MONASH University

MONASH
ENGINEERING
ENG1060

FUNCTIONS, COMMENTING, DEBUGGING AND STRINGS

Edited and Presented by Soon Foo Chong (Joseph)

Slides by Tony Vo

Assisted by Tham Lai Kuan

GROUP OF EIGHT AUSTRALIA

HOUSEKEEPING

MONASH University

- Forums
  - Be aware that there is a News Forum for important announcements
  - You can post any unit-related in the discussion forums.

- Peer Assisted Study Sessions (PASS)
  - Begins in Week 3.

Monday: 3:30-5:30pm (MYT) / 6:30-8:30pm (AEDT) Meeting ID: **891 2853 2133** Password: **941880** https://monash.zoom.us/j/89128532133?pwd=VVVOenhDbW5xZ3h6ZFRZR1dieVhldz09	Tuesday: 12-2pm (MYT) / 3-6pm (AEDT) Meeting ID: **852 2658 1851** Password: **933340** https://monash.zoom.us/j/85226581851?pwd=d0YxeWVHd0tudnplanFRYWU2ZGJRUT09

LABS

- Team activities
  - It *should* be a content refresh
  - Opportunity to teach students who have not prepared or do not understand
  - Team task concepts are applied in the individual tasks

- Individual tasks
  - Preparation is required, especially if you struggled to finish the lab 1
  - Late attendance will result in the forfeit of team task marks
  - Consolidate your skills (supp. questions and assignment, exam-type questions, or help table members)
  - Marks uploaded to Moodle after the deadline

IN THIS WORKSHOP

1. Creating and calling user-defined functions
2. Creating variables through user prompts
3. Providing adequate documentation and comments

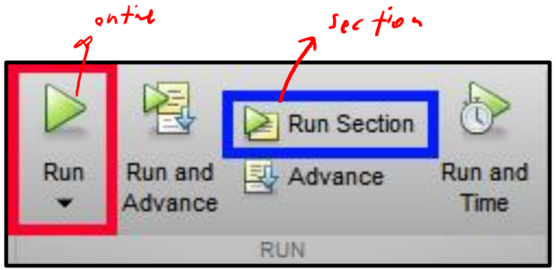
RECAP: COMMENTING

- Code tells you HOW and comments tell you WHY

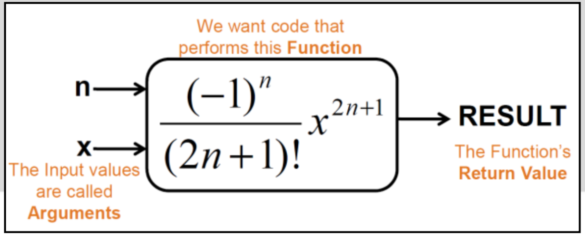
- For complicated programs, comments can be just as important as the code
  - Your name, ID, and date
  - Description of the code
  - clear all; close all; clc;   (if appropriate)

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

- Section/highlight your code using %% symbols
  - <u>Run</u> button runs through the entire script
  - <u>Run Section</u> button runs through sections of a script



RECAP: FUNCTIONS



- Functions are
  - Modular: reuse a pattern of code on different input values
  - Black boxes: details are not important when they're functions are used
  - Inbuilt: sin(), sqrt(), linspace(), plot(), max(), log(), etc.

- The purpose of a function file is to determine and provide its outputs!

- Function declaration: function outputs = function_name(inputs)
  - Multiple inputs: function taylor_result = SineTaylor(x,n)
  - Multiple outputs: function [drop, drop_velocity] = DebrisDrop(conveyer_height)

USER-DEFINED FUNCTIONS

- In function files:
  - Do not use fprintf or disp
  - Do not plot
  - Do not ask for an input prompt
  - Do not use clear all; close all; clc; commands
  - Do not overwrite existing MATLAB functions
  - Do suppress everything!
  - Do document your code appropriately
- Why can't you click "run" on a function file?
  - You must always have a complementary m-file to call your function

