Module 09: Data Structure

Chul Min Yeum
Assistant Professor
Civil and Environmental Engineering
University of Waterloo, Canada

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Module 09: Learning Outcomes

- Explain cell and structure data type
- Describe a problem that require these new data type
- Access values in cell and structure array
- Illustrate the difference between cell and structure data type
- Understand vectorization of the operations involving cell arrays

Cell Array

- A **cell array** is a **data type** that can **store different types and sizes of values** in its elements (cell).
- For example, when you store combinations of text and numbers as one variable which are often read from **spreadsheets**, this cell type is very useful.
- A cell array could be a vector (row or column) or matrix of cells.
- A cell array is an array, so indices are used to refer to the elements.
- The syntax used to create a cell array is curly braces { } instead of [ ] (e.g. `var = {'abc', 3, [1 2 3]}`)
- The cell function can also be used to preallocate empty cells by passing the dimensions of the cell array, e.g. `cell(4,2)`.
- The contents in cells can be access (write and read) by indexing with curly braces {}.
- () is to **refer a set of cells**, for example, to define a subset of the cell array.

Create Cell Array

```
cl_info = {‘Chul Min’, ‘CIVE’, 1076123, [80 90], [70 30 50], ...
[4 5 1 2], ‘4B’};
```

option 1

```
cl_info = cell(1,7);

cl_info{1,1} = 'Chul Min';
cl_info{1,2} = 'CIVE';
cl_info{1,3} = 1076123;
cl_info{1,4} = [80 90];
cl_info{1,5} = [70 30 50];
cl_info{1,6} = [4 5 1 2];
cl_info{1,7} = '4B';
```

option 2

😊: Option 2 is more general when you put multiple input data.

```
>> cl_info

cl_info =

  1×7 cell array

  Columns 1 through 3

    {'Chul Min'}    {'CIVE'}    {[1076123]}

  Columns 4 through 5

    {1×2 double}    {1×3 double}

  Columns 6 through 7

    {1×4 double}    {'4B'}
```

📖: `...` is to continue long statements on multiple lines

Example: How to Create and Access Cell Array

Suppose that you want to store the following data using a cell array:

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

```
cl_info = cell(3,7);

cl_info{1,1} = 'Chul Min';
cl_info{1,2} = 'CIVE';
cl_info{1,3} = 1076123;
cl_info{1,4} = [80 90];
cl_info{1,5} = [70 30 50];
cl_info{1,6} = [4 5 1 2];
cl_info{1,7} = '4B';

cl_info{2,1} = 'Noreen';
cl_info{2,2} = 'ENVE';
```

```
% continue
cl_info{2,3} = 3026327;
cl_info{2,4} = [100 70];
cl_info{2,5} = [10 20 70];
cl_info{2,6} = [2 7 8 9];
cl_info{2,7} = '2A';

cl_info{3,1} = 'Vlad';
cl_info{3,2} = 'ENVE';
cl_info{3,3} = 2046426;
cl_info{3,4} = [50 90];
cl_info{3,5} = [90 60 80];
cl_info{3,6} = [1 2 6 2];
cl_info{3,7} = '2A';
```

⚠: We cannot store multiple character vectors using basic array type when their lengths are different.

Example: How to Create and Access Cell Array (Continue)

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

Q1. What's Chul Min's ID? Assign to '`test_id`'.

```
[nr, nc] = size(cl_info);
for ii=1:nr
   if strcmp(cl_info{ii, 1}, 'Chul Min')
      test_id = cl_info{ii,3};
      break;
   end
end
```

😊: You can shorten the following script

```
test_exam = cl_info{ii,4};
test_grade = test_exam(1);

% shortened
test_grade = cl_info{ii,4}(1);
```

Q2. What's Noreen's the first exam grade? Assign to '`test_grade`'.

```
[nr, nc] = size(cl_info);
for ii=1:nr
   if strcmp(cl_info{ii, 1}, 'Noreen')
      test_exam = cl_info{ii,4};
      test_grade = tecl_exam(1);
      break;
   end
end
```

What’s the Difference Between {} and ()?

