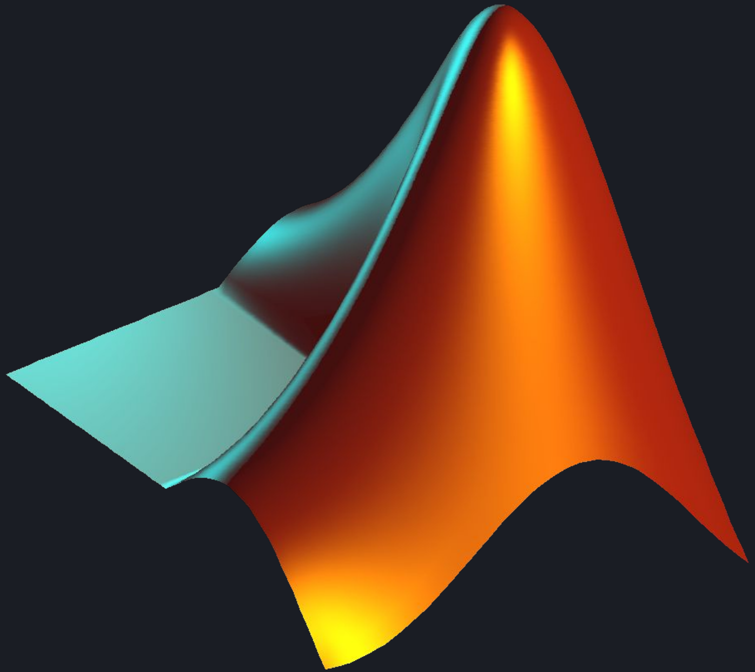
Develop Programming skills and Proficiency in MATLAB



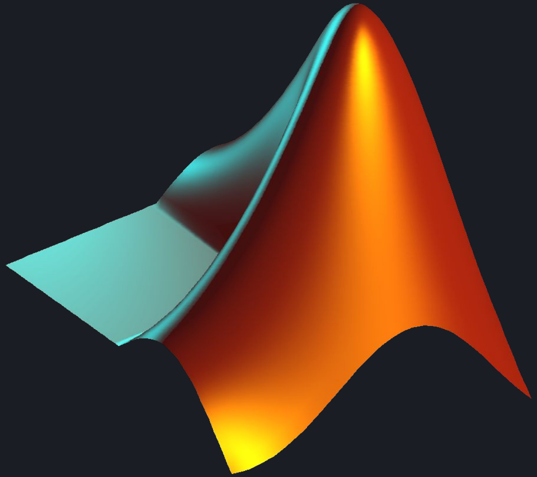
OS-Academy

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

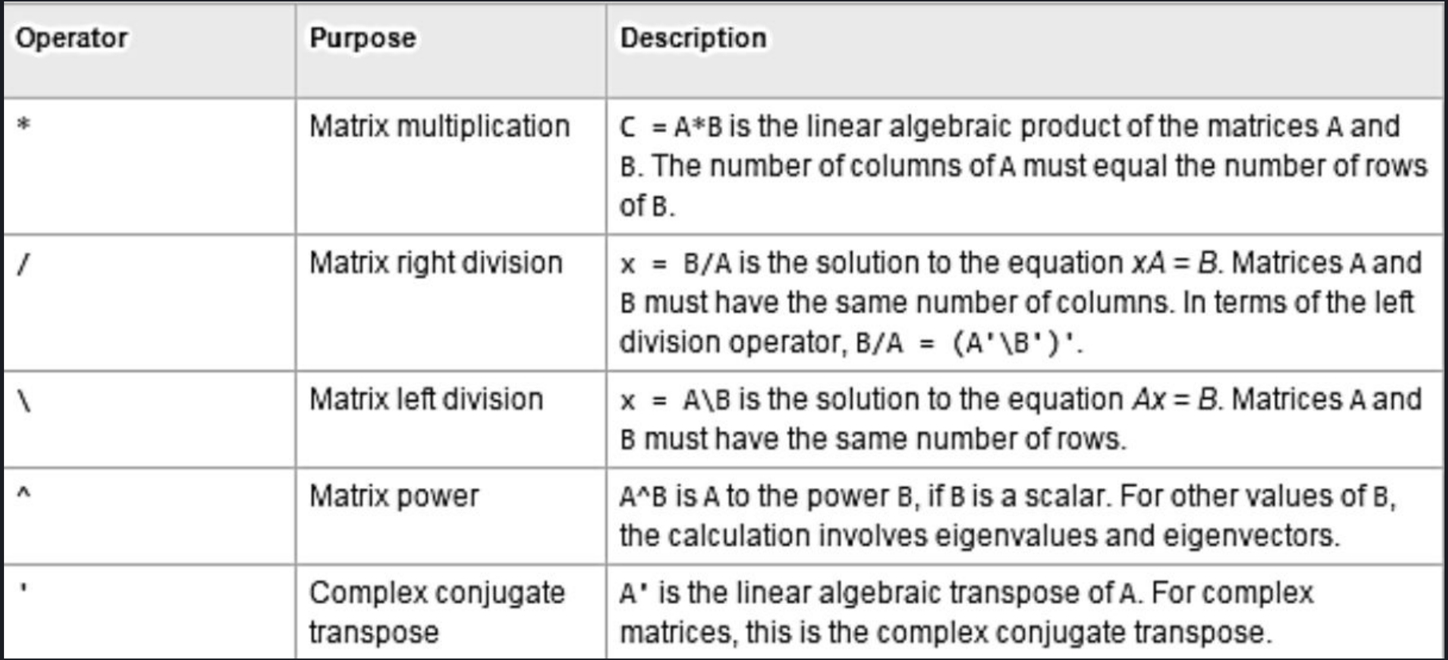
Session Content

- Matrix OPeration
- Input and Output Commands
- Programming with MATLAB
- Dealing with MATLAB functions



Develop Programming skills and Proficiency in MATLAB

Matrix Operations



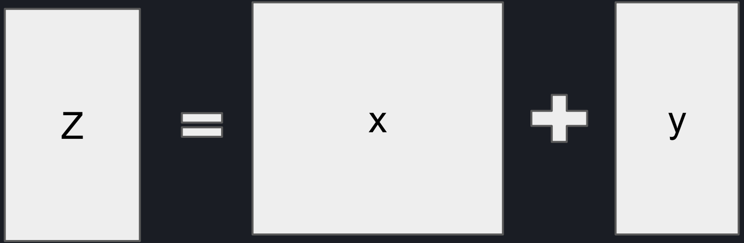
Operator	Purpose	Description
*	Matrix multiplication	C = A*B is the linear algebraic product of the matrices A and B. The number of columns of A must equal the number of rows of B.
/	Matrix right division	x = B/A is the solution to the equation *xA* = *B*. Matrices A and B must have the same number of columns. In terms of the left division operator, B/A = (A'\B')'.
\	Matrix left division	x = A\B is the solution to the equation *Ax* = *B*. Matrices A and B must have the same number of rows.
^	Matrix power	A^B is A to the power B, if B is a scalar. For other values of B, the calculation involves eigenvalues and eigenvectors.
'	Complex conjugate transpose	A' is the linear algebraic transpose of A. For complex matrices, this is the complex conjugate transpose.

Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Multiplication**

$$x = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \qquad y = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$



Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Multiplication**

$$\begin{array}{|c|c|c|}\hline a11 & a12 & a13 \\ \hline a21 & a22 & a23 \\ \hline \end{array} \quad \begin{array}{|c|}\hline b11 \\ \hline b21 \\ \hline b31 \\ \hline \end{array} = \begin{pmatrix} a11*b11 + a12*b21 + a13*b31 \\ a21*b11 + a22*b21 + a23*b31 \end{pmatrix}$$

OS-Academy

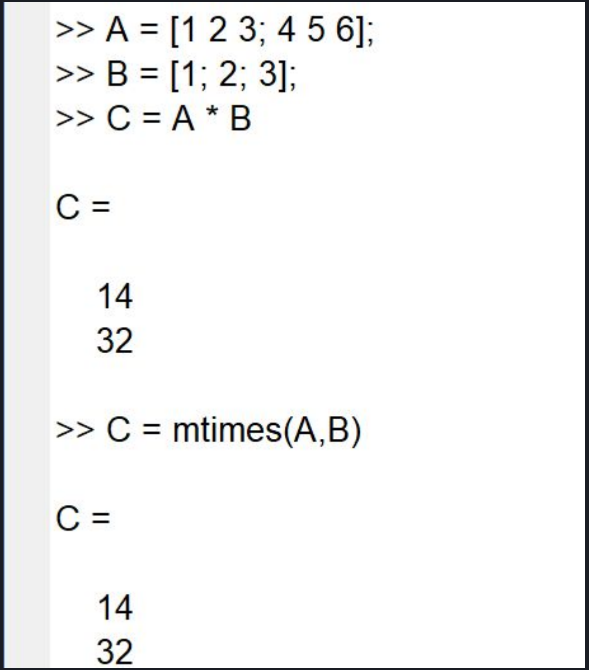
Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Multiplication**

$$x = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \qquad y = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

$$Z = x * y = \begin{pmatrix} 1*1 + 2*2 + 3*3 \\ 4*1 + 5*2 + 6*3 \end{pmatrix} = \begin{pmatrix} 14 \\ 32 \end{pmatrix}$$



```
>> A = [1 2 3; 4 5 6];
>> B = [1; 2; 3];
>> C = A * B

C =

    14
    32

>> C = mtimes(A,B)

C =

    14
    32
```

Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Division**
  - it is used for solving linear equations

$$Ax = B$$

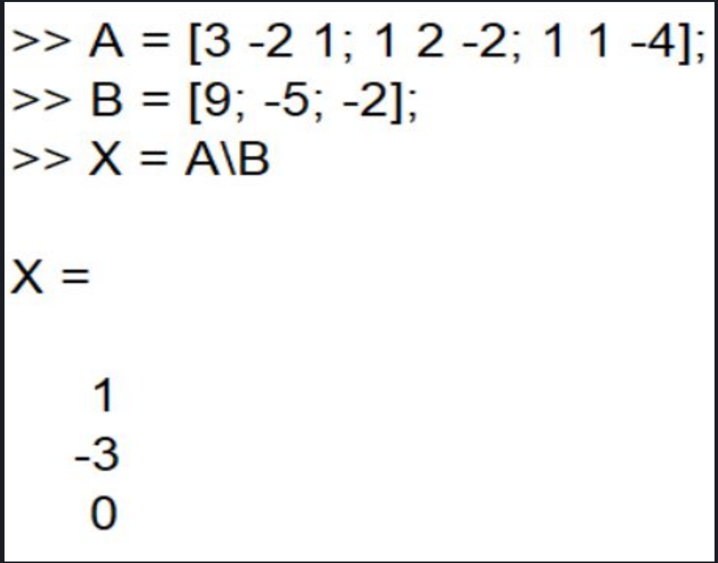
- A is a coefficient matrix,
- x is the column vector of unknowns,
- B is the column vector on the right-hand side.

$$\begin{aligned} a_1x + b_1y + c_1z &= C_1 \\ a_2x + b_2y + c_2z &= C_2 \\ a_3x + b_3y + c_3z &= C_3 \end{aligned} \longrightarrow \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix}$$

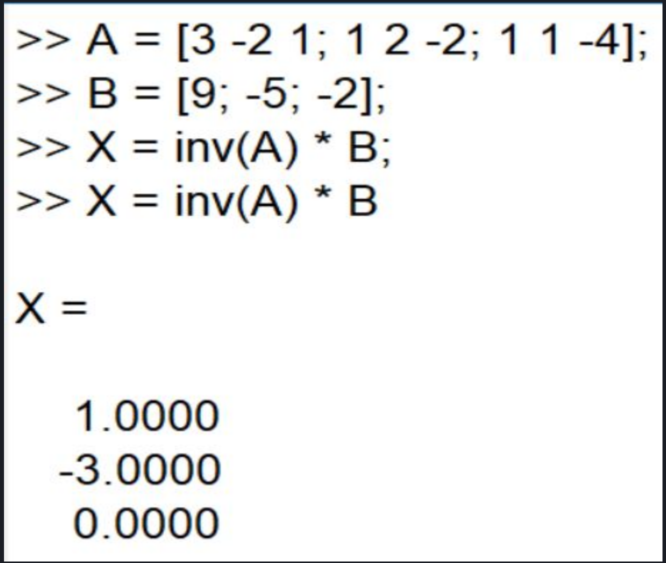
Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Division**
  - it is used for solving linear equations



```
>> A = [3 -2 1; 1 2 -2; 1 1 -4];
>> B = [9; -5; -2];
>> X = A\B

X =

     1
    -3
     0
```