RECAP: FUNCTION DOCUMENTATION

- Include the following information AFTER the function declaration. This will show up after typing *help <function name>* in the command window
  - Function declaration without the word "function"
  - Name, ID and date and description of what the function does
  - Description of the input argument and outputs

SineTaylor.m

```
function taylor_result = SineTaylor(x,n)
% taylor_result = SineTaylor(x,n)
% Written by Tony Vo
% Created on 12/07/2015
%
% Input Arguments
% ----
% x - input values for Taylor term (can be a vector)
% n - n value for Taylor term (must be a scalar)
% Outputs
% ----
% taylor_result - taylor series term values (can be vector)
taylor_result = (-1)^n / factorial(2*n + 1) * x.^(2*n + 1);
```

POOR PROGRAMMING PRACTICES (PPP)

- PPP will be deducted beginning lab 3 (week 4)

- Each PPP item = -0.5 marks (max -2 marks)
  These include but not limited to:
  - Missing or insufficient commenting/documentation in m-files/function files
  - Inefficient coding (e.g. hard coding or copy-paste code instead of loops)
  - Unnecessary outputs (e.g. large matrices)
  - No axis labelling for plots

RECAP: MORE BUILT-IN FUNCTIONS

- A = input('text'): Prints out the text string and awaits an input from the user to store into variable A
  - Only use this if the question asks to prompt the user

- pause: Pauses the m-file until a key is pressed
- pause(N): Pauses for N seconds

- disp(X): Displays the value of variable X
- disp('text'): Displays the text string

fprintf

[15 MINS]

ACTIVITY: INFINITE SERIES

INFINITE_SERIES_TEMPLATE.M

Equations:

$$error = \left|\frac{approx - true}{true}\right| \times 100\%$$

When Ms ENG1060 is bored, she likes to write out converging infinite series. She is trying to show that

LHS RHS

$$\underbrace{\sum_{n=1}^{\infty} \frac{(9/10)^n}{n}}_{\text{LHS}} \approx \sum_{n=1}^{n_{\max} \approx \infty} \frac{(9/10)^n}{n} = \underbrace{\log_e(10)}_{\text{RHS}}$$

n_{max}

$$\text{n all} = [1 \quad 2 \quad 3 \quad \dots \quad 100]$$

$$\text{terms} = [0.9 \quad 0.405 \quad 0.243 \quad \dots \quad 2.656 \times 10^{-7}]$$

$$\textbf{sum of } \boldsymbol{terms} = [\mathbf{0.9} \quad \mathbf{1.305} \quad \mathbf{1.548} \quad \dots \quad \mathbf{2.303}]$$

If $n_{\max} = 2$

sum of terms = $0.9 + 0.405 = \mathbf{1.305}$

If $n_{\max} = 3$

sum of terms = $0.9 + 0.405 + 0.243 = \mathbf{1.548}$

MATLAB commands:

```
plot(…)
xlabel(…)
ylabel(…)
title(…)
hold
legend
figure
```

Activity involves:

1. Computing the sum of a vector
2. Plotting, labelling
3. Creating a new figure

[15 MINS]

ACTIVITY: INFINITE SERIES

INFINITE_SERIES_TEMPLATE.M

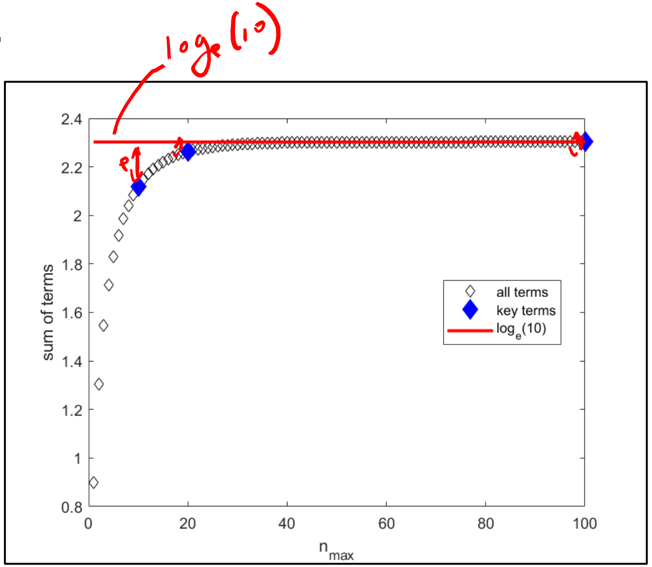
Equations:

$$\sum_{n=1}^{n_{max}} \frac{(9/10)^n}{n}$$

$$error = \left|\frac{approx - true}{true}\right| \times 100\%$$

1. Generate the terms for each n for $n = 1{:}100$
2. Compute the sums for $n_{max} = [10\ 20\ 100]$.
3. Plot and label: 'bd'
   a. The sums against the values of n_{max} as blue diamonds.
   b. $\log_e(10)$ as a red line
4. Calculate the % error between the sums and $\log_e 10$. Plot this against n_{max} in a new figure with logarithmic axes. Calculate the error for n=20

5. How would you do this for $n_{max} = 1{:}100$?



[20 MINS]

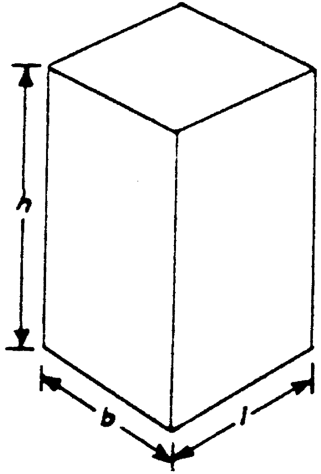
ACTIVITY: RECTANGULAR PRISM

REC_PRISM.M, REC_PRISM_CALLER.M

MATLAB commands:

```
function […] = …(…)
```

Ms ENG1060 is writing a program which can determine the surface area (SA) and volume (V) of several polygons. She starts off with a rectangular prism with dimensions h, b, and l. The units are in metres.



Activity involves:

1. Writing a function file with multiple inputs and outputs.
2. Writing adequate comments for a function file.
3. Calling a function within a script.
4. Changing the inputs and outputs to a function file.

[20 MINS]

ACTIVITY: RECTANGULAR PRISM

REC_PRISM.M, REC_PRISM_CALLER.M

MATLAB commands:

`function […] = …(…)`

1. Determine the function header declaration and complete the function file

2. Determine surface area and volume for $h = 1, b = 1, l = 1$.
   Check that the outputs are $SA = 6\ m^2$ and $V = 1\ m^3$

3. Which of these combinations give
   $SA = 502\ m^2$ and $V = 390\ m^3$?

h	b	l
4	9	13
7	11	4
9	6	7
15	13	2
7	7	7

4. Change your function so that it will provide a third output – *cost* to paint the entire surface of the prism, given a fourth input – *c*: paint cost per m^2.
   a. Calculate the cost to paint a prism with $h = 15, b = 13, l = 2, c = 2.5$

[15 MINS]

ACTIVITY: FOOD

FOOD.M, FOOD_CALLER.M

MATLAB commands:

`input(…)`

`function […] = …(…)`

There are 3 primary macronutrients.
Each Cal contains 4.186 kJ.

- Carbohydrates (C): 4 Cal/g
- Proteins (P): 4 Cal/g
- Fats (F): 9 Cal/g

Activity involves:

1. Writing a function file with multiple inputs and outputs.
2. Writing adequate comments for a function file.
3. Calling a function within a script.
4. Prompting for vector inputs and using it in a function.

[15 MINS]

ACTIVITY: FOOD

FOOD.M, FOOD_CALLER.M

MATLAB commands:

`input(…)`

`function […] = …(…)`

Create a function that accepts C, P and F (in grams) and outputs the total energy intake in Cal and kJ.

1. For the following meals, write a script to determine the energy intake in Cal and kJ:
   A. KFC's 9 for $9.95: C=65g, P=150g, F=120g
   B. Dozen KK donuts: F=121g, P=26g, C=285g

2. In the same script, prompt the user for the macronutrients in a 12-inch Smashed Falafel sandwich in vector form.
   a. Use matrix indexing to pass these as inputs to your function.
   b. How many Krispy Kreme donuts is one sandwich equivalent to?

