Массивы ячеек

Массив ячеек – это массив, элементами которого являются ячейки, содержащие массивы любого типа, в том числе и массивы ячеек. Массивы ячеек позволяют хранить массивы с элементами разных типов и разных размерностей. К примеру, одна из ячеек может содержать действительную матрицу, другая – массив текстовых строк, третья – вектор комплексных чисел (рис. 1.4.1). Можно создавать массивы ячеек любой размерности.

При работе с массивами ячеек можно использовать следующие функции.
- {}, cell – создание массив ячеек;
- cellstr – создание массива ячеек строк из символьного массива;
- cellfun – применение функции к каждому элементу в массиве ячеек;
- celldisp – показ содержимого массива ячеек;
- cellplot – показ графической структуры массива ячеек;
- deal – обмен данными;
- num2cel – преобразование числового массива в массив ячеек;
- cell2mat – преобразование массива ячеек в отдельную матрицу;
- mat2cell – разбиение матрицы на массив ячеек матриц;
- cell2struct – преобразование массива ячеек в структуру;
- struct2cell – преобразование структуры в массив ячеек;
- iscell – определяет, является ли введенная переменная массивом ячеек.

Создание массивов ячеек

Для создания массивов ячеек используются конструкторы {} и cell и некоторые функции для работы с ячейками.

Конструктор {} действует подобно оператору [] для числовых массивов.
Он объединяет данные в ячейки.

Пример 1 . Образование массива из четырех ячеек

```
C = {1 20 33 404}
B = cell(2,3)% пустой массив ячеек 2x3
```

Пример 2 . Можно построить массив ячеек, присваивая данные отдельным ячейкам

```
A(1, 1) = {[1 4 3; 0 5 8]}
A(1, 2) = {'Математика'}
A(2, 1) = {3+7i}
A(2, 2) = {-2:2:6}
```

Пример 3 . Образование массива ячеек с использованием индексации

```
A{1, 1} = {[1 4 3; 0 5 8]}
A{1, 2} = {'Математика'}
A{2, 1} = {3+7i}
A{2, 2} = {-2:2:6}
```

Замечание 2
Система MATLAB не очищает массив ячеек при выполнении оператора присваивания. Могут остаться старые данные в незаполненных ячейках. Полезно удалять массив перед выполнением оператора присваивания.

Замечание 3
Нельзя называть одним именем разные массивы, даже, если один из них числовой массив, а другой – массив ячеек.

- Для отображения содержимого ячеек следует использовать функцию **celldisp(A)**
- Для отображения структуры массива ячеек в виде графического изображения **cellplot(A)**

Пример 4 . Обращение к ячейкам

```
c = A{1, 2}
```

Пример 5. Извлечение элемента с индексами (2,2) из числового массива ячейки A{1,1}

```
d = A{1, 1}(2, 2)
```

Пример 6. Доступ к подмножеству ячеек

```
B=A(1,:)
Ответ:    [2x3 double]    'Математика'
```


Удаление ячеек из массива. Удалять можно либо целую строку, либо столбец.
Пример 7.

```
A(1, :)=[] % удаление 1-й строки
Массив ячеек стал одномерным
A(2)=[] % удаление 2-й ячейки
```

Пример 8. Можно создавать вложенные массивы ячеек

```
A(1, 1) = {magic(5)};
A(1, 2) = {{[5 2 8; 7 3 0; 6 7 3] 'Test 1'; [2-4i 5+7i] {17 []}}}
```

Пример 9. Многомерные массивы ячеек

```
A{1, 1} = 'Name';
A{1, 2} = [4 2; 1 5];
A{2, 1} = 2-4i;
A{2, 2} = 7;
```

```
B{1, 1} = 'Name2';
B{1, 2} = [ 3 5 ]';
B{2, 1} = 0:1:3;
B{2, 2} = 3;
C = cat(3, A, B);
```

Элементы программирования MatLab

Примеры

Оператор if

```
k=randi(3,1)
if k==1
     'one'
elseif k==2
    'two'
else 'three'
end;
```

