

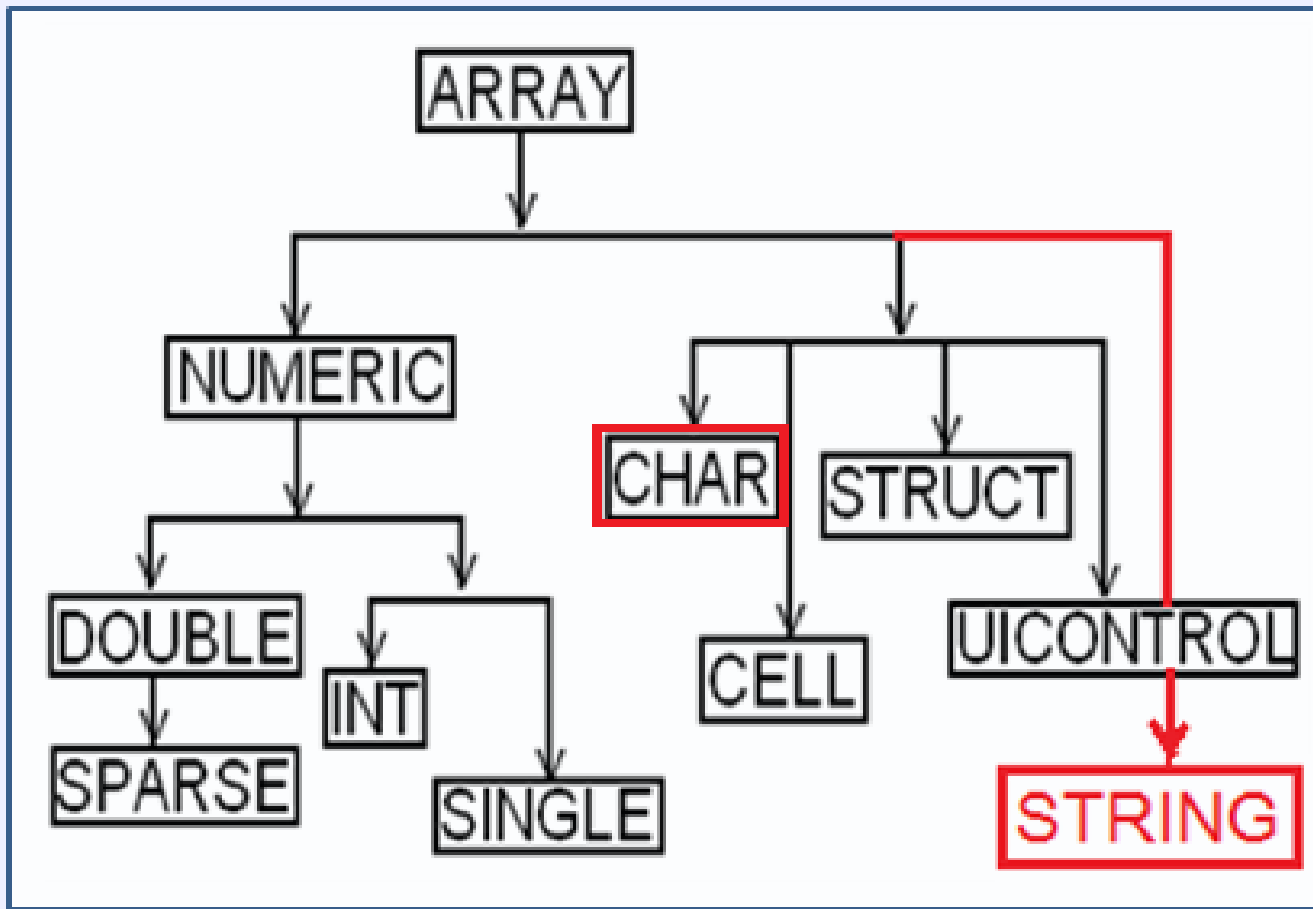


Scientific Computer Software (*MatLab*)

Topic 3. Class hierarchy in MatLab. Classes Double, Char, String, Cell



Class hierarchy





Contents:

Array – all native objects are Array!