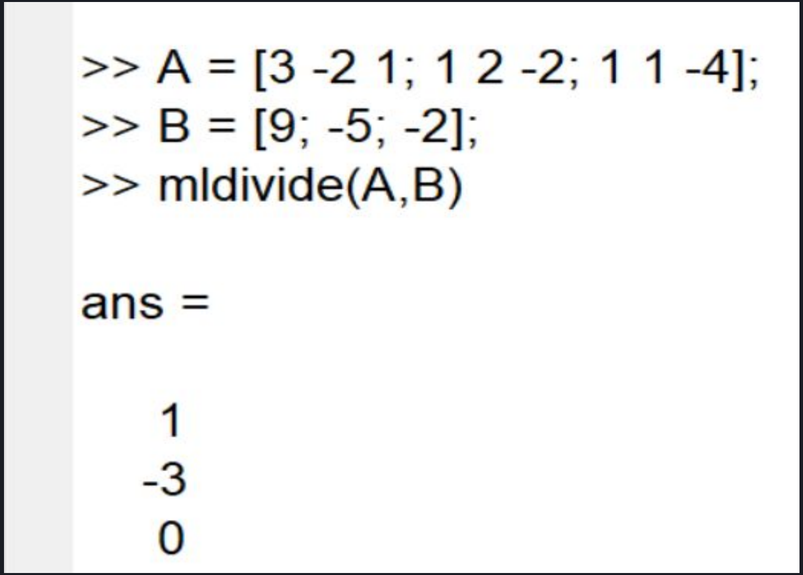


```
>> A = [3 -2 1; 1 2 -2; 1 1 -4];
>> B = [9; -5; -2];
>> X = inv(A) * B;
>> X = inv(A) * B

X =

    1.0000
   -3.0000
    0.0000
```

Develop Programming skills and Proficiency in MATLAB

Matrix Operations

- **Division**
  - it is used for solving linear equations

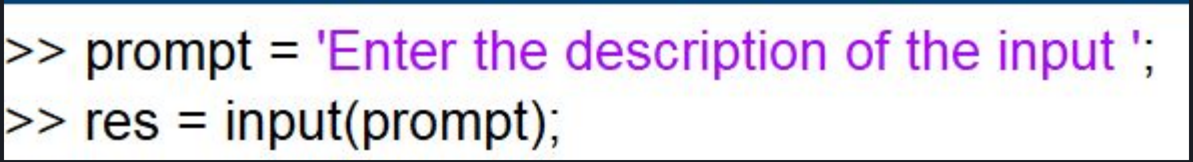


```
>> A = [3 -2 1; 1 2 -2; 1 1 -4];
>> B = [9; -5; -2];
>> mldivide(A,B)

ans =

     1
    -3
     0
```

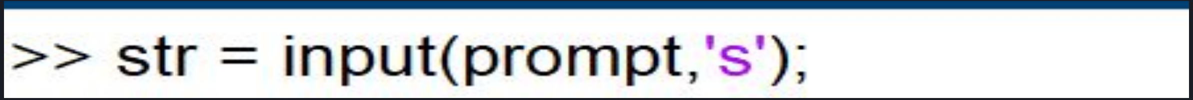
Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

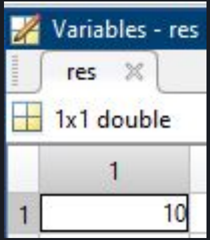
- **Text Interface**
  - **input() :** the `input` function is used to receive user input within a script or function. The function prompts the user for input in the MATLAB command window and returns the entered value.



```
>> prompt = 'Enter the description of the input ';
>> res = input(prompt);
```



```
>> str = input(prompt,'s');
```

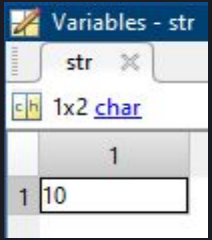


Variables - res

res

1x1 double

	1
1	10

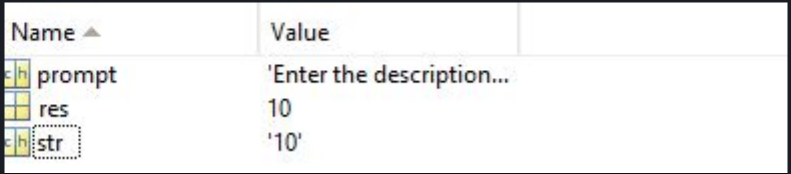


Variables - str

str

1x2 char

	1
1	10



Name	Value
prompt	'Enter the description...
res	10
str	'10'

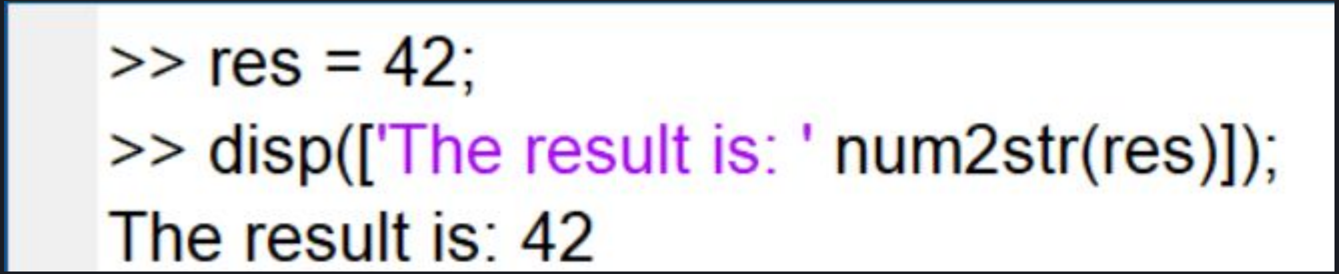
OS-Academy

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **disp() :** the `disp` function is used to display the value of an expression or a message in the command window. It stands for "display" and is commonly used to show the results of calculations or to print informative messages during the execution of a script or function.

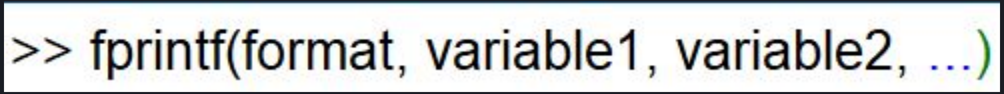


```
>> res = 42;
>> disp(['The result is: ' num2str(res)]);
The result is: 42
```

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **fprintf() :** the `fprintf` function is used to format and print data to the console or to a file. It stands for "formatted print" and allows you to control the appearance of the output by specifying formatting options. This function is particularly useful when you want to display text along with variable values or when writing formatted data to a file.



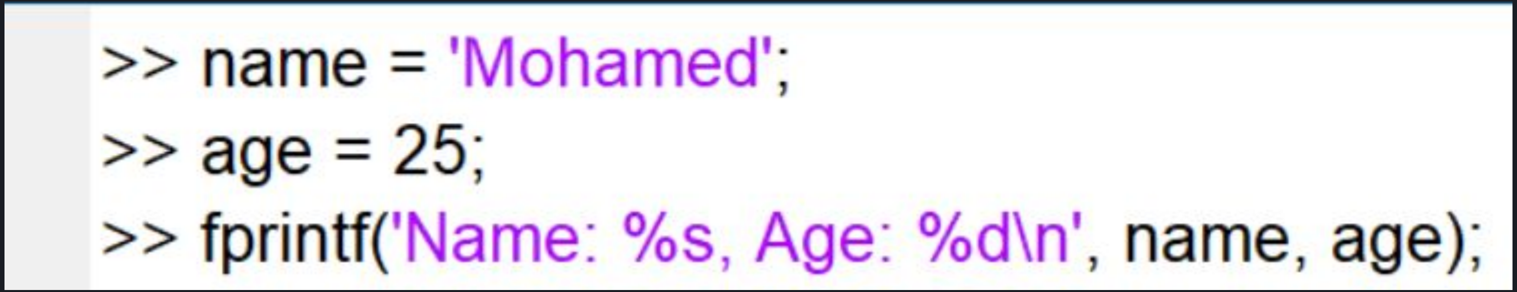
```
>> fprintf(format, variable1, variable2, ...)
```

- format: This is a string specifying the format of the output. It can include placeholders, such as `%s` for strings, `%d` for integers, `%f` for floating-point numbers, and so on.
- variable1, variable2, ...: These are the variables or values that you want to include in the output. The number and types of variables should match the placeholders in the format string.

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **fprintf() :** the `fprintf` function is used to format and print data to the console or to a file. It stands for "formatted print" and allows you to control the appearance of the output by specifying formatting options. This function is particularly useful when you want to display text along with variable values or when writing formatted data to a file.

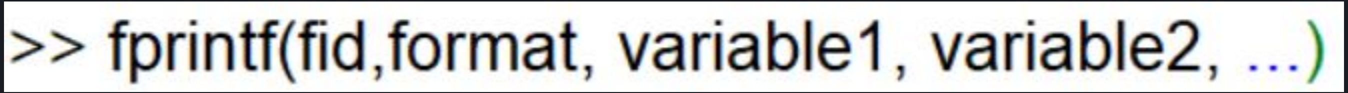


```
>> name = 'Mohamed';
>> age = 25;
>> fprintf('Name: %s, Age: %d\n', name, age);
```

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **fprintf() :** the `fprintf` function is used to format and print data to the console or to a file. It stands for "formatted print" and allows you to control the appearance of the output by specifying formatting options. This function is particularly useful when you want to display text along with variable values or when writing formatted data to a file.



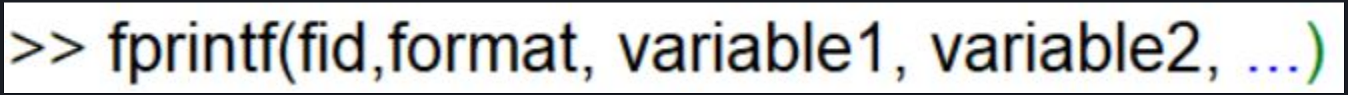
```
>> fprintf(fid,format, variable1, variable2, ...)
```

- To write on a file, fopen(), fclose() are used.
- `fprintf(fid, ...)`: Writes formatted data to the file using the specified file identifier (`fid`). In this example.

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **fprintf() :** the `fprintf` function is used to format and print data to the console or to a file. It stands for "formatted print" and allows you to control the appearance of the output by specifying formatting options. This function is particularly useful when you want to display text along with variable values or when writing formatted data to a file.



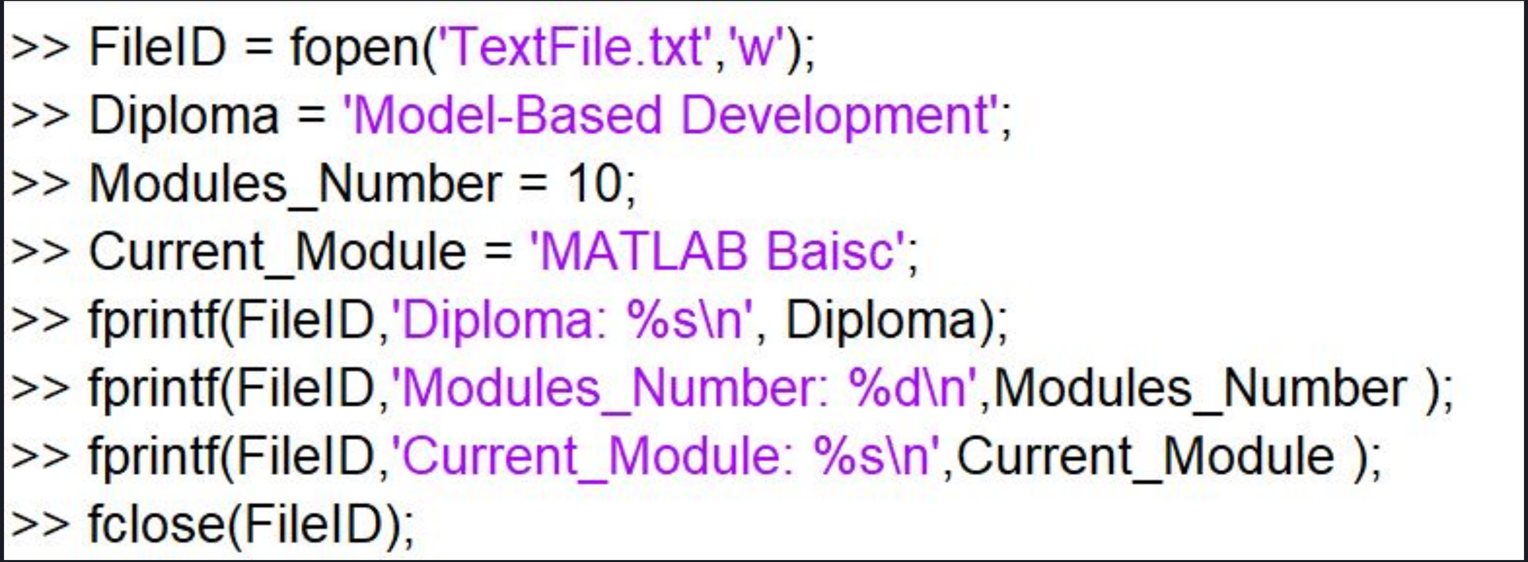
```
>> fprintf(fid,format, variable1, variable2, ...)
```

- To write on a file, fopen(), fclose() are used.
- `fprintf(fid, ...)`: Writes formatted data to the file using the specified file identifier (`fid`). In this example.

