MONASH University

MONASH
ENGINEERING
ENG1060

# DOCUMENTATION

Presented by Tony Vo

Slides by Tony Vo

GROUP OF EIGHT AUSTRALIA

# M-FILE READABILITY

- Is this m-file easy to read?

```
clear all; close all; clc;
x=-pi:0.01:pi;
n=0;
sin_n0 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
n=1;
sin_n1 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
sin_approx = sin_n0 + sin_n1;
MATLAB_sin = sin(x);
plot(x,sin_approx,'b','linewidth',5)
hold on
plot(x,MATLAB_sin,'r','linewidth',5)
xlabel('x (radians)')
ylabel('sin(x)')
legend('sin\_approx','MATLAB\_sin')
```

# COMMENTS

- Comments are represented as green text in MATLAB
  - Declared with a percentage symbol (%)
  - Ignored by MATLAB
  - Used to provide additional information

Editor - C:\Users\Tony\Desktop\ENG1060 lectures\lecture 4\sin_approximation.m

+4 | lab3t5b_sols.m | lab3t5c_sols.m | plotting_mesh.m | sin_approximation.m

```
%Created by: Tony Vo
%class example
clear all; close all; clc;

%create a vector of x values
x=-pi:0.01:pi;

%calculating the first two terms
n=0;
sin_n0 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
n=1;
sin_n1 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
```

# COMMENTS

- Comments are used to document code
  - Documentation conveys the purpose of the code block
- For complicated programs, comments can be just as important as the code itself
  - Especially when working within a team
  - Or when being graded by others
- Code tells you HOW and comments tell you WHY

# WRITING COMMENTS

- Commenting rules
  - Comment your code as you write it
  - Use plain English
  - Keep your comments concise
  - Maintain the same style throughout

- Avoid redundant comments

```
x = y + 1          % x is y plus one
temperature = 30   % the temperature is set to 30
```

# USEFUL COMMENTS

- Include the following comments in your m-files
  - Your name and student ID number
  - The date you created or last modified the m-file
  - A short description of what the m-file does
  - clear all; close all; clc; (if appropriate)

- Remember to use meaningful variable names
  - This makes your code self documenting

- You do not need to comment everything
  - Add comments where you think it will be useful

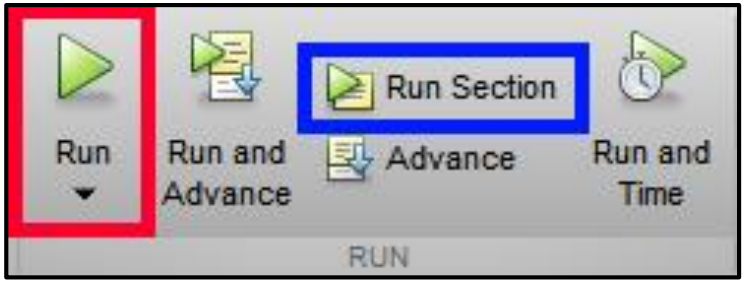
# COMMENTS: THE DIFFERENCE

```
% Written by: Tony Vo, ID: 12345678
% Last modified: 01/07/2015
% Compares several terms from the sine
% Taylor series against MATLAB's sine
% function

% clearing variables and screen, and closing windows
clear all; close all; clc;

% creating a vector of x values
x=-pi:0.01:pi;

% calculating the first two terms
n=0;
sin_n0 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
n=1;
sin_n1 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);

% adding the terms to get the Sine approximation
sin_approx = sin_n0 + sin_n1;
MATLAB_sin = sin(x);
```

```
clear all; close all; clc;
x=-pi:0.01:pi;
n=0;
sin_n0 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
n=1;
sin_n1 = (-1)^n/factorial(2*n+1)*x.^(2*n+1);
sin_approx = sin_n0 + sin_n1;
MATLAB_sin = sin(x);
```

# SECTIONING

- To create a section, use double %% symbols
  - Creates and highlights blocks of code
- The "Run" button runs through the entire script
- The "Run Section" button runs through sections of a script



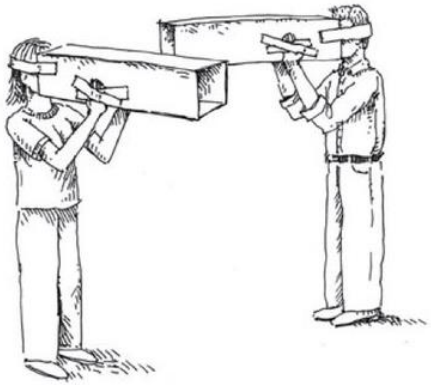
# GOOD PROGRAMMING PRACTICES

- Programming "style" is important
  - Would you write a report with no paragraphs?
  - Indenting helps with structuring
- Good programming practices ensures that your code is
  - Easy to understand
  - Meaningful to others
- Good codes are well documented and readily usable

# GOOD PROGRAMMING PRACTICES

- Unfortunately, engineers are sometimes too focused on the end results and don't follow good programming practices

- You will be graded on good programming practices



# SUMMARY

- Creating comments
- Importance of code documentation
- Code readability
- Good programming practices

- When would indenting code be useful?