

# Основы MATLAB

## Лекция 1

Юдинцев В. В.

Кафедра теоретической механики  
Самарский университет  
<http://yudintsev.info>

30 сентября 2016 г.

# Содержание

- 1 Введение
- 2 Основы работы в MATLAB
- 3 Операторы
- 4 Функции MATLAB
- 5 Функции пользователя
- 6 Ячейки
- 7 Структуры
- 8 Задачи

# **Введение**

# История создания

- MATLAB как язык программирования был разработан Кливом Моулером (Cleve Moler) в конце 1970-х годов для упрощения использования программных библиотек LINPACK и EISPACK без необходимости изучения Фортрана.
- В начале 80-х Джек Литл (Jack Little) Модернизировал эту систему для персональных компьютеров типа IBM PC, VAX и Macintosh.
- В 1984 основана компания The MathWorks inc.
- Последняя версия (09.2016): MATLAB R2016b.

# Структура MATLAB

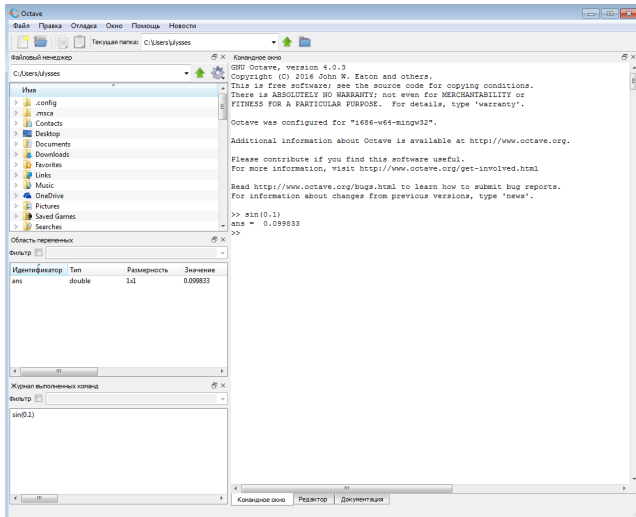
- Высокоуровневый **интерпретируемый** язык программирования, включающий основанные на матрицах структуры данных, широкий спектр функций, интегрированную среду разработки, объектно-ориентированные возможности и интерфейсы к программам, написанным на других языках программирования.
- **Toolboxes** – коллекции MATLAB-функций, для решения определённого класса задач (Optimization Toolbox, Partial Differential Equation Toolbox, Spline Toolbox, Statistic Toolbox).
- **Simulink** – приложение для анализа динамических систем.

# Свободное ПО

Близкое по функциональности свободное ПО:

- **GNU Octave** <https://www.gnu.org/software/octave/>;
- **FreeMat** <http://freemat.sourceforge.net/>;
- **Scilab** <http://www.scilab.org/>;
- **R** (для статистических расчётов) <https://www.r-project.org/>.
- **Python** с библиотеками numpy, scipy, matplotlib <http://www.scipy.org/>;
- **Sagemath** <http://www.sagemath.org/>.

# GNU Octave



# **Основы работы в MATLAB**



# Окно программы

- `Command window` – окно команд;
- `Command history` – окно истории истории команд;
- `Current directory` – окно, содержащие список файлов и папок текущего каталога;
- `Editor` – текстовый редактор.
- `Workspace` – окно со списком переменных текущей сессии.

Name	Value	Min	Max
C	100	100	100
X	<600x784 double>	0	255
X1	<300x784 double>	0	255
Y	<600x1 double>	-1	1
Y1	<300x1 double>	-1	1
accuracy	0.9600	0.96...	0.96...
ans	0	0	0
digit23Trn	<600x784 double>	0	255
digit23Tst	<300x784 double>	0	255
digit23TstT	<300x1 double>	-1	1
err	12	12	12
err_rate	0.0400	0.04...	0.04...
kernel	0	0	0
options	<1x1 struct>		
predictions	<300x1 double>	-1	1
sigma	0.5000	0.50...	0.50...
x	300	300	300