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - fprintf()

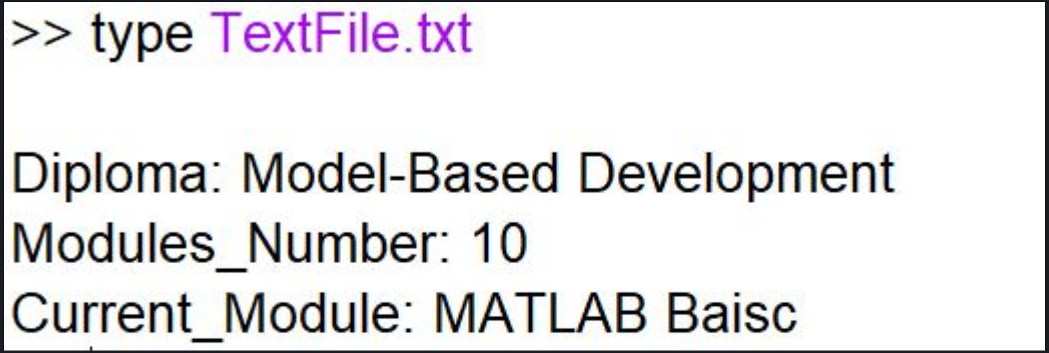


```
>> FileID = fopen('TextFile.txt','w');
>> Diploma = 'Model-Based Development';
>> Modules_Number = 10;
>> Current_Module = 'MATLAB Baisc';
>> fprintf(FileID,'Diploma: %s\n', Diploma);
>> fprintf(FileID,'Modules_Number: %d\n',Modules_Number );
>> fprintf(FileID,'Current_Module: %s\n',Current_Module );
>> fclose(FileID);
```

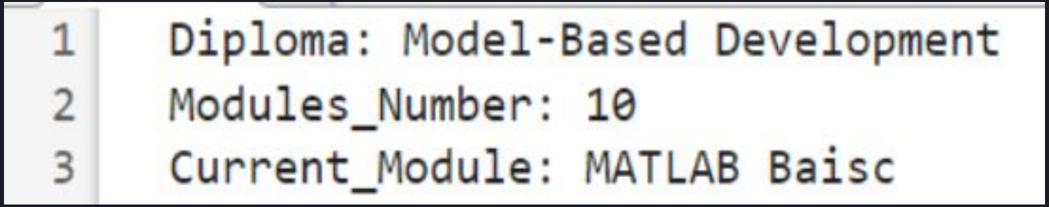
Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- Text Interface
  - fprintf()



```
>> type TextFile.txt

Diploma: Model-Based Development
Modules_Number: 10
Current_Module: MATLAB Baisc
```

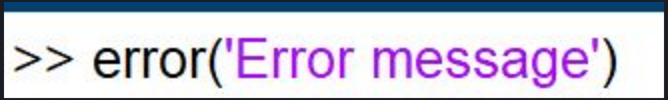


```
Diploma: Model-Based Development
Modules_Number: 10
Current_Module: MATLAB Baisc
```

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **error():** the `error` function is used to generate an error message and terminate the execution of a script or function. It allows you to programmatically handle exceptional conditions by specifying an error message to be displayed when an error occurs.



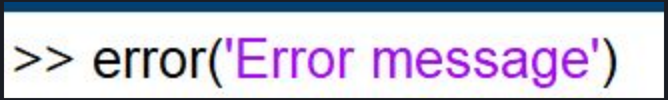
```
>> error('Error message')
```

Error message: This is a string that describes the nature of the error. It is the message that will be displayed when the error occurs.

Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - **error():** the `error` function is used to generate an error message and terminate the execution of a script or function. It allows you to programmatically handle exceptional conditions by specifying an error message to be displayed when an error occurs.



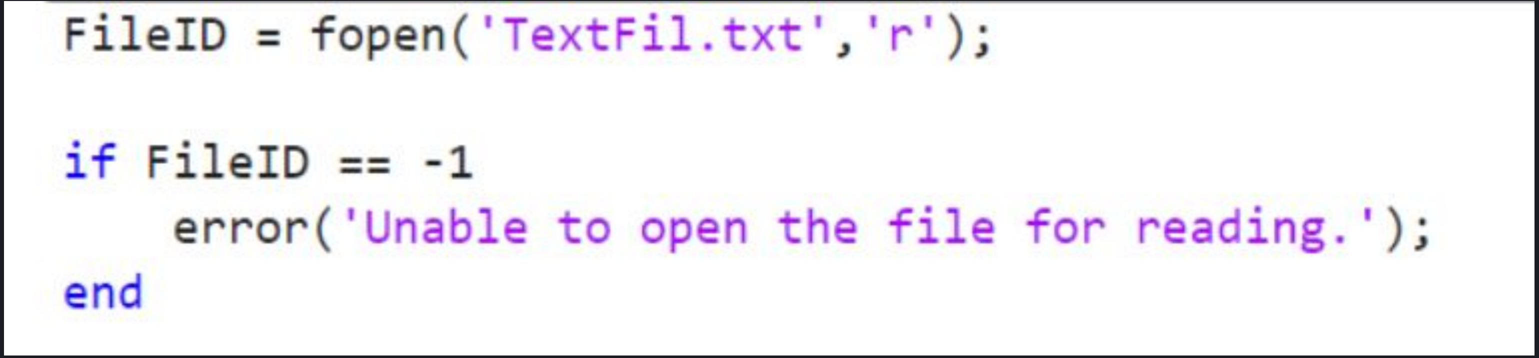
```
>> error('Error message')
```

Error message: This is a string that describes the nature of the error. It is the message that will be displayed when the error occurs.

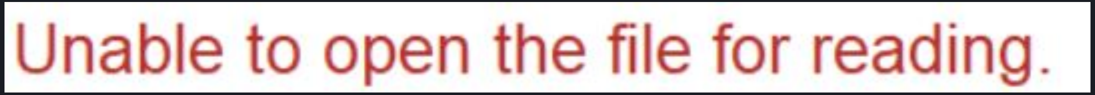
Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- Text Interface
  - error()



```
FileID = fopen('TextFil.txt','r');

if FileID == -1
    error('Unable to open the file for reading.');
end
```



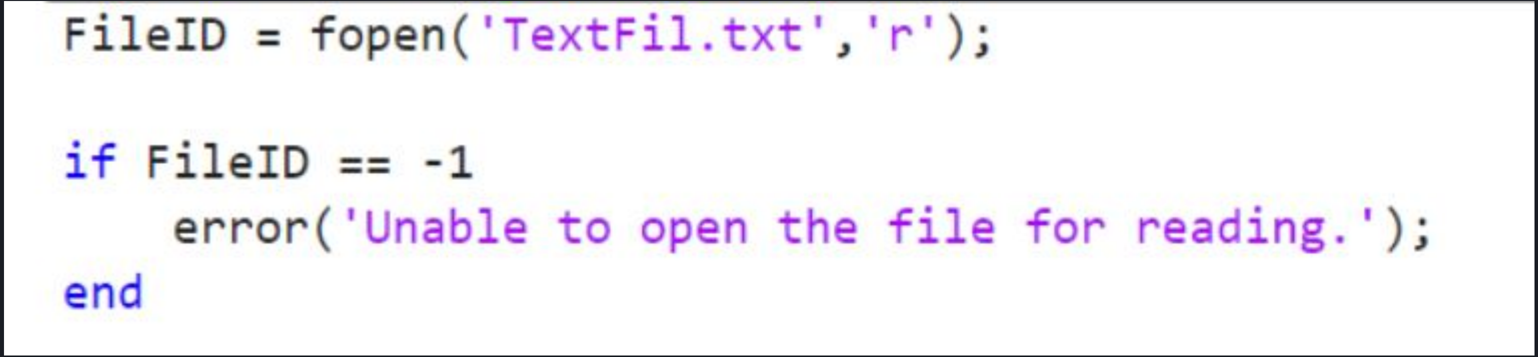
```
Unable to open the file for reading.
```

Model-Based Development Program

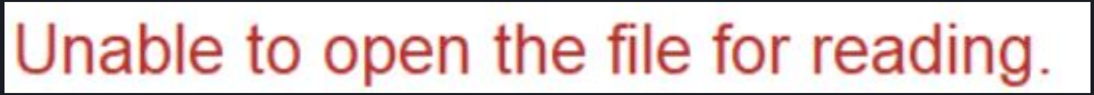
Develop Programming skills and Proficiency in MATLAB

Input & Output Commands

- **Text Interface**
  - error()



```
FileID = fopen('TextFil.txt','r');

if FileID == -1
    error('Unable to open the file for reading.');
end
```