- ndims, size, length, numel
- display

Numeric, Double

- Matrix constructors
- Functions and properties(eig, det, rank, cond...)
- SLAE. Gauss Method.
- Construction of the simplest graphs
- I/O operations

String, Char, Cell

- Constructors
- Basic class functions



Array

nvkurbatova@sfedu.ru

The main functions inherited by descendants

N=ndims(a) – dimensions number

[m, p] = size(a) – number of row and column

numel (a) – total elements number (= **m*p**)

length(a) – default: rows number

length(a,2) – column number (length along row)

display (a); – the semicolon does not suppress
the output for **display**

inherited [in 'heritid]

descendants [di 'sendants]



Numeric. Double

Matrix Constructors:

- **zeros** — the zero matrix
- **eye** — the unit matrix
- **ones** — all elements are equal to 1
- **rand, randi, randn** — random matrix
- **magic(n)** — a matrix with the properties of a magic square, the sums of any row and column are equal (does not exist for any order)
- **blkdiag** — constructor of a block-diagonal matrix, f.e.:

```
C={ones(3),randi(20,2,4),rand(3), triu(ones(2))}; % C∈Cell  
blkdiag(C{:}) % C∈Cell
```



Constructors of:

- **triu** – the upper triangular matrix

`triu(A); triu(A,-k); triu(A,k);`

- **tril** – the lower triangular matrix

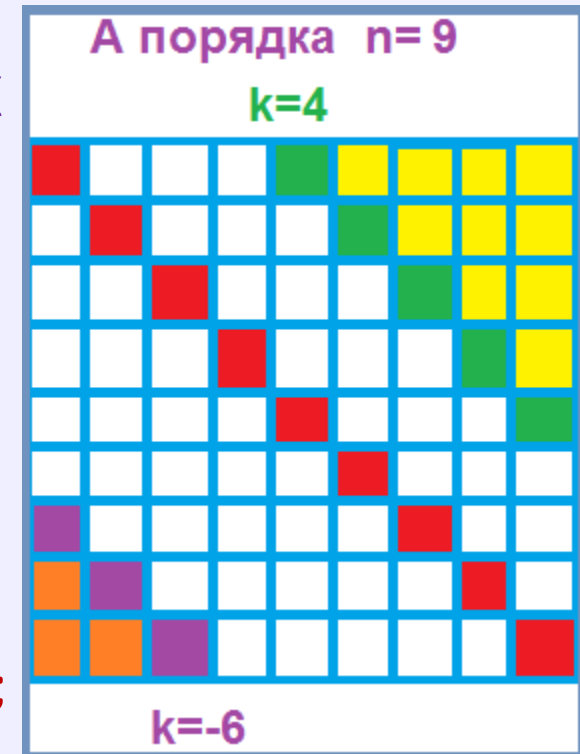
`tril(A); tril(A,-k); tril(A,k);`

- **diag** – the diagonal matrix

`a=diag(A); a=diag(A,k); a=diag(A,-k);`

`A=diag(a); A=diag(a,k); A=diag(a,-k);`

$k > 0$, integer, A – matrix, a – vector





Vector editing of matrices by condition

Syntactic form of searching for elements by condition:

➤ $\text{LogicalMatrix} = \mathbf{A} \text{ RelationOperation } \mathbf{B}$

RelationOperation: $>$, $<$, $=$, ... $\mathbf{A}$ - matrix, $\mathbf{B}$ – constant value or matrix ...

Syntactic [sin'tæktik] form of editing elements by condition:

➤ $\mathbf{A}(\mathbf{A} \text{ RelationOperation } \mathbf{B}) = \text{const}$ – assigning new values to matrix elements that meet the condition (*RelationOperation* $=$, $>$, $<$, $\sim$)

Example

$\mathbf{A} = [0, 0, -2; 0, -1, -1; -2, -2, 0]$

$\text{LogicalMatrix} = \mathbf{A} < -1$ LogicalMatrix – ?

$\mathbf{A}(\mathbf{A} < -1) = 3$ A - ?

A =		
0	0	-2
0	-1	-1
-2	-2	0



Analysis of matrix elements: function: **find**

- **index=find(A)** % finding indexes for elements $A \sim 0$
 $A(\text{index})$ or $A(\text{index}(:))$ **contains nonzero elements;**
- **[i,j]=find(AnyCondition)** % i,j indexes of elements, satisfying the condition, f.e., AnyCondition={ $A > \text{value}$ или $A < \text{value}$ или $A == \text{value}, \dots$ };

A(i,j) the same as Acond

Brain Storm!

index=find(A) % need to determine indexes!
[i,j]=find(A==-2) % - i,j ?

A =		
0	0	-2
0	-1	-1
-2	-2	0



Logical analysis of matrix elements: functions: **all, any**

Analysis of matrix relation operations :

- $\forall$ relation, f.e. **$C=A==B$** , the result is logical matrix C;
0 – equality is not true, 1 - true
- **TrueFalse=all(all(C)); TrueFalse=1**, all elements of C equal to **1 (true)**,
matrices **A,B,C=A==B** have the same dimension
- **TrueFalse=any(any(C)); TrueFalse=1**, if there is at least one element
in A that is equal to the corresponding element in B

Check for an arbitrary matrix a, is the matrix $a'*a$ – symmetric?



