Office Use Only

MONASH University

Semester One 2018 (DRAFT)
Examination Period

Faculty of Engineering

EXAM CODES: **ENG1060**

TITLE OF PAPER: **COMPUTING FOR ENGINEERS - PAPER 1**

EXAM DURATION: 3 hours writing time

READING TIME: 10 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT: (Tick where applicable)

☐ Caulfield ☑ Clayton ☐ Parkville ☐ Peninsula
☐ Monash Extension ☐ Off Campus Learning ☑ Malaysia ☐ Sth Africa
☐ Other (specify)

During an exam, you must not have in your possession any item/material that has not been authorised for your exam. This includes books, notes, paper, electronic device/s, mobile phone, smart watch/device, calculator, pencil case, or writing on any part of your body. Any authorised items are listed below. Items/materials on your desk, chair, in your clothing or otherwise on your person will be deemed to be in your possession.

No examination materials are to be removed from the room. This includes retaining, copying, memorising or noting down content of exam material for personal use or to share with any other person by any means following your exam.
Failure to comply with the above instructions, or attempting to cheat or cheating in an exam is a discipline offence under Part 7 of the Monash University (Council) Regulations.

<u>AUTHORISED MATERIALS</u>

OPEN BOOK ☐ **YES** ☑ **NO**

CALCULATORS ☑ **YES** ☐ **NO**
(Only calculators with an 'approved for use' Faculty sticker are permitted.)

SPECIFICALLY PERMITTED ITEMS ☐ **YES** ☑ **NO**
if yes, items permitted are:

Candidates must complete this section if required to write answers within this paper

STUDENT ID: __ __ __ __ __ __ __ __ DESK NUMBER: __ __ __ __ __

EXAM INSTRUCTIONS

- Complete the **Student ID** and **Desk Number** details on the cover page
- Write all **answers in the answer boxes**
- Write your answers with a pen
- **DO NOT** use a red pen or marker
- Blank sheets are provided at the back of the exam for workings. These workings will not be marked.
- You may detach the blank sheets and formula sheets from the back of the exam paper (last 3 sheets of paper).

EXAM OUTLINE

PART A (40 MARKS)

Attempt ALL Questions

PART B (60 MARKS)

Attempt ALL Questions

Blank sheets for workings (not marked)

MATLAB Information and FORMULAS

Office Use Only

A1 /7	A2 /6	A3 /6	A4 /8	A5 /6	A6 /7	B1 /15	B2 /15	B3 /15	B4 /15	TOTAL

PART A: ATTEMPT **ALL** QUESTIONS

Question A1 (7 marks)

Consider the following matrices:

$$A = \begin{bmatrix} 96 & 96 & 14 \\ 16 & 47 & 42 \\ 97 & 80 & 92 \end{bmatrix} \qquad B = \begin{bmatrix} 39 & 96 & 5 \\ 66 & 3 & 10 \\ 17 & 28 & 82 \end{bmatrix} \qquad C = \begin{bmatrix} 68 & 76 & 74 \end{bmatrix}$$

Where *A*, *B* and *C* are double types.

Note: If a MATLAB statement returns an error, write down "error".

(a) Provide the **syntax to create A, B and C**.

A = [96 96 14; 16 47 42; 97 80 92]; B = [39 96 5; 66 3 10; 17 28 82]; C = [68 76 74];	0.5 marks for 2 correct syntaxes 1 mark for 3 correct syntaxes

(b) Provide the output of **X = B-C**

Error	0.5 marks

(c) Provide the output of **[a,b] = size(C)**

a=1 b=3	0.5 marks

(d) Provide a single-line syntax to create the following matrix by **only addressing entire rows (not individual elements) of A, B and C**.

$$S = \begin{bmatrix} 97 & 80 & 92 \\ 66 & 3 & 10 \\ 68 & 76 & 74 \end{bmatrix}$$

C = [A(end,:); B(2,:); C]	1 mark

(e) Provide the output of **T = transpose(B)**

T =	39	66	17	1 mark
	96	3	28	
	5	10	82	

(f) Provide the output of **U = sum(A)**

U = 209 223 148 1 mark

(g) Provide the output of **V = find(A==B)**

V=4 1 mark

(h) Provide the syntax to add **a 4th column to B** which contains elements in the 1st column of B raised to the power of 3.