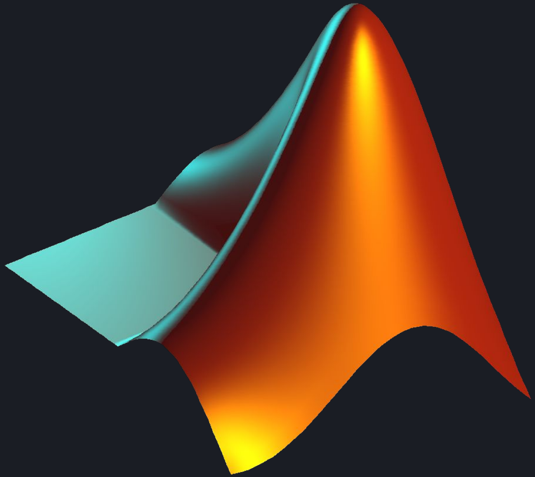


```
Unable to open the file for reading.
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

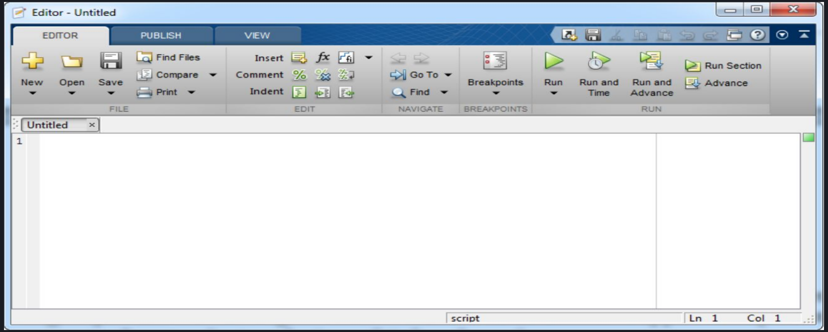
- **Enter commands**
  - Direct
    - Any command written on the command window
    - directly processed and the output will directly appear on the command window
  - Script
    - M-file
    - Function

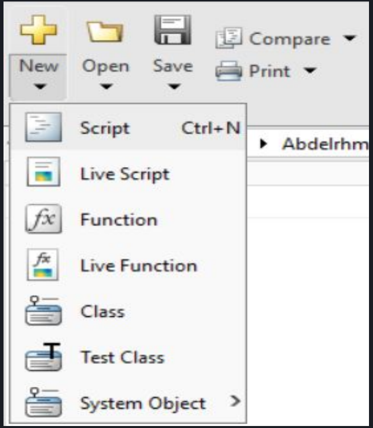


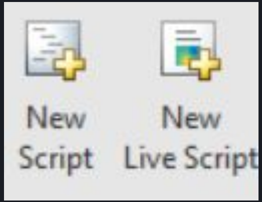
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **M-File Script**
  - A script is a program written in MATLAB language.
  - Scripts allow grouping a series of command lines in a file for easy execution.
  - They offer the advantage of saving and running the same sequence multiple times.
  - The extension .m is used for MATLAB script files.
  - These statements do a certain function.
  - To Create a script click on **New Script** in Command bar



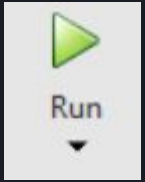




Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

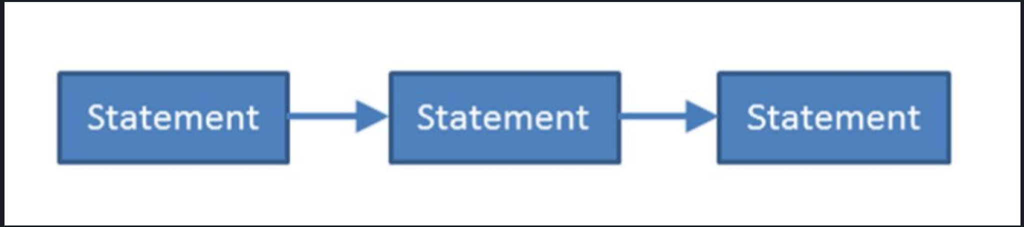
- **M-File Script**
  - Condition for choosing m-file name
    - Refers to the function of the script.
    - Follow Naming Rules of variables.
  - Calling the script
    - Press Run on the editor toolbar
    - write the name of the m-file directly on the command window
    - can be called inside another script

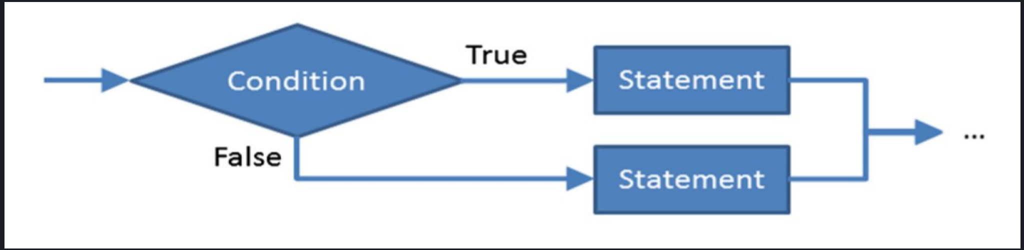


Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - if
  - switch case
  - for loop
  - while
  - try and catch



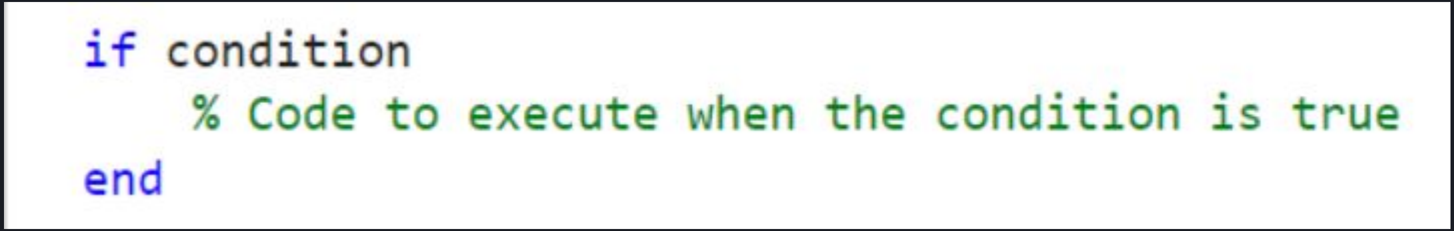


Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
    - **if condition:** the `if` statement is used for conditional execution. It allows you to perform different actions based on whether a specified condition is true or false. The basic structure of an `if` statement looks like this:

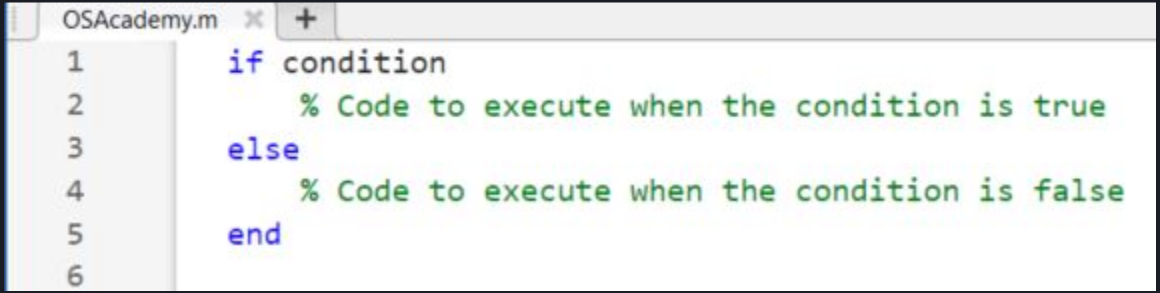


```
if condition
    % Code to execute when the condition is true
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - if-else condition: If the condition is true , a group of commands are executed, otherwise another group of commands are executed then the rest of the program is completed normally.



OSAcademy.m

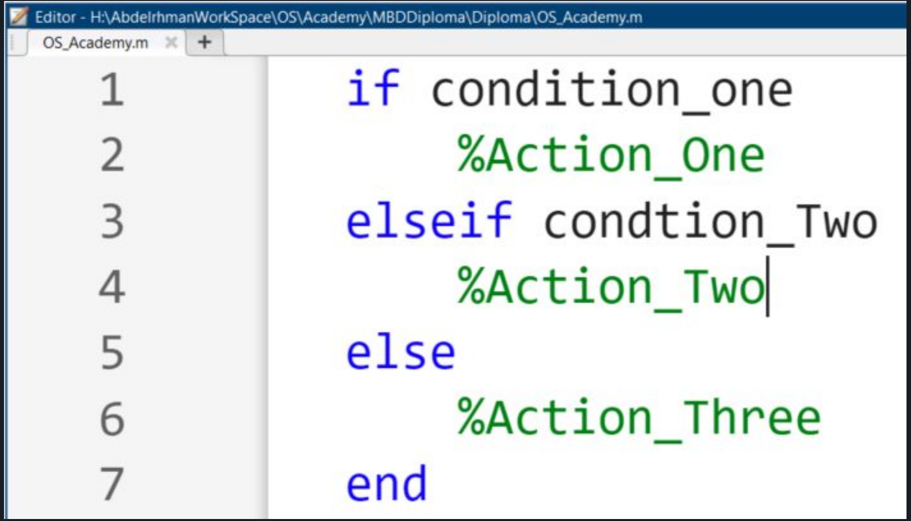
```
if condition
    % Code to execute when the condition is true
else
    % Code to execute when the condition is false
end
```

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - if-elseif-else condition


Editor - H:\AbdelrhmanWorkSpace\OS\Academy\MBDDiploma\Diploma\OS_Academy.m
OS_Academy.m

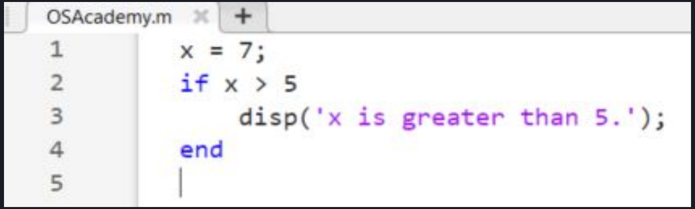
```
if condition_one
    %Action_One
elseif condtion_Two
    %Action_Two
else
    %Action_Three
end
```

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

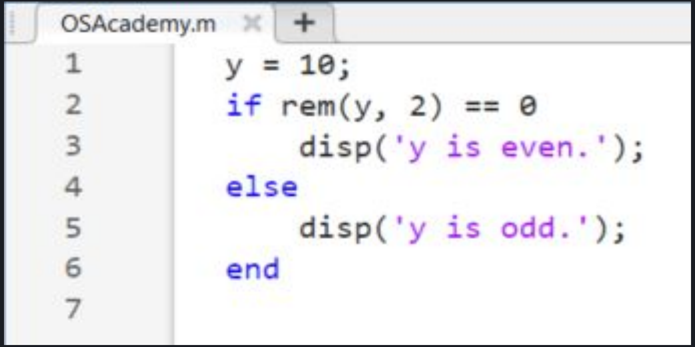
Programming With Matlab

- Control flow
  - if condition examples



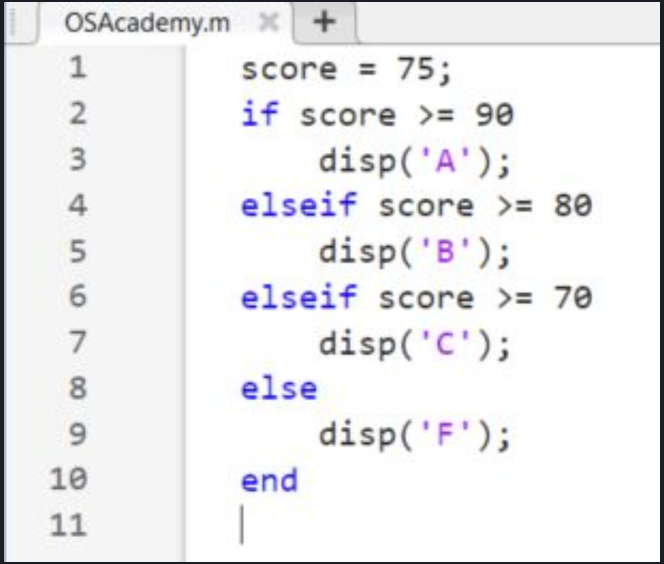
OSAcademy.m

```
x = 7;
if x > 5
    disp('x is greater than 5.');
end
```



OSAcademy.m

```
y = 10;
if rem(y, 2) == 0
    disp('y is even.');
else
    disp('y is odd.');
end
```



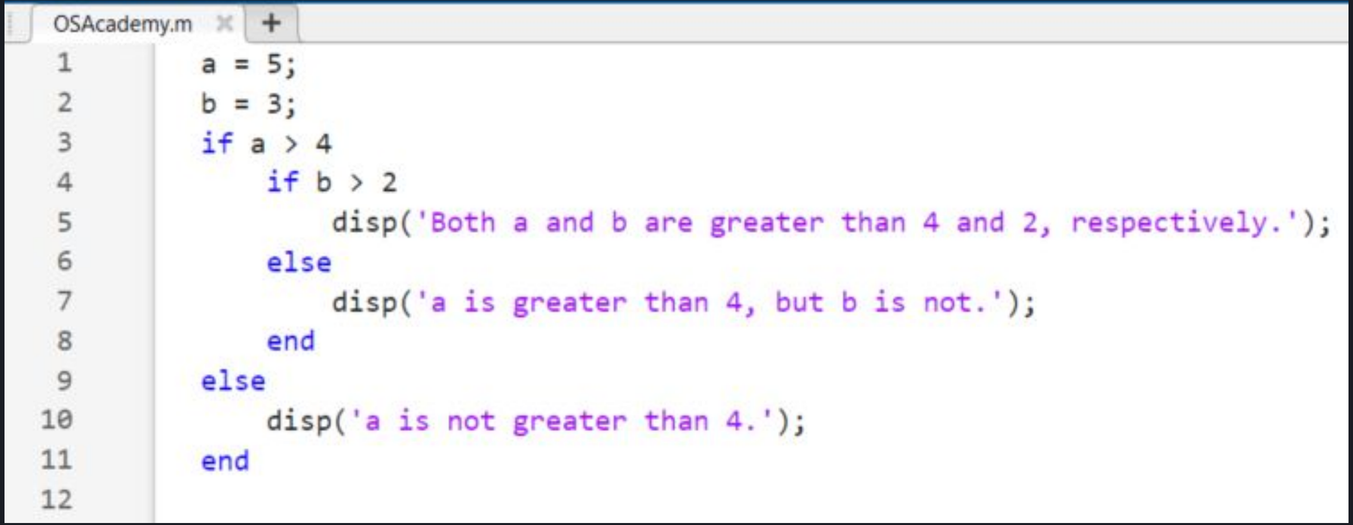
OSAcademy.m

```
score = 75;
if score >= 90
    disp('A');
elseif score >= 80
    disp('B');
elseif score >= 70
    disp('C');
else
    disp('F');
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - if condition examples



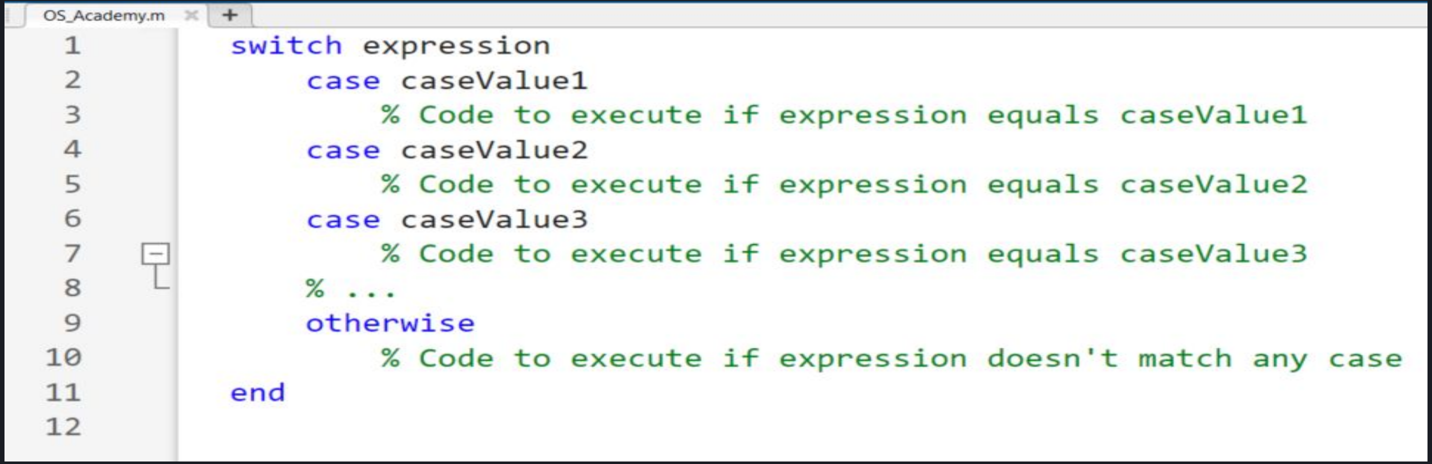
OSAcademy.m