## Command History

```

--load predictions
predictions
predictions = sign(predictions)
-- 4/17/08 9:42 PM --
SA
Loss
-- 4/18/08 11:06 AM --
svmclassify
-- 4/19/08 3:33 PM --
-- 4/19/08 3:38 PM --
X= load('digit23Trn.mat')
X = X.digit23Trn
X = X.digit23Trn
Y = load('digit23TrnT')
X = load('digit23Trn.mat');
X = X.digit23Trn;
Y = load('digit23TrnT');
svmlwrite('Train',X,Y);
X1 = load('digit23Tst.mat');
X1 = X1.digit23Tst;
Y1 = load('digit23TstT');
svmlwrite('Test',X1,Y1);

```

## Command Window

1 New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

Runtime (without I/O) in cpu-seconds: 0.00

Accuracy on test set: 96.00% (288 correct, 12 incorrect, 300 total)

Precision/recall on test set: 96.62%/95.33%

```
err_rate =
```

```
0.0400
```

??? Error: File: PROJmain.m Line: 38 Column: 24

The expression to the left of the equals sign is not a valid target for an assignment.

Writing 100 200 300 400 500 600 done.

Writing 100 200 300 done.

Calling SVMlight:

```
svm_learn -c 100 -t 0 Train model
```

Scanning examples...done

Reading examples into memory...100..200..300..400..500..600..0K. (600 examples read)

Optimizing.....

Optimization finished (0 misclassified, maxdiff=0.00099).

Runtime in cpu-seconds: 0.21

Number of SV: 88 (including 0 at upper bound)

L1 loss: loss=0.00000

Norm of weight vector: |w|=0.00907

Norm of longest example vector: |x|=3499.11360

Estimated VCDim of classifier: VCDim<=1008.17259

Computing XiAlpha-estimates...done

Runtime for XiAlpha-estimates in cpu-seconds: 0.00

XiAlpha-estimate of the error: error<=13.67% (rho=1.00,depth=0)

XiAlpha-estimate of the recall: recall>=85.00% (rho=1.00,depth=0)

XiAlpha-estimate of the precision: precision>=87.33% (rho=1.00,depth=0)

Number of kernel evaluations: 29199

Writing model file...done

Calling SVMlight:

```
svm_classify Test model predictions
```

Reading model...0K. (88 support vectors read)

Classifying test examples..100..200..300..done

Runtime (without I/O) in cpu-seconds: 0.00

Accuracy on test set: 96.00% (288 correct, 12 incorrect, 300 total)

Precision/recall on test set: 96.62%/95.33%

```
accuracy =
```

```
0.9600
```

```
>>
```

# MATLAB как калькулятор

- $a=1.2$  – присвоение некоторого значения переменной  $a$

```
1 >> a=1.2  
2 a = 1.2
```



- $b=\sin(a)*\sqrt{a+2}$  – вычисление выражения и вывод результата

```
1 >> b=sin(a)*sqrt(a+2)  
2 b =  
3     1.6673
```

- ; – точка с запятой в конце выражения подавляет вывод результата:

```
1 >> b=sin(a)*sqrt(a+2);  
2 >>
```

## Базовые команды редактора (Command window)

- ↑ – возврат к предыдущей команде.
- `help имя функции` – справка по функции (или F1).
- `cls` – очистить окно команд (Command window).
- `clear` – удалить все переменные в текущей сессии.
-   - (в окне команд) прерывание вычислений.

# Синтаксис

- Имена переменных могут состоять из латинских букв, цифр, знака подчёркивания: `a`, `a1`, `x1`, `x_1`. Имена переменных чувствительны к регистру: `A` и `a` – это разные переменные.
- При создании переменной может быть явно указан тип (по умолчанию `double`):

```
1 >> a=int16(25);
```

# Типы данных

Тип	Описание
<code>double</code>	вещественный, 64 бит
<code>single</code>	вещественный, 32 бит
<code>int8</code>	знаковый целочисленный, 8 бит
<code>int16</code>	знаковый целочисленный, 16 бит
<code>int32</code>	знаковый целочисленный, 32 бит
<code>int64</code>	знаковый целочисленный, 64 бит
<code>uint8</code>	беззнаковый целочисленный, 8 бит
<code>uint16</code>	беззнаковый целочисленный, 16 бит
<code>uint32</code>	беззнаковый целочисленный, 32 бит
<code>uint64</code>	беззнаковый целочисленный, 64 бит