B(:,4) = B(:,1).^3 1 mark

Question A2 (6 marks)

Consider the following MATLAB function:

```
function [vr, reality] = helminth(sn,bt,dcp)

pre = sum([sn,bt,dcp]);
trans = sqrt(abs(bt – sn));
post = sn.*bt;
reality=0;
vr=0;

if  pre < 5
    reality = floor(pre);
elseif pre > 15
    reality = ceil(pre);
else reality = pre.^2;
      vr = post.^3;
end
```

Note: <u>If a MATLAB statement returns an error, write down "error"</u>.

(a) What are the input and output variables in the function declaration above?

Input variables = sn, bt, dcp	**0.5 marks**
Output variable = vr, reality	**0.5 marks**

(b) Provide the **name of the function** and the **extension format of the file**?

Filename = helminth	**0.5 marks**
Extension = .m	**0.5 marks**

(c) What is the output of the following command?

[vr, reality] = helminth(9,6,3)

vr = 0	**0.5 marks**
reality = 18	**0.5 marks**

(d) Consider **x = [9, 9]**, **y = [6, 6]**, and **z = [3, 3]**. What is the result of:
[a, b] = helminth(x, y, z)?

a = 0	0.5 marks
b = 36	0.5 marks

(e) Is it possible to convert the function provided at the start of this question to an anonymous function? If yes, provide the syntax. If no, write "error" and explain why.

Error/No	0.5 marks
It is not possible because there are multiple outputs.	0.5 marks

(f) MATLAB provides two warnings for the function provided at the start of this question. **Describe ONE of these warnings**.

The value assigned to 'trans' might be unused or	1 mark
The value assigned to 'reality' might be unused	

Question A3 (6 marks)

Consider the following matrix defined by **L = [-4:3:8; 4:-2:-4]** and **K = [1:5 ; 6:10]**, and logicals **A = 0 (false) and B = 1 (true).** Write the results of the MATLAB statement as specified in each question below.

Note: **<u>If a MATLAB statement returns an error, write down "error"</u>.**

(a) Provide the output of **A | B**

True or 1	0.5 marks

(b) Provide the output of **A & B**

False or 0	0.5 marks

(c) Provide the output of **F = L > 0**

F = 0 0 1 1 1
    1 1 0 0 0

1 mark

(d) Provide the output of **G = (~(L > 0) & (K < 4))**

G = 1 1 0 0 0
    0 0 0 0 0

1 mark

(e) Replace the □ **(square) symbol** in the following syntax **H = K □ L** so that it provides the logical result:

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Provide the complete expression below.

H = K<L	1 mark

(f) Provide a **single-line syntax** to create **a logical-type matrix** which contains **true elements when the equivalent elements of L are odd.**

logical(rem(L,2)) or **logical(mod(L,2))**	**0.5 marks for rem/mod** **0.5marks for logical conversion**

(g) Describe why **short-circuit operators** are used.

Short-circuit operators allow AND and OR operators to be stopped at the earliest point when a result can be determined/evaluated.	**1 mark**

Question A4 (8 marks)

Answer the following multiple-choice questions by writing the letter corresponding to the correct answer in the table provided below. Note: only one letter can be written in each box for each question. An example is provided below:

EXAMPLE: Which exam is this unit for?

A. ENG1001
B. ENG1002
C. ENG1003
D. ENG1005
E. ENG1060

QE:	E
Q1:	B
Q2:	D
Q3:	C
Q4:	B

Q5:	C
Q6:	B
Q7:	C
Q8:	A

1 MARK EACH

1. Which one of the following is an **invalid variable** in MATLAB?
   A. Witchwood = 5+6;
   B. Hagatha witch = 3^2;
   C. Phantom9 = pi + [1 2 3];
   D. R0tt3n = [3, 5]
   E. Militia_shaw = [3 5; 6 7]

2. What is the MATLAB function for finding the **natural logarithm of x**?
   A. log10(x)
   B. nlog(x)
   C. 10log(x)
   D. log(x)
   E. None of the above

3. Which of the following statements creates a **logarithmically spaced vector from 10^0 to 10^5 (inclusive) with 60 points**?
   A. logspace(10^0, 10^5,60)
   B. logspace(10^0,60,10^5)
   C. logspace(0,5,60)
   D. logspace(0,60,5)
   E. None of the above

4. What are the plot characteristics of the following command? **plot(t, d, ‘ro-‘)**
    A. Red circles, dashed line
    B. Red circles, continuous line
    C. Orange rectangles, dashed line
    D. Red, dashed-dot line
    E. Orange rectangles, continuous line