```matlab
a = 5;
b = 3;
if a > 4
    if b > 2
        disp('Both a and b are greater than 4 and 2, respectively.');
    else
        disp('a is greater than 4, but b is not.');
    end
else
    disp('a is not greater than 4.');
end

```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - **switch case:** the `switch` statement is used to select one of many code blocks to be executed based on the value of a specific expression. It's an alternative to using a series of `if` and `else` statements when you have multiple conditions to check. The syntax of a `switch` statement is as follows

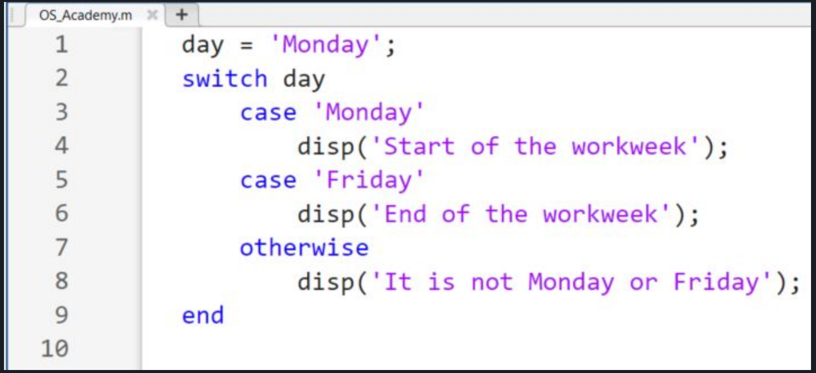


```matlab
switch expression
    case caseValue1
        % Code to execute if expression equals caseValue1
    case caseValue2
        % Code to execute if expression equals caseValue2
    case caseValue3
        % Code to execute if expression equals caseValue3
    % ...
    otherwise
        % Code to execute if expression doesn't match any case
end
```

Develop Programming skills and Proficiency in MATLAB

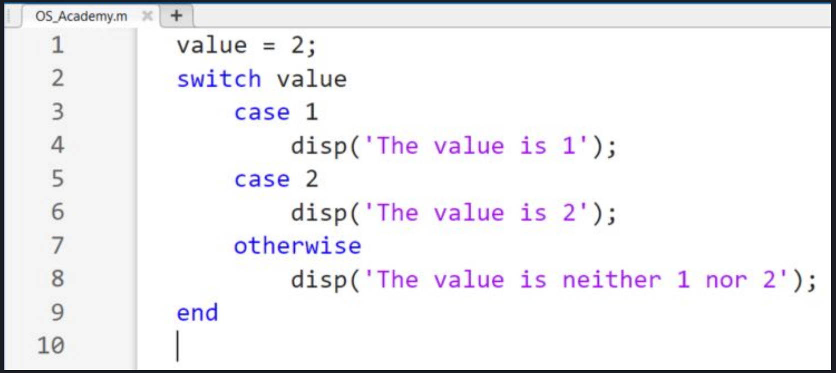
Programming With Matlab

- Control flow
  - switch case examples



OS_Academy.m

```
day = 'Monday';
switch day
    case 'Monday'
        disp('Start of the workweek');
    case 'Friday'
        disp('End of the workweek');
    otherwise
        disp('It is not Monday or Friday');
end
```



OS_Academy.m

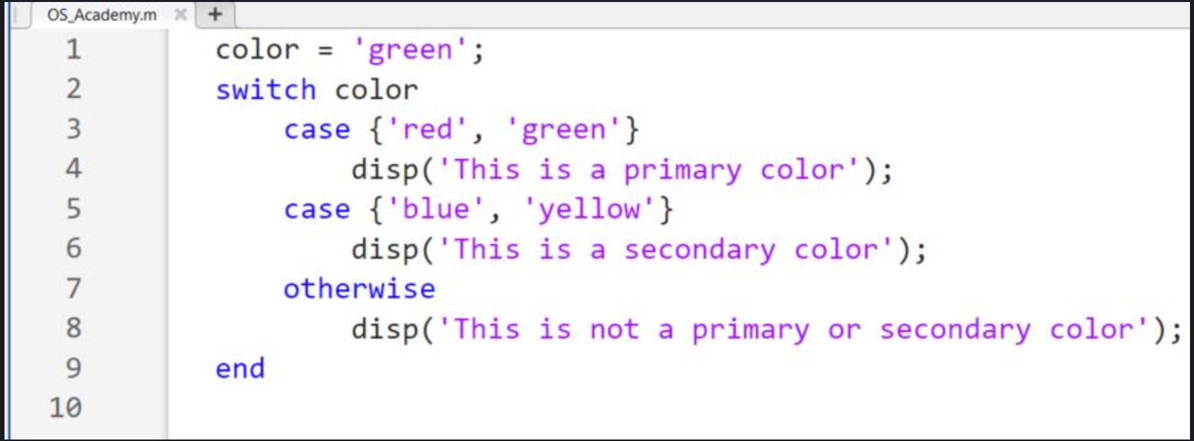
```
value = 2;
switch value
    case 1
        disp('The value is 1');
    case 2
        disp('The value is 2');
    otherwise
        disp('The value is neither 1 nor 2');
end
```

OS-Academy

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - switch case examples

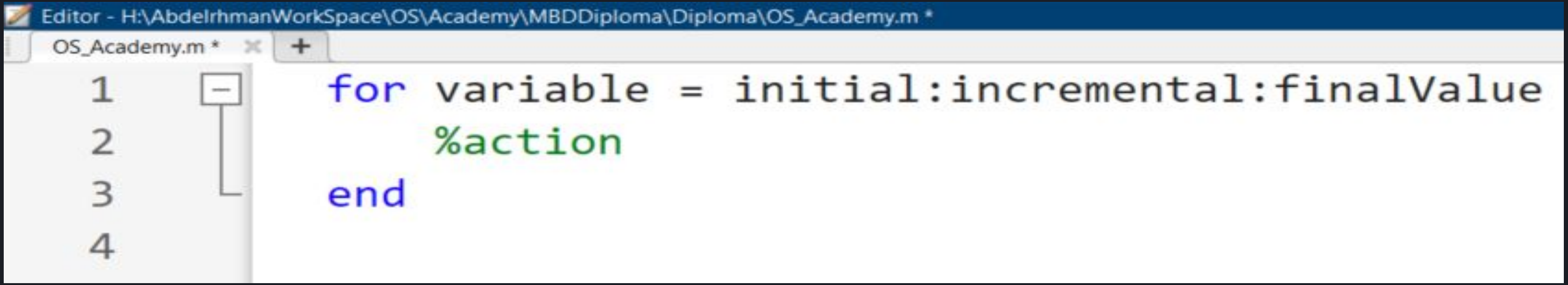


```
color = 'green';
switch color
    case {'red', 'green'}
        disp('This is a primary color');
    case {'blue', 'yellow'}
        disp('This is a secondary color');
    otherwise
        disp('This is not a primary or secondary color');
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - **for loop:** a "for loop" is a control structure that allows you to repeatedly execute a block of code a specified number of times or over a range of values. It's particularly useful for performing repetitive tasks, iterating through arrays or matrices, and automating calculations.

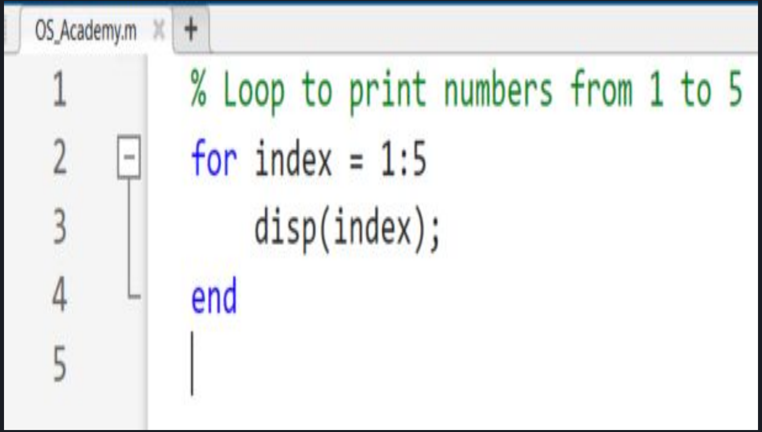

Editor - H:\AbdelrhmanWorkSpace\OS\Academy\MBDDiploma\Diploma\OS_Academy.m *
OS_Academy.m *

```
for variable = initial:incremental:finalValue
    %action
end
```

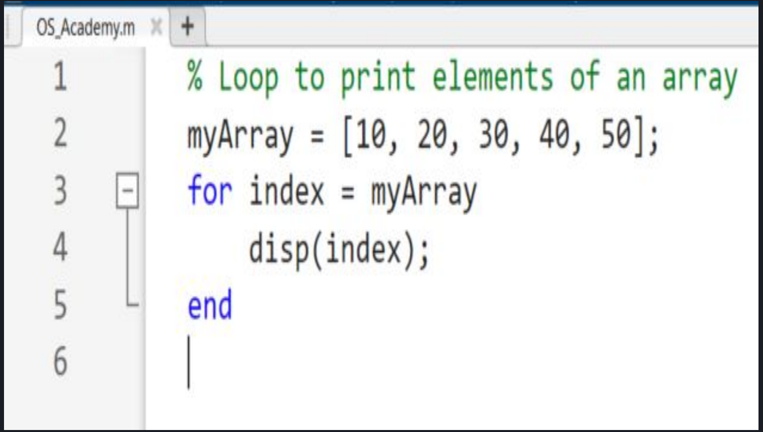
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - for loop examples



```matlab
% Loop to print numbers from 1 to 5
for index = 1:5
    disp(index);
end
```

