Table of Contents

Cell Arrays

Cell arrays contain data in cells that you access by numeric indexing. Common applications of cell arrays include storing separate pieces of text and storing heterogeneous data from spreadsheets. For example, store temperature data for three cities over time in a cell array.

```
temperature(1,:) = {'01-Jan-2010', [45, 49, 0]};
temperature(2,:) = {'03-Apr-2010', [54, 68, 21]};
temperature(3,:) = {'20-Jun-2010', [72, 85, 53]};
temperature(4,:) = {'15-Sep-2010', [63, 81, 56]};
temperature(5,:) = {'31-Dec-2010', [38, 54, 18]};

% Plot the temperatures for each city by date.
allTemps = cell2mat(temperature(:,2));
dates = datenum(temperature(:,1), 'dd-mmm-yyyy');

plot(dates,allTemps,'--*')
datetick('x','mmm')

% Cell array heterogeneous data
A{1,1} = {'This is the first cell.'};
A{1,2} = {[5+j*6 , 4+j*5]};
A{2,1} = {[1 2 3; 4 5 6; 7 8 9]};
A{2,2} = {{'Tim'; 'Chris'}}

% Graphical way to indicate the data format of the cell elements
cellplot(A)
% display the data inside the cell array
celldisp(A)

s=A{2,2}
s{1,1}
[st1 st2]=deal(s{1:2})


A =

    {1x1 cell}    {1x1 cell}
    {1x1 cell}    {1x1 cell}


A{1,1}{1} =
```

```
This is the first cell.


A{2,1}{1} =

     1     2     3
     4     5     6
     7     8     9



A{1,2}{1} =

   5.0000 + 6.0000i   4.0000 + 5.0000i



A{2,2}{1}{1} =

Tim


A{2,2}{1}{2} =

Chris


s =

    {2x1 cell}


ans =

    'Tim'
    'Chris'


Index exceeds matrix dimensions.

Error in cell_structure (line 33)
[st1 st2]=deal(s{1:2})
```

Structure Arrays

Structure arrays contain data in fields that you access by field name. For example, store patient records in a structure array.

```
patient(1).name = 'John Doe';
patient(1).billing = 127.00;
patient(1).test = [79, 75, 73; 180, 178, 177.5; 220, 210, 205];

patient(2).name = 'Ann Lane';
```

```
patient(2).billing = 28.50;
patient(2).test = [68, 70, 68; 118, 118, 119; 172, 170, 169];
% Create a bar graph of the test results for each patient.
numPatients = numel(patient);
for p = 1:numPatients
   figure
   bar(patient(p).test)
   title(patient(p).name)
end
```

Combine different structure to a larger structure

Generate a structure for the social science

```
clear all;
S(1).name = 'Ed Plum';
S(1).score = 83;
S(1).grade = 'B+';
S(2).name = 'Toni Miller';
S(2).score = 91;
S(2).grade = 'A-';
% or an entire element can be added with a single statement.
S(3) = struct('name','Jerry Garcia','score',70,'grade','C');
```

generate another class

```
T.name = 'Ed Plum';
T.score = [ 83  89];

T(2).name = 'Toni Miller';
T(2).score = 91;

% or an entire element can be added with a single statement.
T(3) = struct('name','Jerry Garcia','score',70);


class(1).name='MATH101';
class(1).student=S;
class(2).name='LAN102';
class(2).student=T;

class(3).name='MATLAB102';
class(3).student=S;

% access the data with field name
class(2).student(2).score
class(3).student(1).name
class(3).student(1)= struct('name','Jerry
 Garcia','score',70,'grade','C');
class(3).student(1).name
```

Mapping the data to a matrix

```
clear student  % ## student ##
student(1) = struct('name', 'Banny', 'scores', [85,80,92,78]);
student(2) = struct('name', 'Joey', 'scores', [80,85,90,88]);
student(3) = struct('name', 'Betty', 'scores', [88,82,90,80]);

% check the field name in your structure
nm=fieldnames(student);
numb=numel(student);

%
% clear student
% N=input('   number of student:   ');
% score=zeros(2,N);
% % input the name and score of the student evaluate the average score
% for i=1:N
%     str1= input('student name:','s');
%     student(i).name=str1;;
% score(1,i)=input('math score:   ');
% score(2,i)=input('english score:   ');
% student(i).scores=[score(1,i) score(2,i)];
% avg(i)=(score(1,i)+score(2,i))/2; % avg(i) = sum(score(:,i))/2;
% student(i).avg=avg(i);
% end
%
% % output value
% for i=1:N
%      fprintf('the average score of %s is %3.2f
 \n',student(i).name,student(i).avg);
% end
% save student

% Dynamic field name; use the current data as the field name
currentDate = datestr(now,'mmmdd'); % field name will change based on
 the date
myStruct.(currentDate) = [1,2,3]; % the currentDate is a char string
myStruct(2).(currentDate) = [4,5,6];

% change the value of the structure using field name
% one can also using the getfield & setfield
clear student  % ## student ##
student(1) = struct('name', 'Banny', 'scores', [85,80,92,78]);
student(2) = struct('name', 'Joey', 'scores', [80,85,90,88]);
student(3) = struct('name', 'Betty', 'scores', [88,82,90,80]);

% use setfield to set the field data and getfield to get the data of
 the field
student(2).name='john';
student=setfield(student,{2},'name','johns')
score3=student(2).scores(2)
score4=getfield(student,{2},'score',{2})
```

```
% one can also using deal to map the value of the structure to the
 output
% variable
[nam1,nam2,nam3]=deal(student.name);
[sc1,sc2,sc3]=deal(student.scores);

% calculate the avg score
a1=cat(1,student.scores)
avg=mean(a1)


% The commands struct2cell transfer structure to cell array

% create the structure
clear s, s.category = 'tree';
s.height = 37.4; s.name = 'birch';

% Converting the structure to a cell array,
c = struct2cell(s);
c
c{1,1}
```

Published with MATLAB® R2016a