# Синтаксис

- Квадратные скобки используются для создания векторов и матриц: `a=[1,2,3,4]`.
- Круглые скобки используются для вызова функций и для группировки выражений: `sin(1.2)+a*(c+1)`.
- Фигурные скобки для создания массивов ячеек:  
`a={'Масса',10}`

# Создание последовательностей

- `var=начальное значение:шаг:конечное значение` по умолчанию шаг равен +1

```
1 >> c = 1:2:10
2 c =
3     1     3     5     7     9
```

## Создание вектора-столбца

```
1 >> c=(1:2:10) '
2 c =
3     1
4     3
5     5
6     7
7     9
```



# Управление переменными

- **whos** показать переменные текущей сессии

```
1 >> whos
2      Name      Size      Bytes  Class  Attributes
3
4      a         1x1         8    double
5      ans        1x5        40    double
6      b         1x1         8    double
```

- **clear** удалить все переменные текущей сессии
- **clear f1,f2** удалить переменные f1, f2 текущей сессии

## Ведение “протокола” работы

- `diary filename` – ведет запись на диск всех команд в строках ввода и полученных результатов в виде текстового файла.
- `diary off` – приостанавливает запись в файл.
- `diary on` – вновь начинает запись в файл.

# Сохранение сессии

Сохранение значения всех переменных в файл `*.mat`

- `save filename` – запись в файл `filename.mat` текущей сессии (значение всех переменных).
- `load filename` – загрузка значений переменных из файла `filename`.

# Ввод чисел

`a=1.25`

`a=9.3e10`

`a=2.36e-5`

$9.3 \cdot 10^{10}$

$2.36 \cdot 10^{-5}$

```
1 >> a=1.25
2 a = 1.2500
```

```
1 >> a=9.3e10
2 a = 9.3000e+10
```

```
1 >> a=2.36e-5
2 a = 2.3600e-05
```

# Формат отображения результата

```
1 >> format short;  
2 >> pi  
3 ans=  
4 3.1416
```

```
1 >> format long;  
2 >> pi  
3 ans=  
4 3.141592653589793
```

```
1 >> format short e;  
2 >> pi  
3 ans=  
4 3.1416e+00
```

```
1 >> format long e;  
2 >> pi  
3 ans=  
4 3.141592653589793e+00
```

# Константы

- `pi` – число  $\pi$

```
1 >> pi
2 ans = 3.1416
```

- `realmin` – минимальное положительное число

```
1 >> realmin
2 ans = 2.2251e-308
```

- `realmax` – максимальное положительное число

```
1 >> realmax
2 ans = 1.7977e+308
```

Использование имен переменных, совпадающих со встроенными именами, не рекомендуется

# Комплексные числа

- `complex(1,3)` – комплексное число  $1 + 3i$

```
1 >> a = complex(1,3);
```

- `i` или `j` – мнимая единица  $\sqrt{-1}$

```
1 >> b = 1+2i;  
2 >> a*b  
3 ans =  
4 -5.0000 + 5.0000i
```

$$(1 + 3i) \cdot (1 + 2i) = 1 + 2i + 3i - 6 = -5 + 5i$$

## ans – результат вычисления предыдущего действия

Встроенная переменная **ans** хранит результат последнего действия:

```
1 >> 2+2  
2 ans =  
3     4
```

Эту переменную можно использовать в выражениях:

```
1 >> ans*5  
2 ans =  
3    20
```



## Типы данных: векторы

$[1.2 \ 1.5 \ 1.9 \ 0.5]$  или  $[1.2, 1.5, 1.9, 0.5]$  вектор-строка

$$\begin{bmatrix} 1.2 & 1.5 & 1.9 & 0.5 \end{bmatrix}$$

$[1.2; 1.5; 1.9; 0.5]$  вектор-столбец

$$\begin{bmatrix} 1.2 \\ 1.5 \\ 1.9 \\ 0.5 \end{bmatrix}$$

# Матрицы и скаляры

Каждая переменная в MATLAB – это матрица, поэтому переменная

```
1 >> a=1;
```

это матрица размерности 1x1:

```
1 >> a(1,1)
2 ans =
3     1
```

## Типы данных: матрицы

- $A = [1.2 \ 1.5 \ 1.9; \ 0.5 \ 0.6 \ 0.7; \ 0.1 \ 1 \ 3]$

$$A = \begin{bmatrix} 1.2 & 1.5 & 1.9 \\ 0.5 & 0.6 & 0.7 \\ 0.1 & 1 & 3 \end{bmatrix}$$

- элементы вводятся построчно;
- элементы матрицы в строке можно разделять пробелами или запятыми;
- для разграничения строк используются точка с запятой;
- элемент матрицы может быть числом или выражением.

