0    0
1   0    0    0
0   2    0    0
0   0    3    0
--> A=[-1 2 0 ;2 1 -1 ;2 1 3]
A =
-1  2     0
 2  1    -1
 2  1     3
--> diag(A) //Главная диагональ матрицы A
ans =
    -1
     1
     3
```

Листинг 3.19

- `rand([n, m, p, …])` – возвращает матрицу случайных чисел, `rand` без аргументов возвращает одно случайно число (листинг 3.20);

```
--> rand(3,3)//Квадратная матрица случайных чисел
ans =
0.9501  0.4860    0.4565
0.2311  0.8913    0.0185
0.6068  0.7621    0.8214
--> R=rand(2,2,2)//Многомерный массив случайных чисел
R(:,:,1) =
0.9355  0.4103
0.9169  0.8936
R(:,:,2) =
0.0579  0.8132
0.3529  0.0099
--> rand //Одно случайное число
ans =
0.4057
```

Листинг 3.20

- `cat(n, A, B, [C, …])` – объединяет матрицы `A` и `B` или все входящие матрицы `A`, `B`, `C`, … При `n`=1 по строкам, при `n`=2 по столбцам. То же что `[A; B]` или `[A, B]` (листинг 3.21);

```
--> A=[1 2;3 4];
--> B=[5 6 ;7 8];
--> cat(2,A,B)%Объединение матриц
ans =
1   2    5    6
3   4    7    8
--> cat(1,A,B) %Объединение матриц
ans =
1   2
3   4
5   6
7   8
```

Листинг 3.21

- `tril(A [, k])` – формирует из матрицы `A` нижнюю треугольную матрицу начиная с главной или с `k`–й диагонали (листинг 3.22);

```
--> A=[1 2 3;4 5 6;7 8 9]
A =
1   2    3
4   5    6
7   8    9
--> tril(A)//Нижняя треугольная матрица, начиная с главной
диагонали
ans =
1   0    0
4   5    0
7   8    9
--> tril(A,0)//Тоже что и tril(A)
ans =
```

```
1  0    0
4  5    0
7  8    9
--> tril(A,1)//Нижняя треугольная матрица,
--> //начиная с первой диагонали (выше главной)
ans =
1  2    0
4  5    6
7  8    9
--> tril(A,-2) )//Нижняя треугольная матрица,
--> //начиная со второй диагонали (ниже главной)
ans =
0  0    0
0  0    0
7  0    0
```

Листинг 3.22

- `triu(A [, k])` – формирует из матрицы `A` верхнюю треугольную матрицу начиная с главной или с `k`–й диагонали (листинг 3.23);

```
--> A=[1 2 3;4 5 6;7 8 9];
--> triu(A)//Верхняя треугольная матрица
ans =
1  2    3
0  5    6
0  0    9
--> triu(A,2) )//Верхняя треугольная матрица,
--> //начиная со второй диагонали (выше главной)
ans =
0  0    3
0  0    0
0  0    0
--> triu(A,-1) )//Верхняя треугольная матрица,
--> //начиная с первой диагонали (ниже главной)
ans =
1  2    3
4  5    6
0  8    9
```

Листинг 3.23

- `size(A)` – определяет число строк и столбцов матрицы `A`, результатом ее работы является вектор `[n, m]` (листинг 3.24);

```
--> A=[2 4;1 3;7 9;5 8];
--> size(A)//Размерность матрицы A
ans =
4  2
--> V=[2 4 6 8 1 3 5 7];
--> size(V)//Размерность вектора-строки V
ans =
1  8
--> size(V')//Размерность вектора-столбца V'
ans =
8  1
-->//Формирование нулевой матрицы, такого же размера как и A
```

```
--> zeros(size(A))
ans =
0  0
0  0
0  0
0  0
-->//Формирование единичного вектора, такого же размера как и V
--> ones(size(V))
ans =
1  1    1    1    1    1    1    1
```

Листинг 3.24

- `det(A)` – вычисляет определитель квадратной матрицы `A` (листинг 3.25);

```
--> A=[3 2;4 3];
--> det(A)//Определитель матрицы
ans =
1
```

Листинг 3.25

- `trace(A)` – вычисляет след матрицы `A`, то есть сумму элементов главной диагонали (листинг 3.26);

```
--> A=[1 2 3;4 -2 1;7 0 -1]
A =
1   2    3
4  -2    1
7   0   -1
--> trace(A)//След матрицы A
ans =
-2
-->//Сумма элементов главной диагонали, то же что и trace(A)
--> sum(diag(A))
ans =
-2
```

Листинг 3.26

- `inv(A)` – возвращает матрицу обратную к `A` (листинг 3.27);