Examples

Поиск элементов с заданным условием:

*% Генерируем целочисленную матрицу
% размера 2x3, величина элементов
% принадлежит отрезку [-9,9]:*

```
a=randi([-9,9],2,3); n=nnz(a)
```

*% Находим nnz - ненулевые элементы
% и их индексы i,j, такие, что $a(i(k),j(k)) = anz(k)$, $k=1:n$*

```
[i,j,anz]=find(a);
```

```
[i,j,anz] % упорядочены, см. ans в CommandWindow  
for k=1:n,anz (k)=a(i(k),j(k)),end % testing
```

And what will be the result? Explain it!!

```
[k]=find(a)
```

```
a(k(:)) % см. Рис.
```

```
>> [i, j, anz]  
ans =  
1 1 -7  
1 2 9  
2 2 -3  
1 3 2  
2 3 -5  
  
>> a  
a =  
-7 9 2  
0 -3 -5
```

Рис.



Functions on the of the matrices

- ✓ $\sin(a)$, $\cos(a)$, $\tan(a)$ – a (in radians)
- ✓ $\text{sin}(b)$, $\text{cos}(b)$, $\text{tan}(b)$ – b (degrees)
- ✓ atan , asin , acos – as usual
- ✓ $\log(x) \sim \ln(x)$; $\log_{10}$ – clear!
- ✓ imag , real – (imaginary and parts)

```

12 - b=[pi/3 pi/2; 3*pi/2 pi]
13 - a=sin(b)
14 -

```

Command Window

b =

0.8660	1.0000
-1.0000	0.0000

a: 2x2 double =

1.0472	1.5708
4.7124	3.1416

sum, prod, abs, min, max, sort – the functions perform the operation on the columns, along the rows – default, f.e. **sum(a)~sum(a,1);**

```

15 - a=ones(3), b= repmat([1,2,3],3,1)
16 - a=a.*b, s=sum(a)
17 -

```

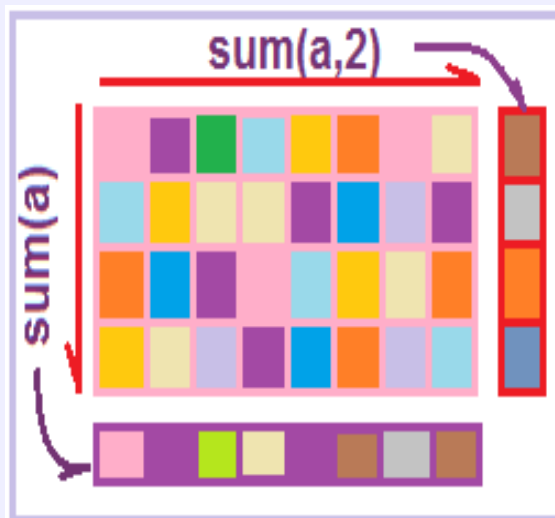
Command Window

s: 1x3 double =

3	6	9
---	---	---

a =

1	2	3
1	2	3
1	2	3



sum(a,2); line by line, along all columns

sum(sum(a)) ?



Functions defining the properties of matrices

- **$s = \det(a)$** – the value of the determinant is calculated
- **$[v, D] = \text{eig}(a)$** – **v** – a matrix of eigenvectors-columns, **D** – diagonal matrix, $D(i, i)$ – eigenvalues
- **$\text{cond}(a)$** the number of conditionality; characterizes the stability of the solution
- **issymmetric** (ishermitian for complex), **isfull** , **issparse** , **isequal**
- **$\text{rank}(a)$** – clear
- **$\text{inv}(a)$** or **a^{-1}** inverse matrix ☹
- **$\text{norm}(a, p)$** →

Виды норм:

p	Matrix	Vector
1	$\max(\text{sum}(\text{abs}(X)))$	$\text{sum}(\text{abs}(X))$
2	$\max(\text{svd}(X))$	$\text{sum}(\text{abs}(X).^2)^{(1/2)}$
real value p	-	$\text{sum}(\text{abs}(X).^p)^{(1/p)}$
Inf	$\max(\text{sum}(\text{abs}(X')))$	$\max(\text{abs}(X))$
-Inf	-	$\min(\text{abs}(X))$