# Операторы

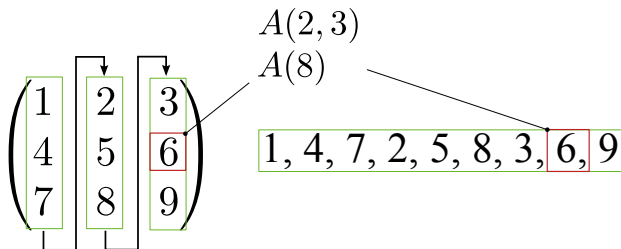
# Оператор ()

`help ops` вывести список всех операторов

`(i,j,k,...)` – доступ к элементам матрицы. `A(i,j)` –  $i$  строка,  $j$  столбец.

# Оператор ()

$A(i)$  –  $i$  элемент вектора (строки или столбца) или  $i$  элемент матрицы, при расположении элементов по столбцам.



## Логическое индексирование

Логическое индексирование позволяет выбрать из вектора или матрицы элементы, удовлетворяющие заданному условию.

```
1 >> a = sin(1:0.5:5)
2 a =
3     0.8415     0.9975     0.9093     0.5985     0.1411    -0.3508
      -0.7568    -0.9775    -0.9589
```

```
1 >> ind=a<0
2 ind =
3     0     0     0     0     0     1     1     1     1
```

```
1 >> a(ind)
2 ans =
3    -0.3508    -0.7568    -0.9775    -0.9589
```

# Скалярное произведение векторов

$$\vec{a} \cdot \vec{b} = |\vec{a}| |\vec{b}| \cos \varphi = a_x b_x + a_y b_y + a_z b_z$$

Вычисление скалярного произведения, используя встроенную функцию **dot**:

```
1 >> a=[1 2 3];  
2 >> b=[4 5 6];  
3 >> dot(a,b)  
4 ans =  
5     32
```



# Скалярное произведение векторов

$$\vec{a} \cdot \vec{b} = |\vec{a}| |\vec{b}| \cos \varphi = a_x b_x + a_y b_y + a_z b_z$$

Вычисление скалярного произведения, используя оператор матричного умножения:

```
1 >> a*b'  
2 ans =  
3     32
```

# Матричное умножение

**A\*B** умножение чисел или матриц (матричное умножение)

```
1 >> a = [1 2 3
2         4 5 6];
3 >> b = [1 2
4         3 4
5         5 6];
6 >> a*b
7 ans =
8      22      28
9      49      64
```

# Поэлементное умножение

$C = A .* B$  поэлементное умножение матриц  $C_{ij} = A_{ij}B_{ij}$

```
1 >> a = [1 2;  
2       3 4];  
3 >> b = [4 7;  
4       1 3];  
5 >> a .* b  
6 ans =  
7     4    14  
8     3    12
```

# Возведение в степень

## Возведение в степень матрицы или числа

```
1 >> a=[1 2  
2      3 4];  
3 >> a^2  
4 ans =  
5      7      10  
6      15      22
```

$$a^2 = a \cdot a = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 7 & 10 \\ 15 & 22 \end{pmatrix}$$

# Возведение в степень

## Возведение в степень всех элементов матрицы

```
1 >> a=[1 2
2       3 4];
3 >> a.^2
4 ans =
5     1     4
6     9    16
```

# Операторы

- $A'$  – транспонирование матрицы:

```
1 >> A=[1.2 0.3;  
2      0.5 2.1];  
3 >> A'  
4 ans =  
5      1.2000      0.5000  
6      0.3000      2.1000
```

- $C=A/B$  – деление  $C = AB^{-1}$
- $C=A./B$  – поэлементное деление  $C_i = A_i/B_i$