Protein (g)	14.7
Fat (g)	13.6
Carbohydrates (g)	61.9

[10 MINS]

ACTIVITY: SUNFLOWER

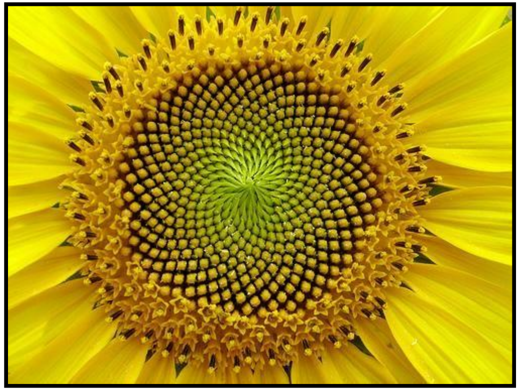
SUNFLOWER.M, SUNFLOWER_CALLER.M

MATLAB commands:

```
function […] = …(…)
subplot(…)
plot(…)
axis …
xlabel(…)
ylabel(…)
title(…)
```

Mr ENG1060 has an unhealthy obsession with sunflowers. He is in pursuit of discovering the perfect sunflower using the following model.

The arrangement of seeds in a flower such as a sunflower follows a fixed mathematical pattern. Mr ENG1060 is attempting to recreate this pattern using a mathematical model.



Activity involves:

1. Writing a function file with multiple inputs and outputs.
2. Writing adequate comments for a function file.
3. Calling a function within a script.
4. Subplots and labelling.

[10 MINS]

ACTIVITY: SUNFLOWER

SUNFLOWER.M, SUNFLOWER_CALLER.M

The n^{th} seed is at position $r = \sqrt{n}$ with angular coordinate $\pi dn/180$ radians, where d is the constant angle of divergence (in degrees) between two successive seeds.

1. Create a function which takes in the number of seeds and angle of divergence and outputs the (x, y) coordinates of the seeds. Hint: $x = rcos(\theta)$, $y = rsin(\theta)$.

2. Compute the coordinates for 1000 seeds with $d = 60°, 91.6°$, and $137.5°$

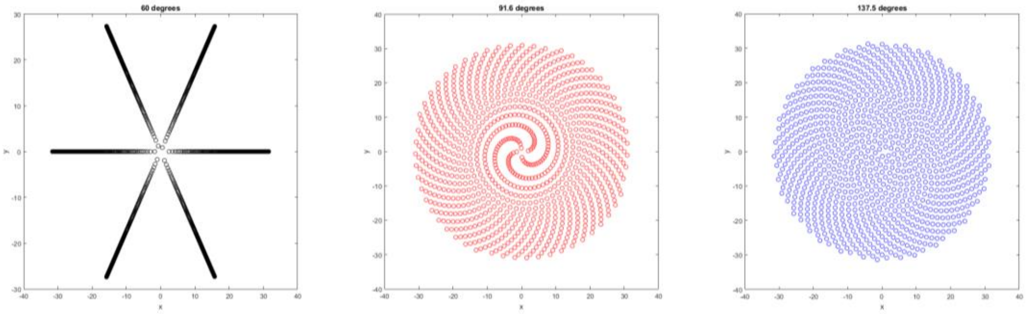
   What is the x-coordinate of the 888th seed if $d = 137.5°$?

3. Create a 1x3 subplot for each d
   Label the figure as appropriate.

   Left: 60°, black circles

   Middle: 91.6°, red circles

   Right: 137.5°, blue circles



IN THIS WORKSHOP

1. Creating and calling user-defined functions
2. Creating variables through user prompts
3. Providing adequate documentation and comments

PART A: MATLAB PROGRAMMING

1. ~~Introduction, variables and matrices~~
2. ~~Matrix calculations and plotting~~
3. ~~Functions, commenting, debugging and strings~~
4. Input, output and IF statements
5. Loops and debugging
6. Loops, advanced functions and MATLAB limitations

You can now complete lab 3!