```
--> A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];
--> P=inv(A)//Матрица обратная к A
P =
 1.3333      -0.6667       0.3333       -1.0000
-0.0741       0.2593       1.1481       -0.1111
 0.3704      -0.2963       0.2593       -0.4444
 0.2593      -0.4074      -0.5185       -0.1111
--> A*P //Проверка A*P=E
ans =
1.0000  -0.0000       -0.0000        0.0000
0        1.0000        0.0000        0.0000
0.0000  -0.0000        1.0000       -0.0000
0.0000  -0.0000       -0.0000        1.0000
```

Листинг 3.27

- `linsolve(A, b)` – возвращает решение системы линейных уравнений ***Ax=b*** (листинг 3.28);

```
--> A=[1 2 3;-2 -4 -6]; b=[5;6];
--> //Система не имеет решений
-->linsolve(A,b)
WARNING:Conflicting linear constraints!
 ans  =
     []
-->//-----------------------------------------------
--> A=[2 -1 1;3 2 -5;1 3 -2]; b=[0;1;4];
--> x=linsolve(A,b)//Решение линейной системы
x =
0.4643
1.6786
0.7500
--> A*x //Решение верно
ans =
0.0000
1.0000
4.0000
```

Листинг 3.28

- `rref(A)` – осуществляет приведение матрицы `A` к треугольной форме, используя метод исключения Гаусса (листинг 3.29);

```
--> A=[3 -2 1 5;6 -4 2 7;9 -6 3 12]
A =
3  -2   1    5
6  -4   2    7
9  -6   3    12
--> rref(A)
ans =
1.0000       -0.6667       0.3333        0
0            0             0             1.0000
0            0             0             0
```

Листинг 3.29

3.4. Решение некоторых задач алгебры матриц

Матричное уравнение это уравнение вида $A \cdot X = B$ или $X \cdot A = B$, где X это неизвестная матрица. Если умножить матричное уравнение на матрицу обратную к A, то оно примет вид: $A^{-1} A X = A^{-1} B$ или $X A A^{-1} = B A^{-1}$. Так как $A^{-1} A = A A^{-1} = E$, а $E \cdot X = X \cdot E = X$, то неизвестную матрицу X можно вычислить так: $X = A^{-1} B$ или $X = B A^{-1}$. Понятно, что матричное уравнение имеет единственное решение если A и B – квадратные матрицы n-го порядка и определитель матрицы A не равен нулю.

ЗАДАЧА 3.1. Решить матричные уравнения $A \cdot X = B$ и $X \cdot A = B$ выполнить проверку.

Как решить матричное уравнение в `Scilab`, показано в листинге 3.30.

```
-->A=[3 2;4 3]
 A  =
    3.    2.
    4.    3.
-->B=[-1 7;3 5]
 B  =
```

```
  - 1.    7.
    3.    5.
-->// Решение матричного уравнения AX=B
-->// Первый способ
-->X=A\B
 X  =
  - 9.     11.
    13.  - 13.
-->// Второй способ
-->X=inv(A)*B
 X  =
  - 9.     11.
    13.  - 13.
-->// Решение матричного уравнения XA=B
-->// Первый способ
-->X=B/A
 X  =
  - 31.    23.
  - 11.    9.
-->// Первый способ
-->X=B*inv(A)
 X  =
  - 31.    23.
  - 11.    9.
-->// Второй способ
-->X*A-B
 ans  =
    0.    0.
    0.    0.
```

Листинг 3.30

3.5. Решение систем линейных алгебраических уравнений

Система m уравнений с n неизвестными вида

$$a_{11}x_1+a_{12}x_2+...+a_{1n}x_n=b_1,$$
$$a_{21}x_1+a_{22}x_2+...+a_{2n}x_n=b_2,$$
$$.....$$
$$a_{m1}x_1+a_{m2}x_2+...+a_{mn}x_n=b_m,$$

называется *системой линейных уравнений*, причем x_j – неизвестные, a_{ij} – коэффициенты при неизвестных, b_i – свободные коэффициенты ($i=1..m$, $j=1..n$). Система из m линейных уравнений с n неизвестными может быть описана при помощи матриц: $\boldsymbol{A}\cdot\boldsymbol{x}=\boldsymbol{b}$, где $\boldsymbol{x}=\{x_j\}$ – вектор неизвестных, $\boldsymbol{A}=\{a_{ij}\}$ – матрица коэффициентов при неизвестных или матрица системы, $\boldsymbol{b}=\{b_i\}$ – вектор свободных членов системы или вектор правых частей ($i=1..m$, $j=1..n$). Совокупность всех решений системы (x_1, x_2,..., x_n), называется *множеством решений* или просто *решением системы*.

ЗАДАЧА 3.2. Решить СЛАУ при помощи правила Крамера:

$$2x_1+ x_2-5x_3+ x_4=8,$$
$$x_1-3x_2 \quad -6x_4=9,$$
$$2x_2- x_3+2x_4=-5,$$
$$x_1+4x_2-7x_3+6x_4=0.$$

Правило Крамера заключается в следующем. Если определитель $\Delta=detA$ матрицы системы из n уравнений с n неизвестными $\boldsymbol{A\cdot x=b}$ отличен от нуля, то система имеет единственное решение $x_1, x_2,..., x_n$, Определяемое по формулам Крамера $x_i=\Delta_i/\Delta$, где Δ_i – определитель матрицы, полученной из матрицы системы $\boldsymbol{A}$ заменой i–го столбца столбцом свободных членов $\boldsymbol{b}$. Текст файла–сценария с решением задачи по формулам Крамера приведен в листинге 3.31.