## Решение системы линейных уравнений

Деление  $A \setminus B$  выполняет операцию  $C = A^{-1}B$  – решение СЛУ с матрицей коэффициентов  $A$  и матрицей правой части  $B$ :

$$\begin{cases} 1.2x_1 + 0.3x_2 = 1; \\ 0.5x_1 + 2.1x_2 = 2. \end{cases}$$

```
1 >> A=[1.2 0.3;0.5 2.1];  
2 >> B=[1;2];  
3 >> A\B  
4 ans =  
5     0.6329  
6     0.8017
```

## Логические операторы < > = >= <= ~=

Результатом операции над матрицей или вектором является логический вектор (матрица):

```
1  >> A=[1,2,3,4,5,6];
2  >> A<5
3  ans=
4      [1,1,1,1,0,0]
5  >> A==3
6  ans=
7      [0,0,1,0,0,0]
8  >> A~=5
9  ans=
10     [1,1,1,1,0,1]
11
```



# Оператор :

- Для создания списков с равноотстоящими значения используется оператор :  
 $n1:s:n2$   $n1, n1 + s, n1 + 2s, \dots, n_k, n_k \leq n2$
- Оператор : может использоваться для доступа к элементам матрицы и вектора
- $A=1:10$  – 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- $A(1:2:10)$  – нечетные элементы вектора: [1, 3, 5, 7, 9]

# Оператор :

Для прямоугольной матрицы  $A$ :

- $A(:, k)$  –  $k$ -ый столбец матрицы  $A$
- $A(k, :)$  –  $k$ -ая строка матрицы  $A$
- $A(:, [2, 4, 5])$  – второй, четвертый и пятый столбец матрицы  $A$

```

1  >> a = magic(4)
2  a =
3      16      2      3     13
4      5     11     10      8
5      9      7      6     12
6      4     14     15      1
7  >> a([1 3], :)
8  ans =
9      16      2      3     13
10     9      7      6     12

```

## Блоки матриц

$A(1:2, 3:4)$  – блок матрицы  $A$ , лежащий на пересечении строк 1 и 2, и столбцов 3 и 4.

```
1 >> a = magic(4)
2 a =
3     16     2     3    13
4     5    11    10     8
5     9     7     6    12
6     4    14    15     1
7 >> a([1 2],[3 4])
8 ans =
9     3    13
10    10     8
```

## Удаление строк и столбцов из матрицы

$A(3,:) = []$  – удалить третью строку из матрицы  $A$

```
1 >> a = magic(4)
2 a =
3     16     2     3     13
4     5     11    10     8
5     9     7     6    12
6     4    14    15     1
7 >> a(3,:) = []
8 a =
9     16     2     3     13
10     5     11    10     8
11     4    14    15     1
```

# Функции MATLAB

# Математические функции

`sin, cos, tan, cot, acos, asin, ...` – тригонометрические  
`log, log10, log2, exp, sqrt, nthroot(x,n), ...`  
`sign(a)` – знак числа `a`: -1, 0, +1.

## Функции округления

- К ближайшему к нулю целому:  
`fix(1.8)=1` но `fix(-1.8)=-1`
- К меньшему целому:  
`floor(-1.9)=-2` но `floor(1.9)=1`
- К большему целому:  
`ceil(1.3)=2`
- К ближайшему целому:  
`round(1.4)=1`, `round(-1.5)=-2`

## Функции комплексных чисел

- `abs(z)` – модуль.
- `angle(z)` – аргумент.
- `conj(z)` – комплексно-сопряженное число.
- `imag(z)` – мнимая часть.
- `real(z)` – вещественная часть.
- `isreal(z)` – 1, если  $z$  – вещественное число, 0 – мнимое.



# Специальные матрицы

- `eye(n)`, `eye(n,m)` – единичные матрицы  $n \times n$ .

```
1 >> eye(3,4)
2 ans =
3     1     0     0     0
4     0     1     0     0
5     0     0     1     0
```

- `rand(n)`, `rand(n,m)` – матрица псевдослучайных чисел
- `ones(n,m)` – матрица  $n \times m$ , заполненная единицами
- `zeros(n,m)` – матрица  $n \times m$ , заполненная нулями

# Специальные матрицы

`magic(n)` – квадратная матрица  $n \times n$ , с одинаковыми суммами элементов по строкам и столбцам и диагоналям.