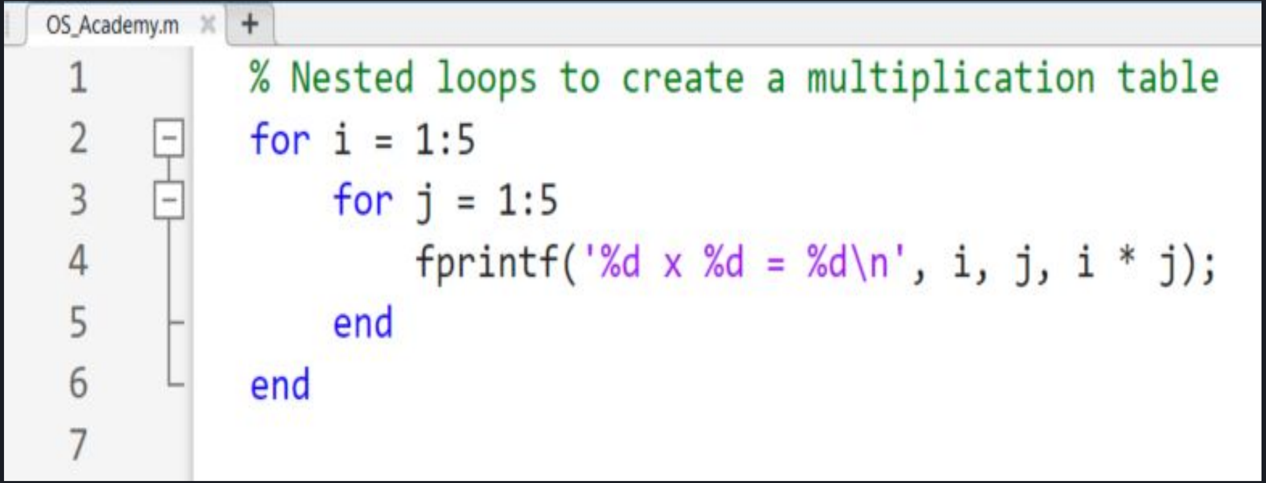


```matlab
% Loop to print elements of an array
myArray = [10, 20, 30, 40, 50];
for index = myArray
    disp(index);
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - for loop examples

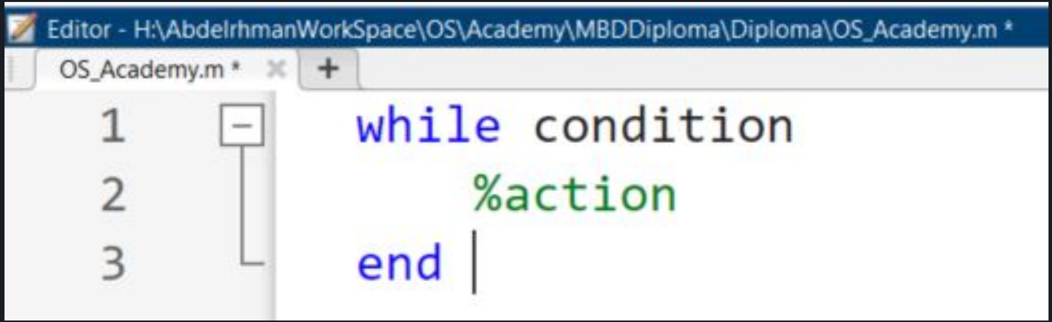


```
% Nested loops to create a multiplication table
for i = 1:5
    for j = 1:5
        fprintf('%d x %d = %d\n', i, j, i * j);
    end
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - **while loop:** a "while loop" is a control structure that allows you to repeatedly execute a block of code as long as a specified condition is true. It's used when you want to continue executing a piece of code until a certain condition is no longer met.

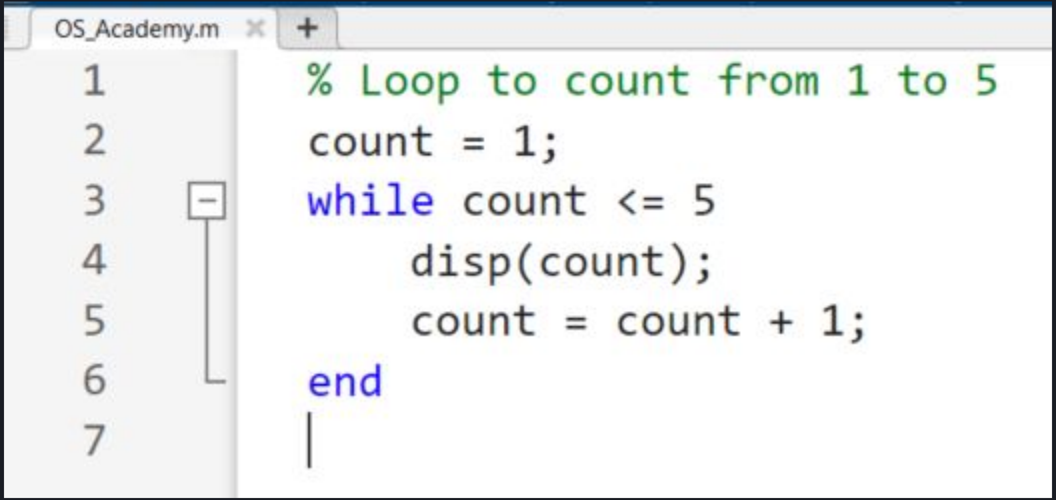

Editor - H:\AbdelrhmanWorkSpace\OS\Academy\MBDDiploma\Diploma\OS_Academy.m *
OS_Academy.m *

```
while condition
    %action
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - while loop example

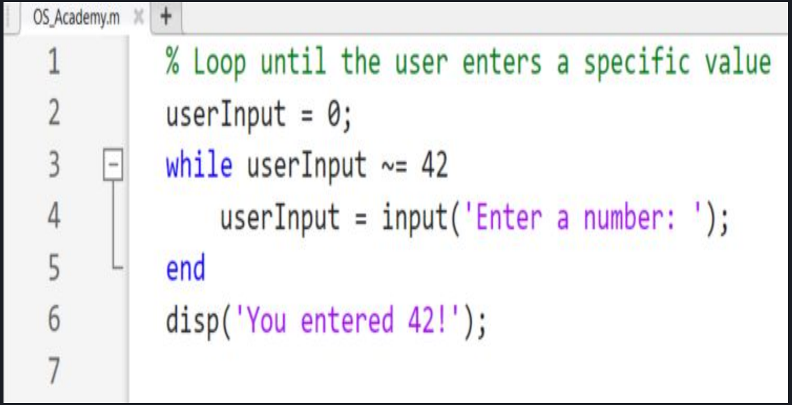


```matlab
% Loop to count from 1 to 5
count = 1;
while count <= 5
    disp(count);
    count = count + 1;
end
```

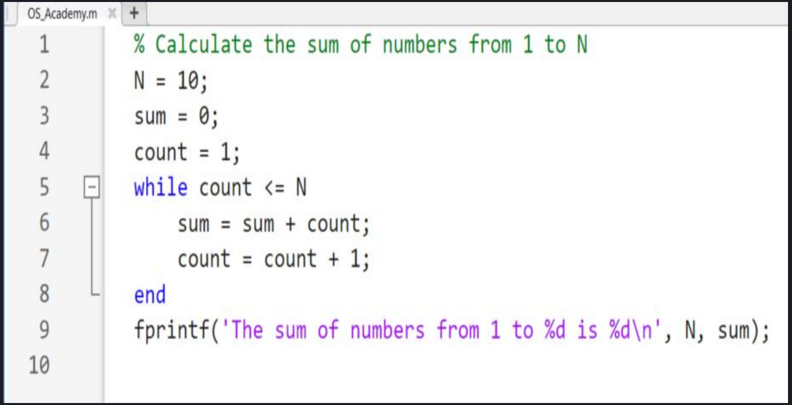
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - while loop example



```matlab
% Loop until the user enters a specific value
userInput = 0;
while userInput ~= 42
    userInput = input('Enter a number: ');
end
disp('You entered 42!');
```

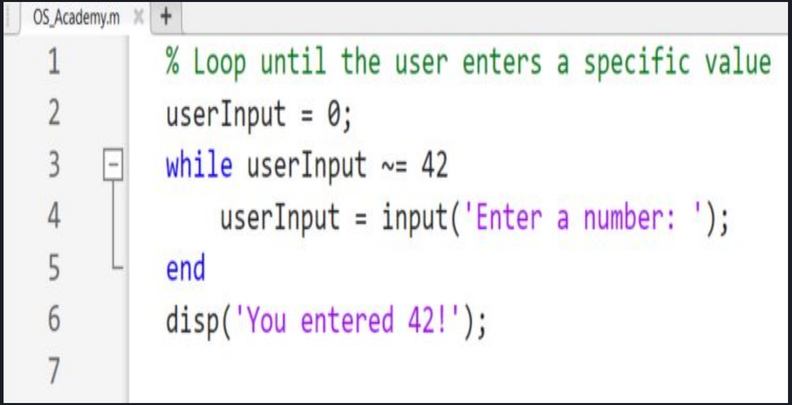


```matlab
% Calculate the sum of numbers from 1 to N
N = 10;
sum = 0;
count = 1;
while count <= N
    sum = sum + count;
    count = count + 1;
end
fprintf('The sum of numbers from 1 to %d is %d\n', N, sum);
```

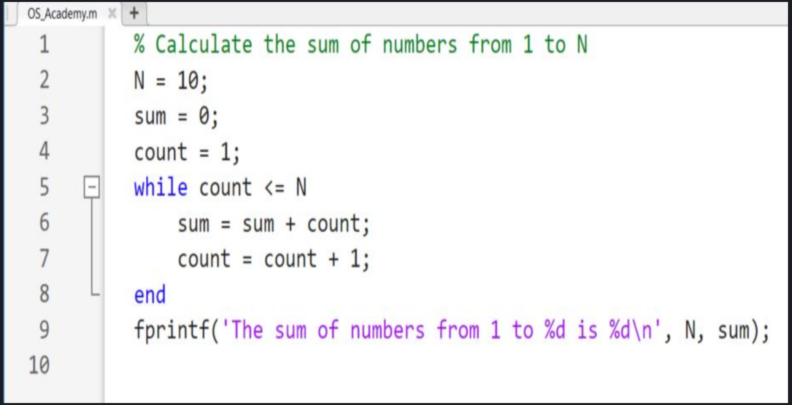
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - while loop example



```matlab
% Loop until the user enters a specific value
userInput = 0;
while userInput ~= 42
    userInput = input('Enter a number: ');
end
disp('You entered 42!');
```

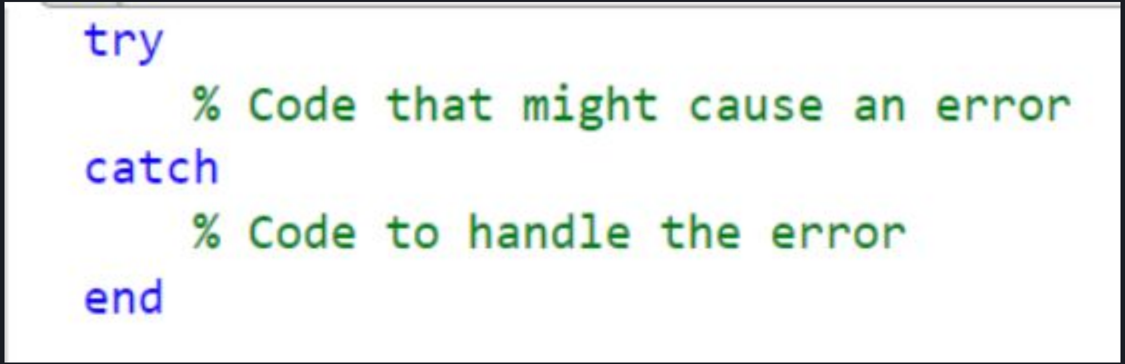


```matlab
% Calculate the sum of numbers from 1 to N
N = 10;
sum = 0;
count = 1;
while count <= N
    sum = sum + count;
    count = count + 1;
end
fprintf('The sum of numbers from 1 to %d is %d\n', N, sum);
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
  - **try and catch** :the `try` statement is part of the error-handling mechanism. It allows you to create a block of code where you anticipate errors and specify how MATLAB should respond if those errors occur. The `try` block is followed by one or more `catch` blocks, which contain code to handle specific types of errors.