Decomposition svd:

$$[U, S, V] = \text{svd}(X);$$

$$X = U * S * V'$$

U, V - unitary matrices,

$$(U * U') = I, (V' * V) = I,$$

' - transposition and complex conjugation

Where is the Euclidean norm? We are looking for! 12



SLAE. The Gauss method

$$\mathbf{Ax}=\mathbf{b}$$

$$\begin{cases} 3x + y + z = 5 \\ -x + y = 1 \\ x + 2y - z = 5 \end{cases} \quad \mathbf{A} = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 1 & 0 \\ 1 & 2 & -1 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 5 \\ 1 \\ 5 \end{bmatrix}$$

B ML:

```
>>A=[3,1,1; -1 1 0; 1, 2, -1]
```

```
>>b =[5 1 5]'
```

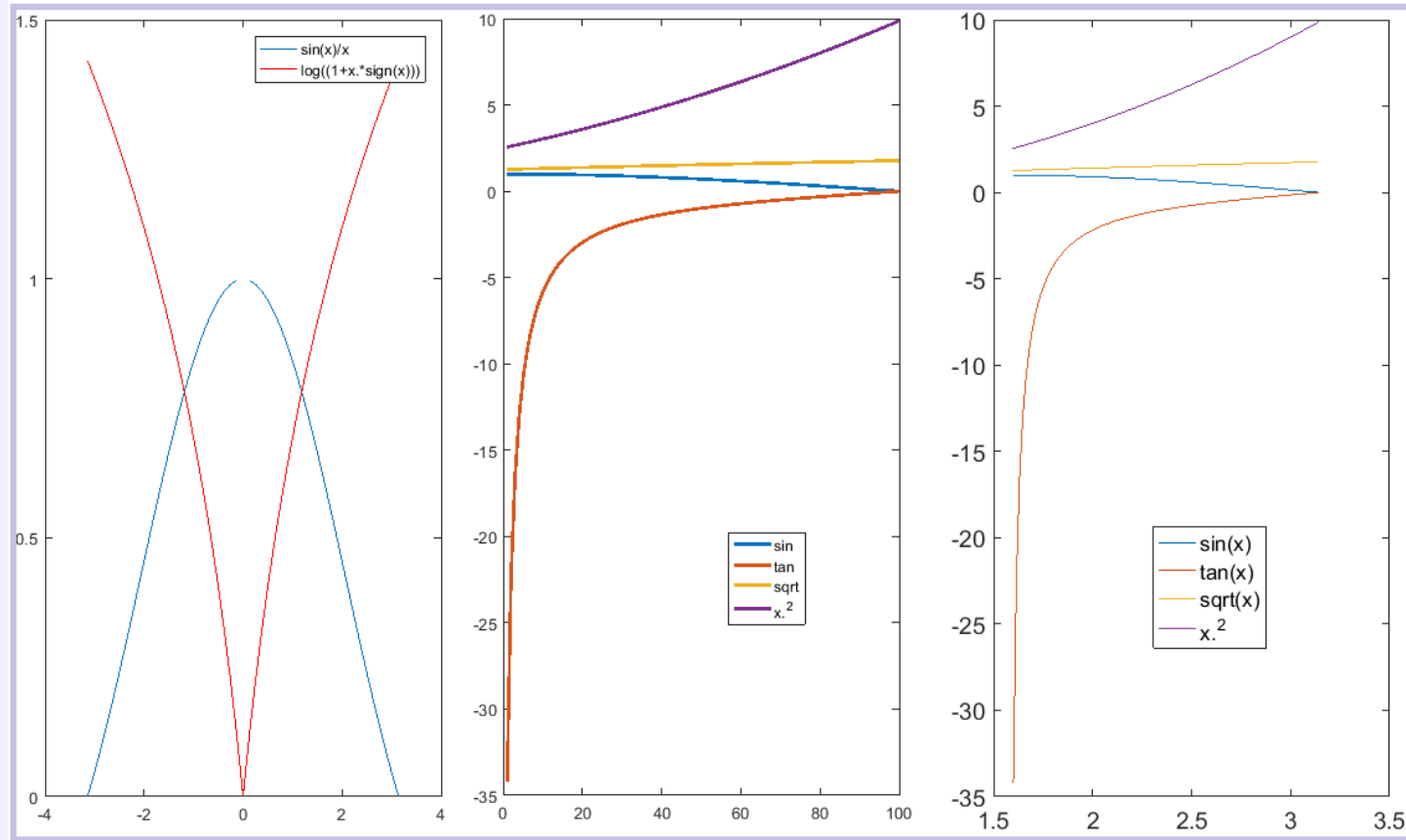
```
>>x=A\b % The Gauss method or inverse division
```