Challenging

```
cl_info = cell(3,7);

cl_info{1,1} = 'Chul Min';
cl_info{1,2} = 'CIVE';
cl_info{1,3} = 1076123;
cl_info{1,4} = [80 90];
cl_info{1,5} = [70 30 50];
cl_info{1,6} = [4 5 1 2];
cl_info{1,7} = '4B';

cl_info{2,1} = 'Noreen';
cl_info{2,2} = 'ENVE';
cl_info{2,3} = 3026327;
cl_info{2,4} = [100 70];
cl_info{2,5} = [10 20 70];
cl_info{2,6} = [2 7 8 9];
cl_info{2,7} = '2A';

cl_info{3,1} = 'Vlad';
cl_info{3,2} = 'ENVE';
cl_info{3,3} = 2046426;
cl_info{3,4} = [50 90];
cl_info{3,5} = [90 60 80];
cl_info{3,6} = [1 2 6 2];
cl_info{3,7} = '2A';
```

```
cl_info = cell(3,7);

cl_info(1,:) = {‘Chul Min’, ‘CIVE’, 1076123, [80 90], ...
[70 30 50], [4 5 1 2], ‘4B’};

cl_info(2,:) = {‘Chul Min’, ‘CIVE’, 1076123, [80 90], ...
[70 30 50], [4 5 1 2], ‘4B’};

cl_info(3,:) = {‘Chul Min’, ‘CIVE’, 1076123, [80 90], ...
[70 30 50], [4 5 1 2], ‘4B’};
```

```
>> class(cl_info{1,1})

ans =

    'char'

>> class(cl_info(1,1))

ans =

    'cell'
```

Read the value inside the cell

Read the cell itself

📖: ( ) is to refer a set of cells. You refer cell(s) themselves and assign a cell or cell array to the space. {} is to refer a value in a cell. You refer a space inside the cell and assign the value to the cell. Remember that cell is a data type!

📖: **`class`** is a function of determining a class of object.

How to Create and Access Cell Array using Vectorization Challenging

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

Q1. What's Noreen's ID? Assign to '`test_id`'.

```
lg_vec = strcmp(cl_info(:,1), 'Chul Min');
idx = find(lg_vec);
test_id = cl_info{idx, 3};
```

: We don't need to do
`find(lg_vec == 1)`
because `find` function is designed to find the 1 (`true`) values.

```
lg_vec          idx       1

  1
  0             test_id   1076123
  0
```

: `strcmp` supports a cell array of character vectors as inputs. The output produces the logical array with the same size as the input array. Here, the input is the column vector of a cell array that contains character vectors (names). The function compares each element with the second input of the character vector.

How to Create and Access Cell Array using Vectorization Challenging

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

Q2. What's name of the student whose ID is 3026327? Assign the name to '`test_name`'.

```
mat_id = [cl_info{:,3}];
lg_vec = mat_id == 3026327;
idx = find(lg_vec);
test_name = cl_info{idx, 1};
```

📖: This is the way of vectorizing the numeric cell array. `cl_info{:,3}` is the script to read all values in the referred cells. If it is inside `[]`, this becomes a numeric array. You can use a `cell2mat` built-in function. Once we convert the cell array to the numeric array, we can solve this problem using a vectorized script.

`mat_id`

1076123
3026327
2046426

`lg_vec`

0
1
0

`idx`

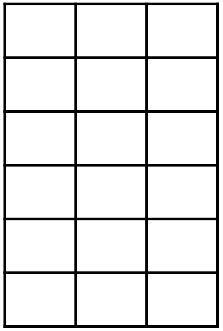
2

`test_name`

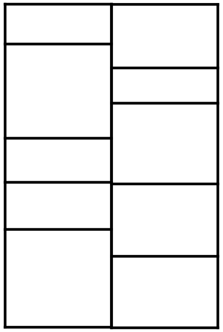
N	o	r	e	e	n

Structure Variables

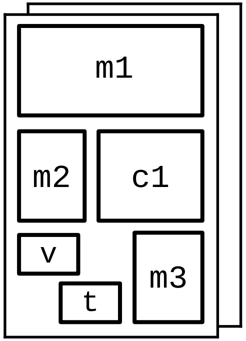
- A structure array is a data type that groups related data using data containers called field.
- Each field can contain any type of data.
- Fields are given names; they are referred to as **structurename.fieldname** using the dot operator.



Basic array (A)



Cell array (B)



Structure array (C)

```
A = [1 2 3; 4 5 6];
B = {'A', 'B', 'C'};
C.a = 'CIVE';
C.b = 'ENVE';
```

`A(1)`	`1`
`B{2}`	`'B'`
`C.a`	`'CIVE'`

Create Structure Array

```
st_info(1) = struct('name', 'Chul Min', 'program', 'CIVE', 'id', ...
    1076123, 'exam', [80 90], 'quiz', [70 30 50], 'homework', ...
    [4 5 1 2], 'cohort', '4B');
```

option 1

```
st_info(1).name = 'Chul Min';
st_info(1).program = 'CIVE';
st_info(1).id = 1076123;
st_info(1).exam = [80 90];
st_info(1).quiz = [70 30 50];
st_info(1).homework = [4 5 1 2];
st_info(1).cohort = '4B';
```

option 2

```
>> st_info

st_info =

  struct with fields:

        name: 'Chul Min'
     program: 'CIVE'
          id: 1076123
        exam: [80 90]
        quiz: [70 30 50]
    homework: [4 5 1 2]
      cohort: '4B'
```