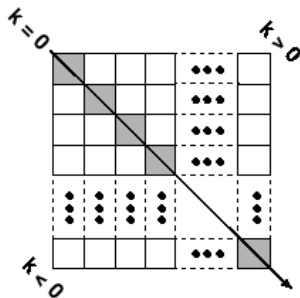
```
1 >> magic(3)
2 ans =
3     8     1     6
4     3     5     7
5     4     9     2
```

## Функции над матрицами

- `diag(A)`, `diag(A,k)` – строка, содержащая диагональные элементы матрицы  $A$ , если  $A$  – матрица ( $k=0$  для главной диагонали).
- `diag(V)` – диагональная матрица с элементами вектора  $V$  на диагонали, если  $V$  – вектор.
- `tril(A)`, `tril(A,k)` – нижняя треугольная матрица, построенная по матрице  $A$ .
- `triu(A)`, `triu(A,k)` – верхняя треугольная матрица, построенная по матрице  $A$ .

# Функции над матрицами

Параметр  $k$  в функциях `diag(A,k)`, `tril(A,k)`, `triu(A,k)`



## Функции над векторами

- `cross(a,b)` – векторное произведение;
- `dot(a,b)` – скалярное произведение;
- `sum(a)` – сумма элементов вектора;
- `sort(a)` – сортировка вектора; если `a` – матрица, то производится сортировка элементов в столбцах.  
Другой вариант вызова `[res,i]=sort(a)`. Переменная `i` содержит индексы элементов исходного вектора `a`, расположенные по порядку сортировки.

# Сумма элементов вектора или матрицы

`sum(a)`

Сумма элементов вектора:

```
1 >> a=[1 2 3 4];  
2 >> sum(a)  
3 ans=  
4 10  
5
```

Сумма элементов матрицы:

```
1 >> a=[1 2;  
2      3 4];  
3 >> sum(a)  
4 ans=  
5 4 6  
6
```

# Сумма элементов матрицы

Сумма элементов матрицы по заданному измерению:

```
1 >> a=[1 2;  
2       3 4];  
3 >> sum(a,2)  
4 ans =  
5     3  
6     7  
7
```

# Сортировка

`sort(a)` – сортировка вектора или матрицы. Если `a` – матрица, то производится сортировка элементов в столбцах (по первому измерению):

```
1  >> a = magic(3)
2  a =
3      8      1      6
4      3      5      7
5      4      9      2
6  >> sort(a)
7  ans =
8      3      1      2
9      4      5      6
10     8      9      7
11
```



# Сортировка

Вторым аргументом функции `sort` может быть номер измерения, по которому должна выполняться сортировка:

```
1  >> a = magic(3)
2  a =
3      8      1      6
4      3      5      7
5      4      9      2
6  >> sort(a,2)
7  ans =
8      1      6      8
9      3      5      7
10     2      4      9
11
```

# Сортировка

`[res,i]=sort(a)`. `i` содержит индексы элементов исходного вектора `a`, расположенные по порядку сортировки.

```
1 >> a = magic(3)
2 a =
3     8     1     6
4     3     5     7
5     4     9     2
6 >> [res,i] = sort(a)
7 res =
8     3     1     2
9     4     5     6
10    8     9     7
11 i =
12     2     1     3
13     3     2     1
14     1     3     2
```

## Битовые функции

**bitand(a,b)** – поразрядное И

```
1 >> bitand(5,1)
2 ans =
3      1
```

**bitor(a,b)** – поразрядное ИЛИ

```
1 >> bitor(4,2)
2 ans =
3      6
```

**bitset(a,bit,v)** – установка бита в позиции bit числа a; v = 1 или 0

```
1 >> bitset(5,3,0)
2 ans =
3      1
```

# Преобразование числа к другому основанию

**dec2bin(a)** – строка с двоичным представлением положительного числа *a*

```
1 >> dec2bin(5)
2 ans =
3     101
4
```

**dec2hex(a)** – строка с шестнадцатеричным представлением положительного числа *a*

```
1 >> dec2hex(255)
2 ans =
3     FF
4
```

# Преобразование числа к другому основанию

**bin2dec(a)** – преобразование строки с двоичным представлением числа к десятичной системе *a*

```
1 >> bin2dec('111')  
2 ans =  
3     7  
4
```