5. A .txt file which contains only numerical data is imported using **X=importdata()**. Which of the following is true?
    A. X is a structure
    B. X is a string
    C. X is a double
    D. X is empty
    E. X is a character

6. Using **tline = fgetl** at the end of an open file in MATLAB results in tline equal to which of the following?
    A. 0 (logical)
    B. -1 (double)
    C. 1 (double)
    D. 1 (logical)
    E. -1 (string)

7. Which of the following anonymous functions replicates the following function file?
    function R = revs(x,y,z)
    R = x.^2 + y./z
    end
    A. R = @(x) x.^2 + y./z
    B. R @(x,y) = x.^2 + y./z
    C. R = @(x,y,z) x.^2 + y./z
    D. R = @(all) x.^2 + y./z
    E. None of the above

8. Which of the following statements is true about the following code?
    A=2; B = A + eps(A)/100
    A. A is equal to B
    B. A is less than B
    C. A is greater than B
    D. B is undefined
    E. Error

Question A5 (6 marks)

Write MATLAB code for the following scenarios, **ensuring that the commented instructions are followed**.

(a) Prompt the user for a value of *x*, and determine the value of *y* based on the following cases.

$$y(x) = \begin{cases} e^{x+1} & \text{for} \quad x < -1 \\ \cos(x) & \text{for} \quad -1 \leq x \leq 5 \\ 10(x-5) & \text{for} \quad x > 5 \end{cases}$$

```
% prompt the user for x
x = input('Input x: ');
```

0.5 marks for x input() function

```
% use if and elseif statements to
determine y
if x < -1
   y = exp(x+1);
elseif (x <= 5)
   y = cos(x);
else
   y=10*(x-5);
end
```

1 mark for if-statement structure

1 mark for conditions

0.5 marks for y equations

(b) The function primes(N) creates a vector containing prime numbers from 1 to N (inclusive). Given z=primes(1000), determine how many values in *z* are less than 500.

```
z = primes(1000)
counter = 0;

% use a for loop to go through each value of z
% use if statement to determine if counter should be incremented
for i=1:length(z)
   if z(i) < 500
      counter = counter+1;
   end
end
```

1 mark – for loop counter

0.5 marks - indexing

0.5 marks – updating counter

(c) Starting with *x*=13.6, continue to double x until it is larger than 1337.

```
x=13.6;

% use a while loop to check if x is larger than 1337
while x <= 1337
   x = 2*x;
end
```

0.5 marks for condition

0.5 marks for x equation

Question A6 (7 marks)

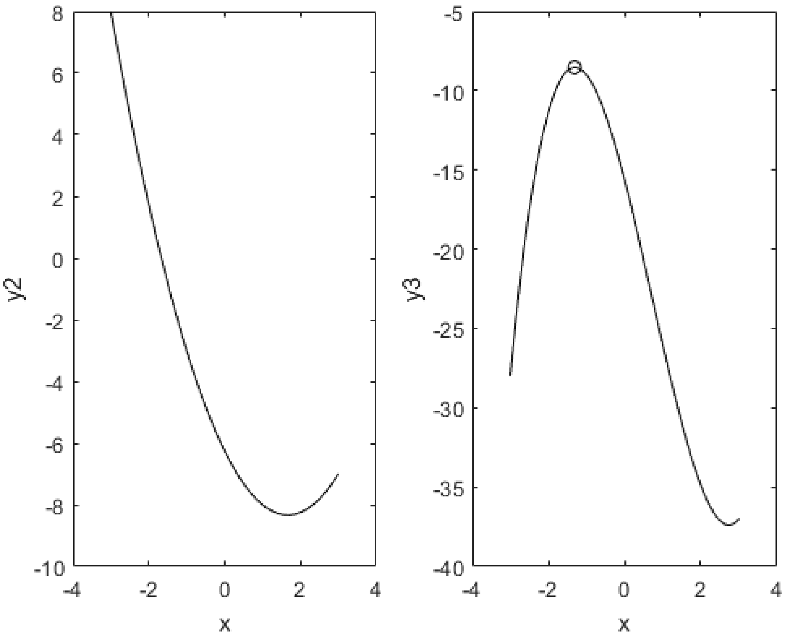
The figure below shows plots *y2* against *x* and *y3* against *x*, where:

$$y2 = x^2 - n^2$$

$$y3 = x^3 - n^3$$

Here, *x* is a vector of linearly spaced values from -3 to 3 (inclusive) with 300 points. The *n* variable is a vector of linearly spaced values from 1 to 4 (inclusive) with 300 points.