Graph.m

```
clear
x=-pi/24:pi;
y= sin(x)./x; % element by
element (vector operation)
z=log((1+x.*sign(x)));
subplot(1,3,1);
plot(x,y); hold on;
plot(x,z,'r-');
legend('sin(x)/x',...
'log((1+x.*sign(x)))');
subplot(1,3,2);
% clear x % (if it needs)
x=linspace(1.6,3.1415,100); %
z=[sin(x)', tan(x)', sqrt(x)' x.^2'];
r=plot(z);
legend({'sin','tan','sqrt','x.^2'});
subplot(1,3,3)
plot(x,z);
legend({'sin(x)','tan(x)',...
'sqrt(x)','x.^2'});
set(r(:),'linewidth',2);
set(gca,'fontsize',14);
```

Simple plots. Class Double



subplot(m,n,k) % m×n windows, k-an active;

hold on change NextPlot of Figure:

Before - **replace** (default) ; after - **add**;

What is the risk?

x=linspace(a,b,n) % vector n – n points on [a,b];

gca (graphic current axes) - an active axes



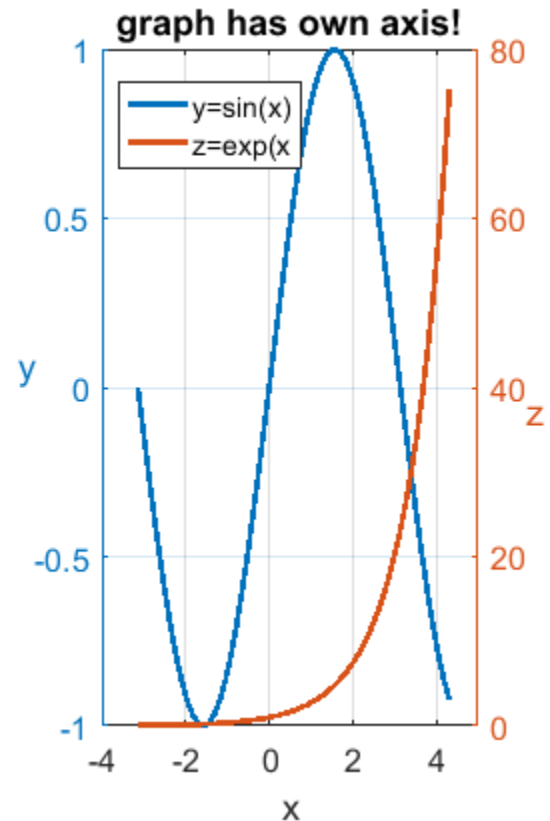
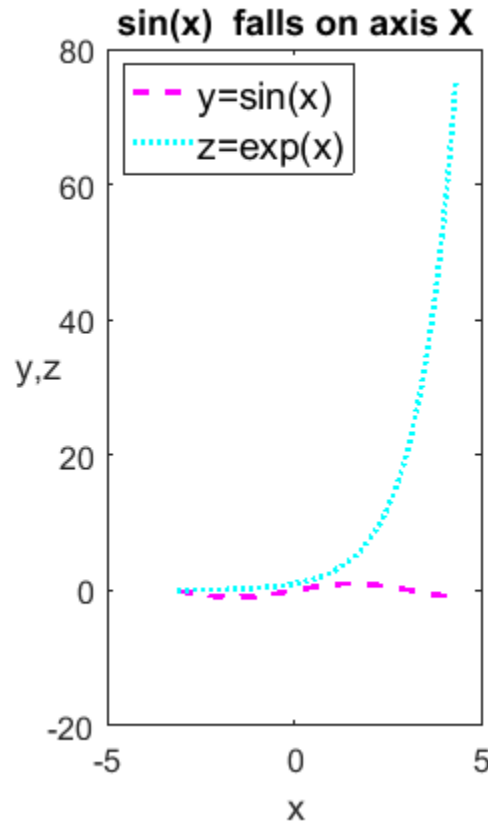
DifferentIncrease.m

Graphs of functions of different growth. plotyy -> yyaxis (for last versions)