**hex2dec(a)** – преобразование строки с шестнадцатеричным представлением числа к десятичной системе *a*

```
1 >> hex2dec('F1')  
2 ans =  
3    241  
4
```

## Работа с множествами

**intersect(A, B)** – пересечение векторов A и B как множеств:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> intersect(a,b)  
4 ans =  
5      1      7
```

**ismember(A, S)** – содержит ли A элементы из S:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> ismember(a,b)  
4 ans =  
5      1      0      0      1      0      1      0      1
```

## Работа с множествами

`setdiff(A, B)` – элементы, входящие в A, но отсутствующие в B (A-B):

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> setdiff(a,b)  
4 ans =  
5      2      3      8
```

`setxor(A, B)` – элементы, не входящие в результат пересечения множеств A и B:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> setxor(a,b)  
4 ans =  
5      2      3      5      6      8
```

# Работа с множествами

`union(A, B)` – объединение множеств:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> b = [7 1 5 6];  
3 >> union(a,b)  
4 ans =  
5      1      2      3      5      6      7      8
```

`unique(A)` – элементы вектора A без повторений:

```
1 >> a = [1 2 3 7 8 7 3 1];  
2 >> unique(a)  
3 ans =  
4      1      2      3      7      8
```



## **Функции пользователя**

# Способы задания функции

- inline функции;
- анонимные функции;
- файл-функции .

# inline-функция

```
1 >> f1 = inline('x1^2+x2^2','x1','x2');  
2 >> f1(4,2)  
3 ans =  
4     20
```

Переменные из рабочей области недоступны.

## Анонимная функция

```
1 >> f = @ (x1,x2) x1^2+x2^2;  
2 >> f(4,2)  
3 ans =  
4     20
```

Переменные из рабочей области доступны, но рассматриваются как константы:

```
1 >> a=2;  
2 >> f = @ (x1) a*x1;  
3 >> f(1)  
4 ans =  
5     2  
6 >> a=1; f(1)  
7 ans =  
8     2
```

# Ячейки

# Ячейки

Массивы ячеек (cells) могут хранить данные различных типов: числа, строки, структуры, массивы ячеек.

```
1 >> data{1} = 'Текст'
2 data =
3     'Текст'
4 >> data{2} = 1
5 data =
6     'Текст'     [1]
7 >> data{5} = [1 2 3]
8 data =
9     'Текст'     [1]     []     []     [1x3 double]
```

## Доступ к элементам массива ячеек

```
1 >> data{1}
2 ans =
3     Текст
4
5 >> data{2}
6 ans =
7     1
8
9 >> data{5}
10 ans =
11     1     2     3
```

# Функция cell

## Формирование пустой матрицы ячеек

```

1 >> a = cell(3,4)
2 a =
3     []     []     []     []
4     []     []     []     []
5     []     []     []     []

```

## Присвоение значения элементу новой матрицы

```

1 >> a{2,3} = 'элемент 2,3'
2 a =
3     []     []     []     []
4     []     []     'элемент 2,3'   []
5     []     []     []     []

```



# Функции для работы с ячейками

## Преобразование числовой матрицы в матрицу ячеек

```
1 >> a=magic(5)
```

```
2 a =
```

```
3     17     24      1      8     15
4     23      5      7     14     16
5      4      6     13     20     22
6     10     12     19     21      3
7     11     18     25      2      9
```

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
```

```
2 b =
```

```
3     [2x2 double]     [2x1 double]     [2x2 double]
4     [3x2 double]     [3x1 double]     [3x2 double]
```

## Функции для работы с ячейками

### Преобразование числовой матрицы в матрицу ячеек

```
1 >> a=magic(5)
```

```
2 a =
```

```
3      17      24       1       8      15
4      23       5       7      14      16
5       4       6      13      20      22
6      10      12      19      21       3
7      11      18      25       2       9
```

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
```

```
2 b =
```

```
3      [2x2 double]      [2x1 double]      [2x2 double]
4      [3x2 double]      [3x1 double]      [3x2 double]
```

# Преобразование матрицы ячеек в числовую матрицу

Функция `mat2cell`:

```
1 >> b = mat2cell(a,[2 3],[2 1 2])
2 b =
3     [2x2 double]     [2x1 double]     [2x2 double]
4     [3x2 double]     [3x1 double]     [3x2 double]
5 >> cell2mat(b)
6 ans =
7     17     24     1     8     15
8     23     5     7    14    16
9     4      6    13    20    22
10    10    12    19    21     3
11    11    18    25     2     9
```

# Структуры

# Структура

Структура – это элемент данных, который может содержать разнотипные **поля**:

```
имя = struct('поле1', значение, 'поле2', значение, ...)
```

Структура, описывающая параллелепипед:

```
1 box=struct('height', 100, 'width', 10, 'depth', 5)
2 box=
3     height: 100
4     width: 10
5     depth: 5
```

# Доступ к полям структуры

- Считывание значения поля

```
1 >> box.height
2 ans =
3     100
```

- Имя поля может быть задано строкой

```
1 >> box.( 'height' )
2 ans =
3     100
```

# Список полей структуры

```
1 >> names = fieldnames(box)
2 names =
3     'height'
4     'width'
5     'depth'
6     'mass'
```

## Изменение значения поля

```
1 >> box.height = 20
2 box =
3     height: 20
4     width: 10
5     depth: 5
```



## Добавление поля

Для добавления поля к структуре необходимо присвоить значение новому полю, как существующему:

```
1 >> box.mass=5
2 box =
3     height: 20
4     width: 10
5     depth: 5
6     mass: 5
```

## Создание структуры

Создание новой структуры без использования функции **struct**:

```
1 >> sphere2.radius=2  
2 sphere2 =  
3     radius: 2
```

## Задачи

# Задание 1

Напишите кратчайшее однострочное выражение для построения квадратной матрицы вида

$$\begin{bmatrix} 1 & 2 & 3 & 4 & \dots & n \\ 1 & 2 & 3 & 4 & \dots & n \\ \dots & \dots & \dots & \dots & \dots & n \\ 1 & 2 & 3 & 4 & \dots & n \end{bmatrix}$$

## Задание 2

Напишите выражение, определяющее индекс элемента вектора с наименьшим отклонением от среднего значения элементов вектора  $\mathbf{P}$

$$i : \min |P_i - m(\mathbf{P})|$$

Например, для

$$P = [8, 1, 2, 2, 0, 3]$$

ответом будет 6, т.е. шестой элемент массива находится ближе всего к среднему значению элементов массива. Среднее значение определяется при помощи функции **mean**

```
1 >> P=[8 1 2 2 0 3];  
2 >> mean(P)  
3 ans = 2.6667  
4 >>
```

## Задание 3

Напишите код, который находит матрицу **B**, отличающуюся от матрицы **A** перестановкой столбцов по возрастанию суммы элементов столбца.

$$\mathbf{A} = \begin{bmatrix} 2 & 3 & 5 \\ 4 & 1 & 9 \\ 3 & 0 & 2 \end{bmatrix} \Rightarrow \mathbf{B} = \begin{bmatrix} 5 & 2 & 3 \\ 9 & 4 & 1 \\ 2 & 3 & 0 \end{bmatrix}$$

$$\boxed{5 + 9 + 2} > \boxed{2 + 4 + 3} > \boxed{3 + 1 + 0}$$

## Задание 4

В матрице **A** записаны координат точек на плоскости: в первом столбце – координаты  $x$ , во втором столбце –  $y$ :

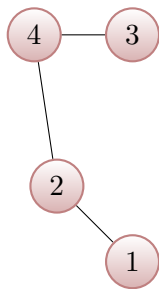
$$\mathbf{A} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ \dots & \dots \\ x_n & y_n \end{bmatrix}$$

Постройте матрицу **B**, которая составлена из строк матрицы **A** так, что с увеличением номера строки с координатами точки растёт расстояние от этой точки до начала координат.

## Задание 5

Структура графа описана при помощи списка смежности, записанного в виде матрицы:

$$\mathbf{L} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \\ \dots & \dots \\ 3 & 4 \end{bmatrix},$$



В каждой строке матрицы  $\mathbf{L}$  записываются индексы вершин, связанные дугой.

Постройте матрицу смежности графа  $\mathbf{S}$ , элемент которой –  $s_{ij}$  отличен от нуля, только если вершины  $i$  и  $j$  смежные, т. е. соединены дугой. Остальные элементы матрицы  $\mathbf{S}$  равны нулю.