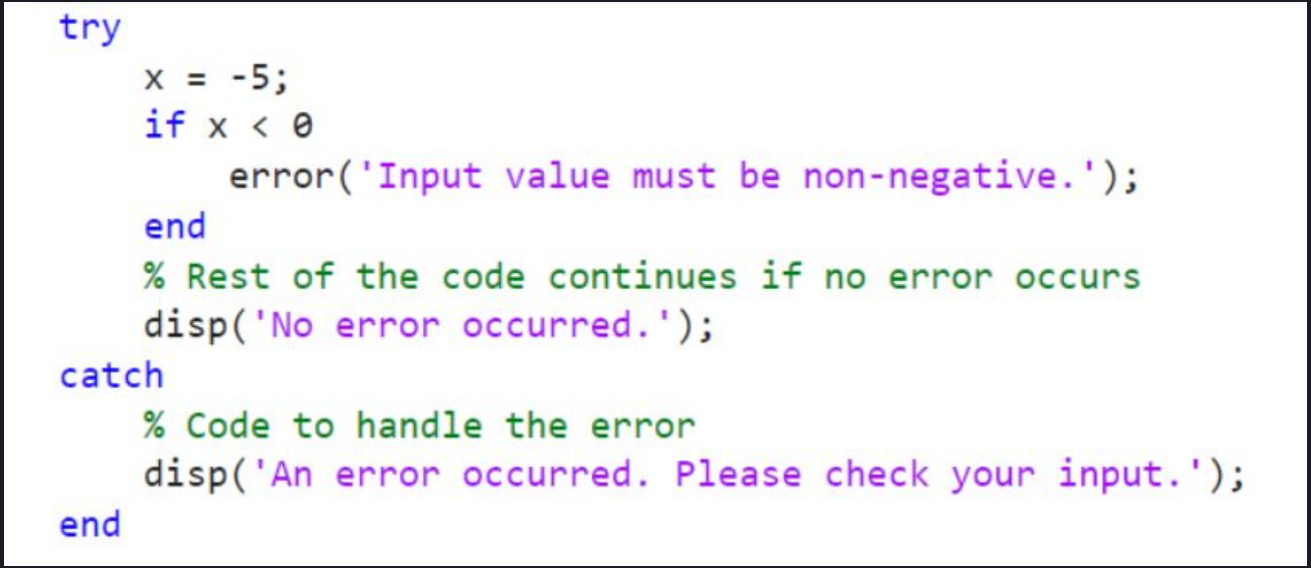
```
try
    % Code that might cause an error
catch
    % Code to handle the error
end
```

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Control flow**
    - try and catch examples

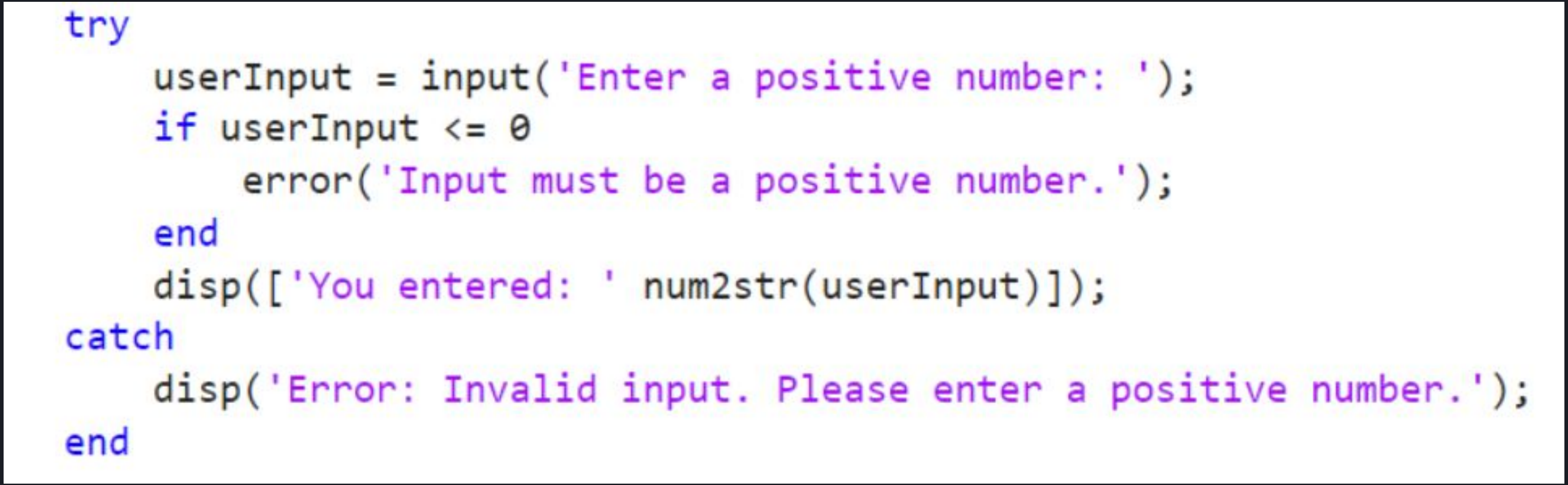


```
try
    x = -5;
    if x < 0
        error('Input value must be non-negative.');
    end
    % Rest of the code continues if no error occurs
    disp('No error occurred.');
catch
    % Code to handle the error
    disp('An error occurred. Please check your input.');
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - try and catch examples

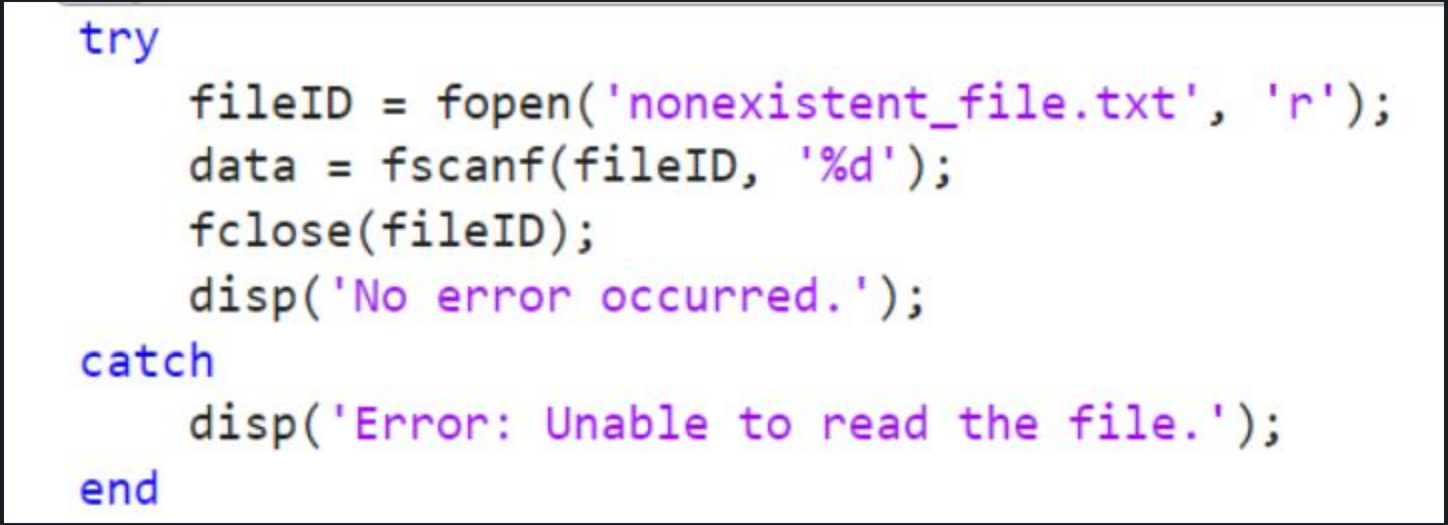


```
try
    userInput = input('Enter a positive number: ');
    if userInput <= 0
        error('Input must be a positive number.');
    end
    disp(['You entered: ' num2str(userInput)]);
catch
    disp('Error: Invalid input. Please enter a positive number.');
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - try and catch examples

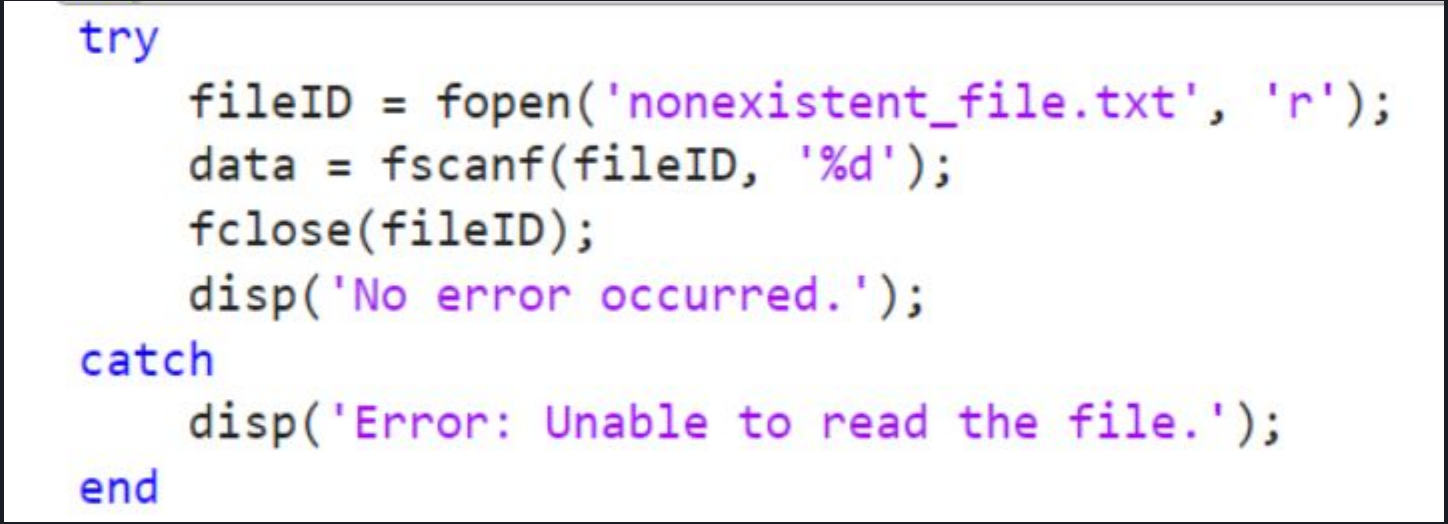


```
try
    fileID = fopen('nonexistent_file.txt', 'r');
    data = fscanf(fileID, '%d');
    fclose(fileID);
    disp('No error occurred.');
catch
    disp('Error: Unable to read the file.');
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Control flow
  - try and catch examples



```
try
    fileID = fopen('nonexistent_file.txt', 'r');
    data = fscanf(fileID, '%d');
    fclose(fileID);
    disp('No error occurred.');
catch
    disp('Error: Unable to read the file.');
end
```

Develop Programming skills and Proficiency in MATLAB

Lab 1: Click Here To Start

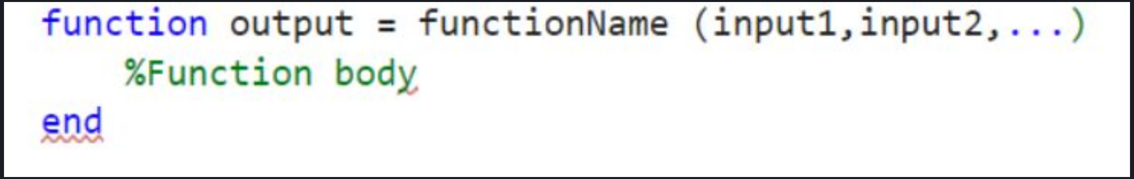
FOS-Academy

Model-Based Development Program

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Dealing with MATLAB functions**
  - **Function:** a function is a reusable block of code that performs a specific task. Functions are essential for code organization, modularity, and making your code more readable. They take input arguments, process them, and return output values.



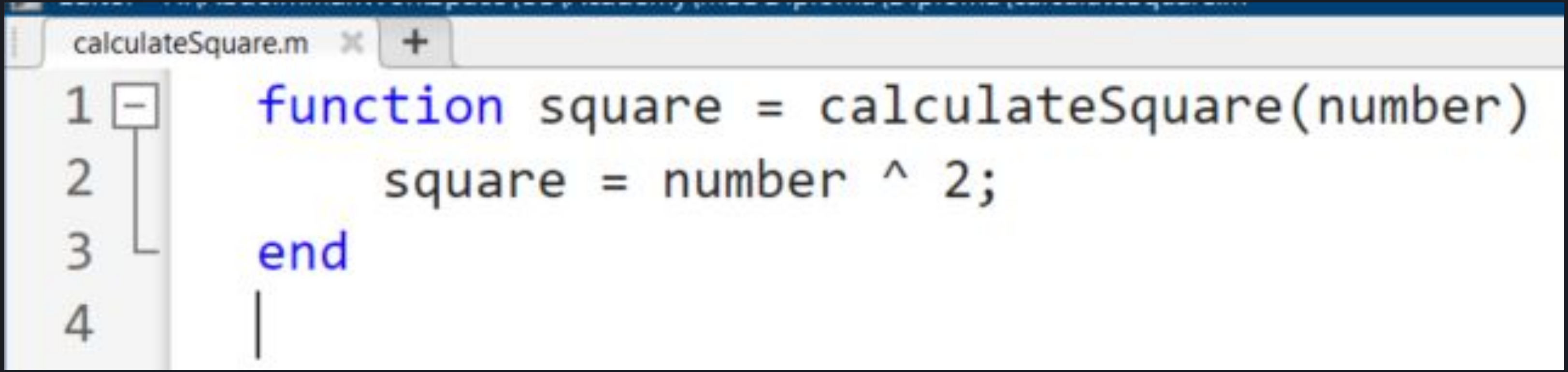
```
function output = functionName (input1,input2,...)
    %Function body
end
```

- `function` is the keyword used to declare a function.
- `output` is the variable that stores the result of the function.
- `functionName` is the name you choose for your function.
- `input1`, `input2`, etc., are the input arguments the function receives.
- `%` denotes comments in MATLAB.

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Dealing with MATLAB functions
  - Function examples

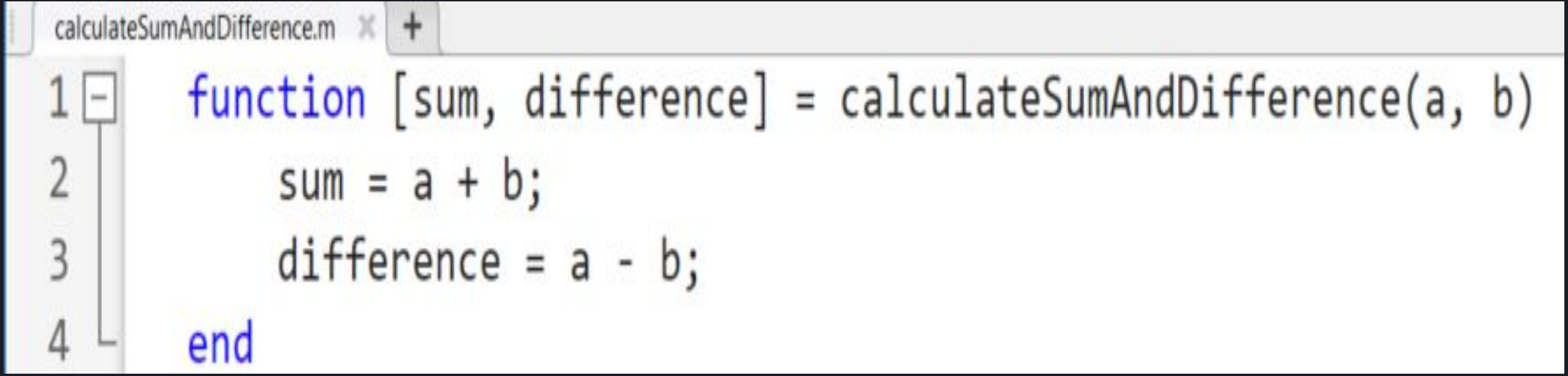

calculateSquare.m