😌: You can initialize the structure array in the beginning however, there might not be a formal way to do it. If you know the size of the structure array (e.g., # of students in the above case), you can do

```
st_info(10) = struct;
```

Example: How to Create and Access Structure Array

Suppose that you want to store the following data using a structure array:

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

```
st_info(1).name = 'Chul Min';
st_info(1).program = 'CIVE';
st_info(1).id = 1076123;
st_info(1).exam = [80 90];
st_info(1).quiz = [70 30 50];
st_info(1).homework = [4 5 1 2];
st_info(1).cohort = '4B';

st_info(2).name = 'Noreen';
st_info(2).program = 'ENVE';
st_info(2).id = 3026327;
st_info(2).exam = [100 70];
```

```
st_info(2).exam = [100 70];
st_info(2).quiz = [10 20 70];
st_info(2).homework = [2 7 8 9];
st_info(2).cohort = '2A';

st_info(3).name = 'Vlad';
st_info(3).program = 'ENVE';
st_info(3).id = 2046426;
st_info(3).exam = [50 90];
st_info(3).quiz = [90 60 80];
st_info(3).homework = [1 2 6 2];
st_info(3).cohort = '2A';
```

Example: How to Create and Access Structure Array (Continue)

Name	Program	ID	Exam	Quiz	Homework	Cohort
Chul Min	CIVE	1076123	[80 90]	[70 30 50]	[4 5 2 1]	4B
Noreen	ENVE	3026327	[100 70]	[10 20 70]	[2 7 8 9]	2A
Vlad	ENVE	2046426	[50 90]	[90 60 80]	[1 2 6 2]	2A

Q1. What's Chul Min's ID? Assign to '`test_id`'.

```
nst = numel(st_info);
for ii=1:nst
   if strcmp(st_info(ii).name, 'Chul Min')
      test_id = st_info(ii).id;
      break;
   end
end
```

😊: You can shorten the following script

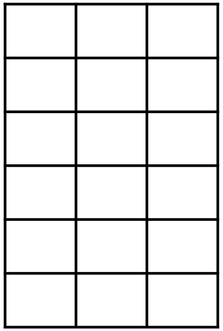
```
test_exam = st_info(ii).exam;
test_grade = test_exam(1);

% shortened
test_grade = st_info(ii).exam(1);
```

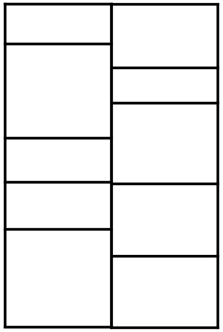
Q2. What's Noreen's the first exam grade? Assign to '`test_grade`'.

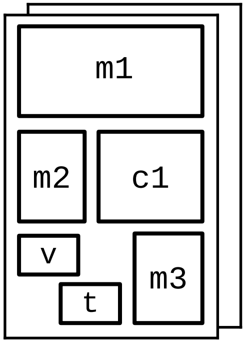
```
nst = numel(st_info);
for ii=1:nst
   if strcmp(st_info(ii).name, ‘Noreen')
      test_exam = st_info(ii).exam;
      test_grade = test_exam(1);

   end
end
```

Cell Arrays vs Structures

- Cell arrays are arrays, so they can be indexed. That means that you can loop though the elements in a cell array – or have MATLAB do that for you by using a vectorized code
- **Structs are not indexed**, so you cannot loop. However, the field names are mnemonic, so it is clearer what is being stored in a struct.
- For example: variable{1} vs. variable.weight: which is more mnemonic?


Basic array (A)


Cell array (B)



Structure array (C)

```
A = [1 2 3; 4 5 6]
A(1)
```

```
B = {‘A’, ‘B’, ‘C’};
B{1}
```

```
B.a = ‘CIVE’;
B.b = ‘ENVE’;
B.a
```

Passing Multiple Input Variable to Function using Structure   Optional

Suppose that you are making a function named 'CompGrade' to compute a student grade using his/her record during a term. The ten records are stored in each variable named 'r1', 'r2', …. 'r10'. Thus, the function accepts for 10 inputs and produce one output. How to design your function?

```
function final_grade = CompGrad(r1, r2, ..., r10)
Do something
end
```

```
function final_grade = CompGrad(record)
r1 = record.r1;
r2 = record.r2;
:
r10 = record.r10;
end
```

😊: Rather than passing all ten input variables to the function, we can pass one singe structure variable and store all ten variables in the structure variable.

script

```
...
record.r1 = r1;
record.r2 = r2;
:
record.r10 = r10;
final_grade = ...
CompGrad(record);
...
```