PLOT

PLOTYY

```
x=-pi:pi/24:1.4*pi; y= sin(x); z=exp(x);
subplot(1,2,1);
title('sin(x) {\it falis} on axis X')
plot(x,y,'m--','linewidth',2); hold on
plot(x,z,'c:','linewidth',2);
lg1=legend('y=sin(x)','z=exp(x)');
lg1.FontSize=14;
xlabel('x'),
yl=ylabel('y,z','rotation',0),
yl.FontSize=14
set(gca,'fontsize',14)
subplot(1,2,2);
title('Each graph has its own vertical axis!')
[haxes,line1,line2]=plotyy(x,y,x,z);
line1.LineWidth=2;
line2.LineWidth=2;
xlabel('x'); grid on;
ylabel(haxes(1),'y','rotation',0);
ylabel(haxes(2),'z','rotation',0);
legend('y=sin(x)','z=exp(x)');
set(haxes(1),'fontsize',14);
set(haxes(2),'fontsize',14)
```



When using the **set** and **get** commands, the case is not specified in the names of the structural fields!
In the notation ObjectPropertyName = PropertyValue,
case is important!



String array

String Constructor – double quotes (" ")

```
1 clear
2 %% STRING Array
3 %% Creation
4 S="MEXMAT FOREVER!"
5 S1=isletter(S);

S: 1x1 string =
    "MEXMAT FOREVER!"

S1: 1x15 logical array =
    1     1     1     1     1     1     0     1     1     1     1     1     1     1     0
```

S - СКАЛЯР!

Line	Code	Command Window
1	clear	
2	%% STRING Array	
3	%% Creation	
4	S="MEXMAT FOREVER!"	S3 =
5	S1=isletter(S);	"MEXMAT .SFEDU"
6	S2=split(S) %split(S,delimiter,dim)	
7	length(S2)	s3 =
8	%% Edition	"mexmat .sfedu"
9	S3=S2(1)+" .SFEDU" % concatenation	
10	s3=lower(S3) % S3=upper(S3) %register	ans =
11	erase(S3,' ') % S3=erase(S3,' ')	
12	S3	"MEXMAT.SFEDU"

Control of letters in a string: ISLETTER

```
S1=isletter(S);

S1: 1x15 logical array =
    1     1     1     1     1     1     0     1     1     1     1     1     1     1     0
```

SPLIT - dividing a string into elements (words), the result is a column vector of string (words); space is the default separator (see S2)
UNIQUE(STRING) - the choice of non-repeating characteristics in string

ERASE – deleting a specified element in a row (см.S3)

LOWER, UPPER – преобразование к соответствующему регистру (см.S3)

What is the result of the line number 12?



Char - character array

Char elements Constructor – single quotes (' ')

Command Window				Workspace	
>> whos				Name	Value
	Name	Size	Bytes	Class	
	a	1x6	12	char	
	b	1x1	8	double	

What are the differences between STRING and Char?

Why does a take 12 bytes and res takes 24 bytes?

It is true for Char and String:

upper, lower – converting a string to uppercase or string (case selection)

fliplr – (flip) inversion from left to right (compare b, b1!)

A Уверен и не реву - palindrome['pælɪndrəʊm]?

length(CHAR) ; strlen(STRING) - Number of characteristics;

>>length(b) % = 9;

>>strlen('TRUE') % = 4

How does flipud work? (Brainstorming)

It is true for Char:

```
2 - a=upper('letter') % as for string
3 - l=length(a);
4 - it=a(4:end) % Total Expense Ratio
5 - res=imag(it)
6 - b='ТОТ ПОТОП'; % almost Palindrom
7 - b1=fliplr(b)
```

Command Window			
K>> b1			
b1 = ПОТОП ТОТ			
K>> whos b res			
	Name	Size	Bytes Class
	b	1x9	18 char
	res	1x3	24 double

An element of the Double class takes up 8 bytes!

eval('clc, format long, r=pi/180, phi=pi/180*30, sin(phi)')



Char , String. Editing

Conversion Cell to String and vice versa :

```

Help Center  mathworks.com/help/matlab/ref/string.html
Documentation  Examples  Functions  Apps  Videos  Answers

A = {'Mercury','Gemini','Apollo';...
     'Skylab','Skylab B','ISS'}

A = 2x3 cell
     {'Mercury'}  {'Gemini' }  {'Apollo'}
     {'Skylab' }  {'Skylab B'}  {'ISS'  }

str = string(A), str(1,2)

str = 2x3 string
     "Mercury"  "Gemini"  "Apollo"
     "Skylab"   "Skylab B" "ISS"

ans =
     "Gemini"
    
```

```

K>> A = ['Mercury','Gemini','Apollo';...
         'Skylab','Skylab B','ISS']
Dimensions of matrices being concatenated are not
consistent.
    
```

```

MercuryGeminiAppolo
SkyLab SkyLab B ISS
    
```

StrArr=string(CellChar) – StrArr has the same size as **CellChar** and consists of Elements of String

Str2mat (A) – Conversion Cell of Char to String (as vector column)

```

10  B=str2mat(A);
    str 2x3 string

B =
Mercury
Skylab
Gemini
Skylab B
Apollo
ISS
    
```

←-----What's wrong? How can we fix it?
Let's formulate a rule!



