MAT 1206 – Introduction to MATLAB

CHAPTER 02: Fundamental Operators and Commands

Lesson 5: Input and output commands

Content

- Request User Input
- Create Dialog Box to Gather User Inputs
- Import and Export Data

Request User Input

x = **input**(prompt) displays the text in prompt and waits for the user to input a value and press the Return key. The user can enter expressions, like pi/4 or rand(3), and can use variables in the workspace.

- If the user presses the Return key without entering anything, then input returns an empty matrix.
- If the user enters an invalid expression at the prompt, then MATLAB® displays the relevant error message, and then redisplays the prompt.

```
prompt = "What is the original value? ";
x = input(prompt)
```

```
What is the original value? 55

x =

    55
```

```
What is the original value? w
Error using input
Unrecognized function or variable 'w'.
```

Cont.

txt = **input(prompt,"s")** returns the entered text, without evaluating the input as an expression.

Example:

```
name=input("What is your name? ","s")
```

```
What is your name?Sandaruwan

name =

    'Sandaruwan'
```

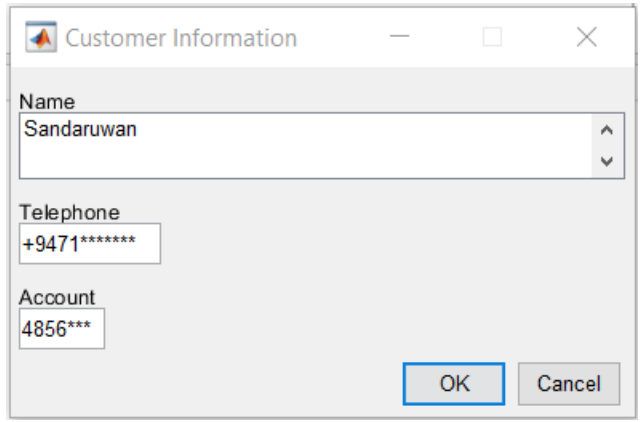
Create Dialog Box to Gather User Input

answer = **inputdlg**(prompt, dlgtitle, dims, definput) creates a modal dialog box containing one or more text edit fields and returns the values entered by the user.

The return values are elements of a cell array of character vectors.

Example:

```
prompt = {'Name','Telephone','Account'};
dlgtitle = 'Customer Information';
dims = [2 50; 1 12; 1 7];
definput = {'Sandaruwan','+9471*******','4856***'};
answer = inputdlg(prompt,dlgtitle,dims,definput)
```



Import and Export Data

Importing data in MATLAB means loading data from external files and exporting data means write data into external files.

MATLAB provides various functions and techniques to import and export data. Depending on the file format and structure of your data, you can choose the appropriate method to import or export data.

To easily read and write data from various file formats in MATLAB, you can use the functions **readmatrix** and **writematrix**. These functions were introduced in MATLAB R2019a and provide a simple way to handle different file formats without the need for complex parsing or formatting.

Cont.

readmatrix: The **readmatrix** function reads numeric data from a file and returns it as a matrix. It can handle various file formats, including text files, CSV files, and Excel files.

```
data=readmatrix('Matrix data file');
```

```
data =

  106.6000    0.0100  262.6000    0.0300  195.4000    0.0300  418.0000    0.0300  262.0000
  117.4000    0.0100  270.4000    0.0300  188.0000    0.0300  469.0000    0.0300  295.4000
  106.6000    0.0100  279.4000    0.0300  197.0000    0.0300  460.4000    0.0300  302.8000
  110.6000    0.0100  273.0000    0.0300  194.2000    0.0300  447.2000    0.0300  295.8000
  106.6000    0.0100  278.6000    0.0300  197.4000    0.0300  455.2000    0.0300  278.2000
  110.4000    0.0100  271.0000    0.0300  204.2000    0.0300  421.8000    0.0300  250.0000
  114.6000    0.0100  276.4000    0.0300  193.6000    0.0300  465.2000    0.0300  285.8000
```

In this example, 'filename' represents the name of the file you want to read. The **readmatrix** function will automatically detect the file format based on the extension and return the numeric data as a matrix (data in the example).

Cont.

writematrix: The **writematrix** function writes a matrix or a 2D array to a file. It supports multiple file formats, including text files, CSV files, and Excel files.

```
data = magic(5);  % Example data
%writematrix(data, 'filename.csv')

writematrix(data,'matrix data write.csv');
```

In this example, data represents the matrix or 2D array you want to write, and 'filename.csv' is the name of the file you want to create. The **writematrix** function will automatically determine the appropriate file format based on the file extension and write the data accordingly.

Tables

Tables are arrays in tabular form whose named columns can have different types.

T = **table(var1,...,varN)** creates a table from the input variables var1,...,varN. The variables can have different sizes and data types, but all variables must have the same number of rows.

Example:

```
LastName = {'Sanchez';'Johnson';'Li';'Diaz';'Brown'};
Age = [38;43;38;40;49];
Smoker = logical([1;0;1;0;1]);
Height = [71;69;64;67;64];
Weight = [176;163;131;133;119];
BloodPressure = [124 93; 109 77; 125 83; 117 75; 122 80];
>> T = table(LastName,Age,Smoker,Height,Weight,BloodPressure)
```

```
T =
  5×6 table
    LastName      Age    Smoker    Height    Weight    BloodPressure
    ____________    ___    ______    ______    ______    _____________

    {'Sanchez'}    38    true        71        176        124        93
    {'Johnson'}    43    false       69        163        109        77
    {'Li'     }    38    true        64        131        125        83
    {'Diaz'   }    40    false       67        133        117        75
    {'Brown'  }    49    true        64        119        122        80
```

Cont.

To extract values from a table, use curly braces or dot indexing.

T{:,2}

T.Age

T{:,2:4}

```
>> T{:,2}
ans =
    38
    43
    38
    40
    49
>> T.Age
ans =
    38
    43
    38
    40
    49
```

```
>> T{:,2:4}
ans =
    38     1    71
    43     0    69
    38     1    64
    40     0    67
    49     1    64
```

Read both numeric and character data

To read both numeric and character data from files in MATLAB, you can use the **readtable** function with appropriate options. The **readtable** function allows you to read tabular data from different file formats, including those containing a mix of numeric and character data.

```
data = readtable('filename');
```

By default, the readtable function will infer the data types for each column based on the content of the file. Numeric values will be stored as double precision numbers, and character values will be stored as strings.

Cont.

To write both numeric and character data to files in MATLAB, you can use the **writetable** function with a MATLAB table that contains a mix of numeric and character variables. The table can then be written to various file formats, including text files, CSV files, and Excel files.

```
% Create an example MATLAB table with mixed numeric and character variables
data = table([1; 2; 3], {'A'; 'B'; 'C'}, 'VariableNames', {'Numbers', 'Letters'});

% Write the table to a CSV file
writetable(data, 'filename.csv‘)

% Write the table to a text file
writetable(data, 'filename.txt')
```

Questions/queries?