Оператор swith

```
k=randi(3,1)
switch k
    case 1
        'one'
    case 2
    'two'
    otherwise
        'three'
end;
```


Цикл for

```
x=[10 20 30]
for i=1:3
    x(i)
end

x=[10 20 30]
for i=x
    x-i
end;
```

Цикл while

```
k=0;
while k<5
     k=k+1
end;
```

Оператор continue

```
k=0;
while k<4
    k=k+1;
    if (k==2)
        continue
    else
            prod(1:k)
     end
end;
```

Оператор break

```
clc
clear
k=0;
while k<6
    k=k+randi(2,1)
    if (k==4)
        break
    end
end;
```

Команды try catch

```
%%
clc
clear
n=randi(2,1)
A=randi(5,n)
b=randi(2,2,1)
try
   linsolve(A,b)
catch
    msgstr = lasterr
end
s='Hello!'
```


ФУНКЦИИ MatLab

Функции могут принимать более одного входного аргумента, и может возвращать более одного выходного аргумента. Имя функции совпадает с именем файла, в который она записана. Файл называется файл-функция.

```
function [ output_args ] = FunName( input_args ) % имя файла FunName.m
% Summary of this function goes here
%   Detailed explanation goes here


end
```

Произвольное количество аргументов функции varargin и varargout

В MATLAB имеются функции, которые могут иметь меняющееся число входных аргументов и меняющееся число выходных параметров. Например, функция S=svd(A) вычисления сингулярных чисел матрицы À. Она может применяться в виде [U,S,V]=svd(A), когда требуется большее число выходных параметров. Другим примером такой функции может служить функция cat(A,B) горизонтального объединения массивов A и B . Она может иметь произвольное число входных массивов, cat(A1,A2,A3,A4) . Для создания таких функций, использующих неопределенной количество аргументов, в список аргументов вставляют переменные **varargin** и **varargout.**

Пример 10. Тип файла – функция. Имя файла – varlist.m

```
function varlist(varargin)
  fprintf('Number of arguments: %d\n',nargin);
% nargin – количество входных аргументов в функции
  celldisp(varargin)
```

Вызов функции:

```
varlist(ones(2),'some text',pi)
```

Что выведет:

```
Number of arguments: 3
varargin{1} =
    1    1
    1    1
varargin{2} =  some text
varargin{3} = 3.1416
```

Пример 11. Тип файла – функция. Имя файла – sizeout.m

```
function [s,varargout] = sizeout(x)
nout = max(nargout,1) - 1;
% nargout – количество выходных аргументов функции
s = size(x);
for k=1:nout
  varargout{k} = s(k);
end
```

Вызов функции:

```
[s,rows,cols] = sizeout(rand(4,5,2))
```

Что выведет:

```
s =     4     5     2
rows =     4
cols =      5
```

Пример 12. Количество входных параметров.

Тип файла – функция. Имя файла – testarg1.m

```
function c = testarg1(a,b)
if (nargin == 1)
   c = a.^2;
elseif (nargin == 2)
   c = a + b;
end
```

Вызовы функции:

```
estarg1([1 2])
testarg1([1 2],[3 4])
```

Что выведет:

```
ans =    1    4
ans =    4    6
```

Анонимные функции в MatLab

Анонимная функция, как инлайн функция в традиционных языках программирования. Она состоит из одного выражения MATLAB и любого количества входных и выходных параметров.

Вы можете определить анонимную функцию прямо в командной строке MATLAB или в пределах функции или скрипта.

Таким образом, вы можете создать простые функции без необходимости создания файла для них.

Синтаксис для создания анонимной функции из выражения является

```
f = @(arglist)expression
```

```
myf=@(x,y)x-y
myf(2,3)
```

Инлайн функции в MatLab

```
f = inline('3*sin(2*x.^2)') % аргументов может быть несколько
f(3)
```

Вложенные функции. Вы можете определить функции в теле другой функции.

Символьные переменные

```
syms x y a b
f(a, b) = a*x^2/(sin(3*y - b));
symvar(f)
```