Array of Cell and Char

Combining objects of different types into a container

- **C=cell(size)** – an object of the given size is being reserved
- **C={A,B,D,...}** – A,B,D,... - $\forall$ MatLab objects

```
Command Window

C =
    2x1 cell array

    'my mind is strong'
    'my mind is resting'

TF =
    logical
     0
ans =
    logical
     1

Editor - F:\Nata\2017\Matlab 2017\2023\Лекции\Work Folder\MyPresentation\Le
KurbatovaStart.m  ForLecture2.m  StringChar.m  +
1 - clear
2 - %% comparisen
3 - C={'my mind is strong';...
4 -   'my mind is resting'}
5 - TF=strncmpi(C{1},C{2}) % C(1),C(2)
6 - strncmp(C{1},C{2},10)
```

```
Command Window

ind =
     4
S =
my mind is strongmy mind is resting
ans =
    1x12 logical array

x 1 1 1 1 1 1 1 1 1 1 1 1
```

```
[ind]=findstr(C{1},'mind')
S=[C{1},C{2}]; CS=string(S) % Размер CS?
[Ced,ind1]=unique(S);
S(ind1)==Ced %
[Ced,ind1]=unique(CS);
```

Comparison :

- **ScalarTrueOrFalse = strcmp(S1,S2)** – comparison of characteristics in general
- **[LogicalVector]=S1==S2;** or **eq(S1,S2);**
- **TF=strncmp(S1,S2,n);** % - for the first n elements
- **TF=strcmpi(S1,S2)** % full comparisen



Class CELL

cell2mat – cell to matrix
num2cell – numeric to cell

% Cell Creation:

%% I)

```
c=cell(1,7)
[c{1:end}]=3l('O, ', ' Небо,', ' Небо,', ...
    'ты', 'мне', 'будешь', 'сниться?')
```

% Compare

```
disp(c') ; celldisp(c) %
c{:} % each element - ans;
```

%% II) Assign:

```
c1={eye(3),'Be healthy',1:8',num2cell(2:6)}
disp(c1) % this is not good idea!
```

%% III) Celldisp, visualization of the cell contents

```
celldisp(c1)
c1{1}, c1{2}(3:end), c1{3}(5), c1{4}{3}
%% Conversion :
nc=num2cell([2 3 4;5 6 7])
nm=cell2mat(nc)
```

Examples:

```
mat =
we are
studing
MatLab

ans =
     3     7
ans = char
ans = 1
```

```
>> num2cell('mat')
ans = 'm' 'a' 't'
>> num2cell(mat)
ans = 'w' 'e' ' ' 'a' 'r' 'e' ' '
      's' 't' 'u' 'd' 'i' 'n' 'g'
      'M' 'a' 't' 'L' 'a' 'b' ' '
```

cell2mat taking into account the size of the submatrices

Ex. Пусть c – Cell, point III

$r=cell2mat(c)$, % $c = \{[1] [2 \ 3 \ 4]; [5; 9] [6 \ 7 \ 8; 10 \ 11 \ 12]\}$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}$$

String ↔ Char ↔ Double ↔ Cell

Charact=char(Str);

- String new class in MatLab!
- Double quotes (") is only in last versions!
- The CHAR class is much better (although it depends on your goal)

CS=strsplit(S,dlm) – conversion of each word **S**, between delimiters **dlm** in **S**, $S \in \text{Char}$ to $\text{CS} \in \text{Cell}$; **dlm** is space default

strsplit('Диван незаразен на вид - Palindrome!'); →

```
res=double('xyz') % ASCII code
% res=120 121 122
str2double('pi/2'); % calculate!
Charac=num2str(ClassDouble) % ∈ Char
legend(Charac)
str2double('5+0.17i') % 5.0000 + 0.1700i
str2num(['1 2';'3 4']) % ans =
    1    2
    3    4
num2str(pi,3) % ans =3.14
```

Str=string(Charact);

If there is not enough functionality in the **CHAR** class, then we convert it to the **String** class and vice versa!

```
ans =
    1×6 cell array
    Columns 1 through 6
    'Диван' 'незаразен' 'на' 'вид' '-' 'ПАЛИНДРОМ!'
```