```
A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];//Матрица коэффициентов
b=[8;9;-5;0]; //Вектор свободных коэффициентов
//Первая вспомогательная матрица
A1=A;A1(:,1)=b;
//Вторая вспомогательная матрица
A2=A;A2(:,2)=b;
//Третья вспомогательная матрица
A3=A;A3(:,3)=b;
//Четвертая вспомогательная матрица
A4=A;A4(:,4)=b;
//Главный определитель отличен от нуля
D=det(A);
//Определители вспомогательных матриц
d(1)=det(A1);
d(2)=det(A2);
d(3)=det(A3);
d(4)=det(A4);
//Вектор неизвестных
x=d/D
//Проверка
P=A*x-b
```

Листинг 3.31

Результаты работы S–файла показаны в листинге 3.31.

```
-->exec('C:\Program Files\scilab-4.0-rc1\bin\kramer.sce');
-->disp('exec done');
 x  =
    3.
  - 4.
  - 1.
    1.
 P  =
   1.0D-14 *
    0.1776357
    0.
  - 0.0888178
    0.1554312
 exec done
```

Листинг 3.32

ЗАДАЧА 3.3. Решить СЛАУ из задачи 1 методом обратной матрицы.

Метод обратной матрицы: для системы из *n* линейных уравнений с *n* неизвестными $\boldsymbol{A \cdot x = b}$, при условии, что определитель матрицы $\boldsymbol{A}$ не равен нулю, единственное решение можно представить в виде $\boldsymbol{x = A^{-1} \cdot b}$[5].

Текст файла–сценария в листинге 3.33, результаты его работы в листинге 3.34.

```
A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];
b=[8;9;-5;0];
//Решение системы: x=A-1*b
x=inv(A)*b
//проверка: A*x=b
A*x
```

Листинг 3.33

Результаты работы S–файла

```
--> x  =
    3.
  - 4.
  - 1.
    1.
 ans  =
    8.
    9.
  - 5.
    4.219D-15
```

Листинг 3.34

ЗАДАЧА 4. Решить систему линейных уравнений методом Гаусса:

$$2x_1 - x_2 + 5x_3 = 0,$$
$$3x_1 + 2x_2 - 5x_3 = 1,$$
$$x_1 + x_2 - 2x_3 = 4.$$

Решение системы линейных уравнений при помощи *метода Гаусса* основывается на том, что от заданной системы, переходят к эквивалентной системе, которая решается проще, чем исходная система.

Метод Гаусса состоит из двух этапов. Первый этап это *прямой ход*, в результате которого расширенная матрица[6] системы путем элементарных преобразований (перестановка уравнений системы, умножение уравнений на число отличное от нуля и сложение уравнений) приводится к ступенчатому виду. На втором этапе (*обратный ход*) ступенчатую матрицу преобразовывают так, чтобы в первых *n* столбцах получилась единичная матрица. Последний, *n*+1 столбец этой матрицы содержит решение системы линейных уравнений. Листинг 3.35 представляет текст файла–сценария, а листинг 3.36 результат работы S–файла.

---

[5] $\boldsymbol{A \cdot x = B \Rightarrow A^{-1} A\, x = A^{-1} B}$, т.к. $\boldsymbol{A^{-1} A = E}$, а $\boldsymbol{E \cdot x = x}$, то $\boldsymbol{x = A^{-1} B}$.

[6] Матрица $(\boldsymbol{A|b})$, которая формируется путем приписывания к матрице коэффициентов $\boldsymbol{A}$ столбца свободных членов $\boldsymbol{b}$, называется *расширенной матрицей* системы.

```
//Решение системы методом Гаусса
A=[2 -1 1;3 2 -5;1 3 -2];
b=[0;1;4];
//Приведение расширенной матрицы к треугольному виду
C=rref([A b]);
//Выделение последнего столбца из матрицы,
//x - решение системы
x=C(1:3,4:4)
A*x //Проверка
```

Листинг 3.35

```
--> x  =
    0.4642857
    1.6785714
    0.75
 ans  =
  - 1.110D-15
    1.
    4.
```

Листинг 3.36