```
function square = calculateSquare(number)
    square = number ^ 2;
end

```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- Dealing with MATLAB functions
  - Function examples



calculateSumAndDifference.m

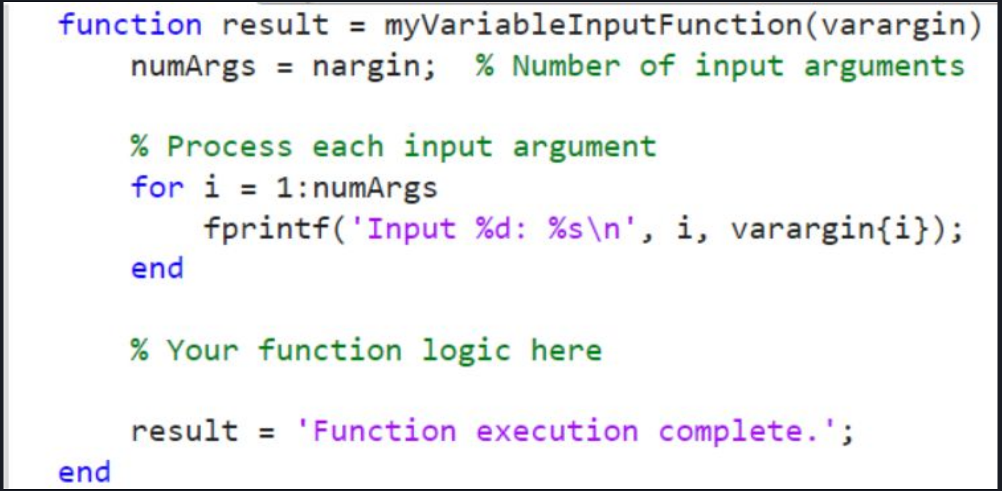
```
function [sum, difference] = calculateSumAndDifference(a, b)
    sum = a + b;
    difference = a - b;
end
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Dealing with MATLAB functions**
  - **Function examples :**you can create functions with a variable number of input or output arguments using varargin (variable input arguments) and varargout (variable output arguments). This allows your function to accept or return different numbers of arguments.

To create a function with a variable number of input arguments, use varargin. Inside your function, you can access the variable inputs using varargin, which is a cell array.

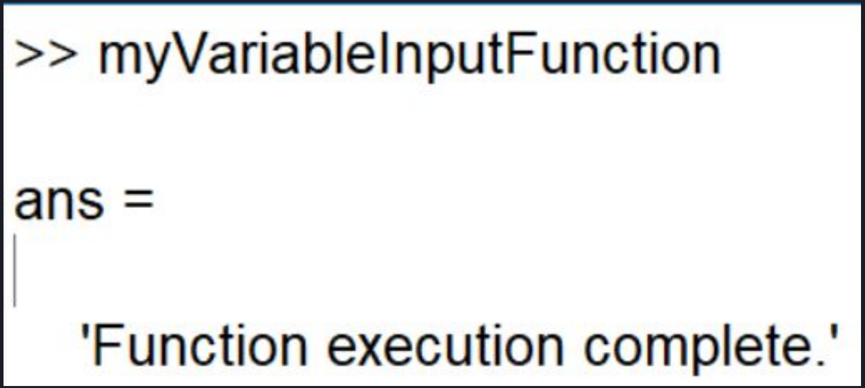


```
function result = myVariableInputFunction(varargin)
    numArgs = nargin;  % Number of input arguments

    % Process each input argument
    for i = 1:numArgs
        fprintf('Input %d: %s\n', i, varargin{i});
    end

    % Your function logic here

    result = 'Function execution complete.';
end
```

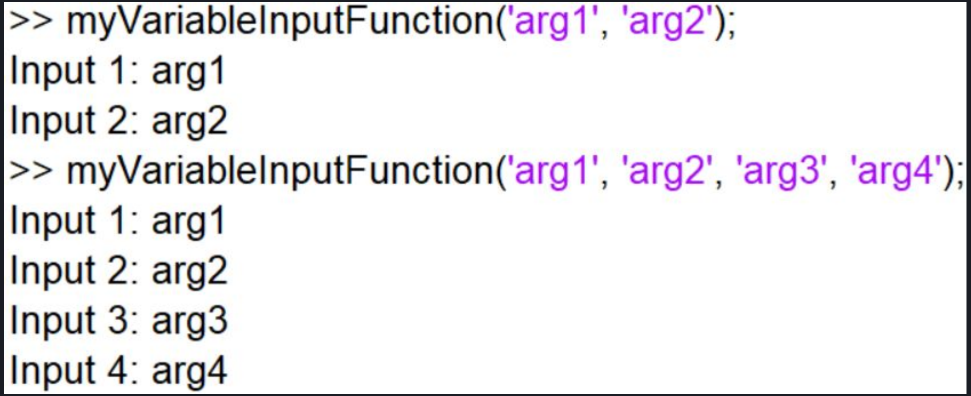
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Dealing with MATLAB functions**
  - Function examples



```
>> myVariableInputFunction

ans =

    'Function execution complete.'
```

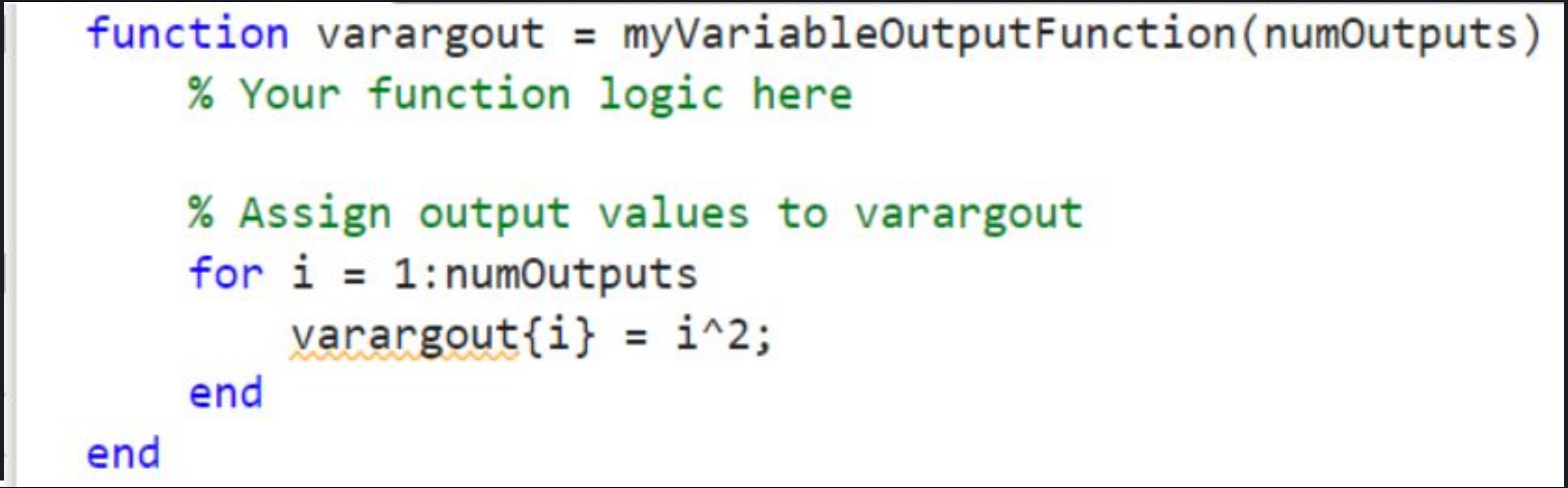


```
>> myVariableInputFunction('arg1', 'arg2');
Input 1: arg1
Input 2: arg2
>> myVariableInputFunction('arg1', 'arg2', 'arg3', 'arg4');
Input 1: arg1
Input 2: arg2
Input 3: arg3
Input 4: arg4
```

Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Dealing with MATLAB functions**
  - **Function examples :**To create a function with a variable number of output arguments, use varargout. Inside your function, you can assign values to varargout, which is a cell array.



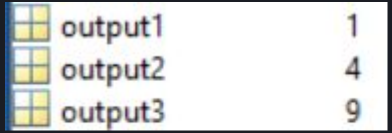
```
function varargout = myVariableOutputFunction(numOutputs)
    % Your function logic here

    % Assign output values to varargout
    for i = 1:numOutputs
        varargout{i} = i^2;
    end
end
```

Model-Based Development Program

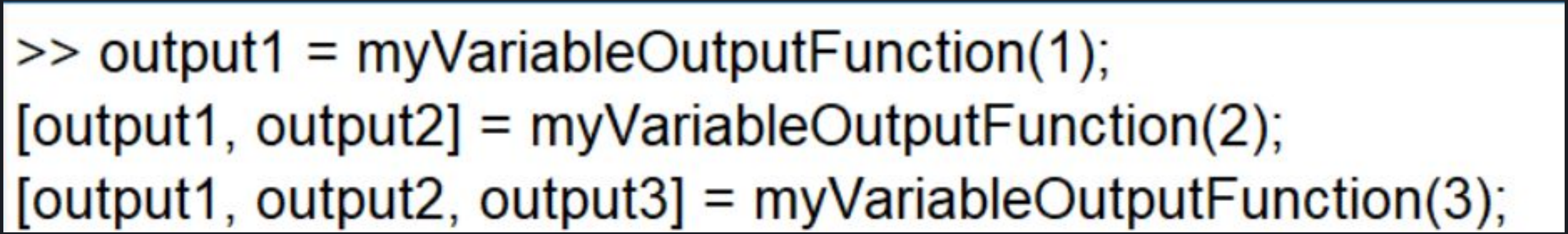
Develop Programming skills and Proficiency in MATLAB

Programming With Matlab

- **Dealing with MATLAB functions**
  - **Function examples :** To create a function with a variable number of output arguments, use varargout. Inside your function, you can assign values to varargout, which is a cell array.



```
output1    1
output2    4
output3    9
```



```
>> output1 = myVariableOutputFunction(1);
[output1, output2] = myVariableOutputFunction(2);
[output1, output2, output3] = myVariableOutputFunction(3);
```

Develop Programming skills and Proficiency in MATLAB

Lab 2: Click Here To Start

Develop Programming skills and Proficiency in MATLAB

Thank You!

OS-Academy

Model-Based Development Program