I/O~ load/save

Import

IMPORT VIEW

Range: A1:C2

Variable Names Row: 1

Column vectors
Numeric Matrix
Cell Array
Table ✓

Define the type of

	A	B	C
	data		
	VarName1	VarName2	VarName3
	Number	Number	Number
1	1	2	3
2	4	5	6

Workspace

Name	Value
data	2x3 table

Binary file: Nfile.mat

load Nfile Vars; **load** Nfile

Text files (saved before)

load file.txt % inp.dat

Home → Import Data

Interactive download

load, save:

File Edit Text Go Cell

7 - load inp.dat

8 inp(:)

9

inp: 4x3 double =

1	2	3
4	5	6
7	8	9
10	11	12

inp.dat



xlsread , xlsxwrite (Excel)

inex.xls

	A	B
1	0,1869	0,7547
2	0,4898	0,276
3	0,6463	0,7655
4	0,7094	0,7952
5	0,7547	0,1869
6	0,6797	0,4456

- Here (example) without Path, file *.xls (*.xlsx) in the current folder
- SheetName – name much better in English
- [STATUS,MESSAGE] = xlsxwrite(file,array,sheet,range), file, array – required parameters

```
10 - exmatr=xlsread('inex.xls')
11 ● → d
12 | exmatr: 6x2 double =
    0.1869    0.7547
    0.4898    0.2760
    0.6463    0.7655
    0.7094    0.7952
    0.7547    0.1869
    0.6797    0.4456
```



Download Excel table. Example

[a,b,c]=xlsread('data1.xlsx')

a – real part of input data

b – Cell array, containing untypical data

c – Cell array, combining a и b.

data1.xlsx - Microsoft Excel				
	A	B	C	D
1	01.01.2023	2	3	1
2	04.01.2023	5	6	4
3	07.01.2023	7	12	0,3
4		1	2	2
5	13.01.2023	2	7	3

a =

2.0000	3.0000	1.0000
5.0000	6.0000	4.0000
7.0000	12.0000	0.3000
1.0000	2.0000	2.0000
2.0000	7.0000	3.0000

b =

5×1 cell array

'01.01.2023'
'04.01.2023'
'07.01.2023'
' '
'13.01.2023'

c =

5×4 cell array

'01.01.2023'	[2]	[3]	[1]
'04.01.2023'	[5]	[6]	[4]
'07.01.2023'	[7]	[12]	[0.3000]
[NaN]	[1]	[2]	[2]
'13.01.2023'	[2]	[7]	[3]



fscanf, fprintf (formatted text)

`Fid=fopen(name, opts)`

`A=fscanf(Fid, format, [m,n])` % [m,n] – size A

`fclose(fid);`

if `n=inf`, then read into m row until the columns are exhausted

```
- fid1=fopen('inpnum.txt','r')
- A=fscanf(fid1,'n=%d eps=%g x0=%f\n',[3,4])
```

```
A: 3x4 double =
    1.0000    2.0000    3.0000    4.0000
    0.2000    0.0500    0.0001    0.0000
    3.2564    3.1419    3.1417    3.1415
```

```
clear
- fid=fopen('inpinf.txt','r')
- A=fscanf(fid,'%d %g \n', [2,inf])
```

```
A: 2x4 double =
    1.0000    2.0000    3.0000    4.0000
    0.2000    0.0500    0.0001    0.0000
```

		- 1.0		+ ÷ 1.1		×	
1	n=1	eps=2.0e-01	x0=3.2564				
2	n=2	eps=5.0e-02	x0=3.1419				
3	n=3	eps=1.0e-04	x0=3.1417				
4	n=4	eps=1.0e-05	x0=3.1415				
inpnum.txt		×					

1	1	2.0e-01
2	2	5.0e-02
3	3	1.0e-04
4	4	1.0e-05
inpinf.txt ×		



Опции форматированного I/O

\n	Переход на новую строку
\t	Горизонтальная табуляция
\b	Возврат назад на один символ
пробелы в строке формата записываются как пробелы	

%c	Последовательность символов
%d	Десятичные числа
%e, %f, %g	Числа с плавающей запятой
%i	Целое число со знаком
%o	Восьмеричное целое число со знаком
%s	Ряд символов без пробелов
%u	Десятичное целое число со знаком
%x	Шестнадцатеричное целое число со знаком

Topics Help:

https://www.mathworks.com/help/matlab/matlab_prog/formatting-strings.html



Thanks for your attention!

“Как бы медленно ты ни шёл, не останавливайся!”

孔子 (*Конфуций*)

No matter how slow you walk, don't stop!