The line specifications for both plots are black continuous lines. The y3 plot has a maximum which is marked by a black circle.



Write MATLAB code in the following parts to reproduce the figure.

(a) This is the start of the m-file. **Clear all variables, close all figure windows and clear the command window.**

% start of m-file **Clear all; close all; clc;** **(0.5 marks total)**

(b) **Create all relevant variables for plotting.** Use element-by-element operators where appropriate.

% variable creation **x = linspace(-3,3,300);** **n = linspace(1,4,300);** **y2 = x.^2 - n.^2;** **y3 = x.^3 - n.^3;** **0.5 marks each for x, n, y2 and y3** **y2 and y3 can be function handles but must be called properly in subsequent parts** **No marks if element-by-element operator used incorrectly.** **(2 marks total)**

(c) Determine the **maximum *y3* value and the corresponding *x* value**.

% max y3 and corresponding x value

```
[maxy3, index] = max(y3);
corr_x = x(index);
```

0.5 marks for max with index
0.5 marks for corresponding x
(1 mark total)

(d) **Plot *y2* against *x*** in the left panel of the subplot and label the plot accordingly. The line specification is a black continuous line.

% plot y2 against x

```
subplot(1,2,1)
plot(x,y2,'k-')
xlabel('x')
ylabel('y2')
```

0.5 marks for subplot syntax
0.5 marks for plot syntax
0.5 marks for correct line specification
0.5 marks for labelling
(2 marks total)

(e) **Plot *y3* against *x*** in the right panel of the subplot and label the plot accordingly. The line specification is a black continuous line. Also, **mark the maximum *y3* value with a black circle** on the same plot. Refer to the figure at the start of this question.

% plot y3 against x
% mark the maximum y3 value

```
subplot(1,2,2)
plot(x,y3,'k-')
hold on
plot(corr_x,maxy3,'ko')
xlabel('x')
ylabel('y3')
```

0.5 marks for subplot syntax
0.5 marks for plot syntax
0.5 marks for plotting maxy3 (may need hold on)
(1.5 marks total)

PART B: ATTEMPT **ALL** QUESTIONS

Question B1 (15 marks)

The average concentration of a substance $\bar{c}$ (g/m^3) in a lake can be computed by integration via:

$$\bar{c} = \frac{\int_0^D c(z)A(z)\,\mathrm{d}z}{\int_0^D A(z)\,\mathrm{d}z}$$

where *z* is the depth below the surface in metres, the area *A* and concentration *c* vary with depth, and *D* is the maximum depth in metres. Here, ***D*=16**. The average concentration can be calculated based on the following data:

***z* (m)**	0	4	8	12	16
***c*(*z*) (g/m^3)**	10	8.5	7.4	5.2	4.1
***A*(*z*) (m^2)**	9.8	5.1	1.9	0.4	0
***c*(*z*)*A*(*z*) (g/m)**	98	43.35	14.06	2.08	0

(a) Use the **Composite Simpson's 1/3** rule with 4 segments to calculate the **numerator integral term** ($\int_0^D c(z)A(z)\,\mathrm{d}z$) in the average concentration equation over the entire depth of the lake. Show ALL your working and provide answers to 4 decimal places.

$$I = \frac{h}{3}\left[f(x_1) + 4\sum_{\substack{i=2,4,6,\ldots \\ i,\text{ even}}}^{n-1} f(x_i) + 2\sum_{\substack{j=3,5,7,\ldots \\ j,\text{ odd}}}^{n-2} f(x_j) + f(x_n) \right]$$

h=4 **0.5 marks**

$$I = \frac{4}{3}[98 + 4 * (43.35 + 2.08) + 2 * (14.06) + 0]$$ **0.5 marks**

$= 410.4533$

$\int_0^D c(z)A(z)\,\mathrm{d}z =$ **410.4533 (1 mark)**

(b) Use the **Composite Trapezoidal** rule with 4 segments to calculate the **denominator integral term** ($\int_0^D A(z)\,\mathrm{d}z$) in the average concentration equation over the entire depth of the lake. Show ALL your working and provide answers to 4 decimal places.

$$I = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n-1} f(x_i) + f(x_n) \right]$$

h=4 **0.5 marks**

$$I = \frac{4}{2}[9.8 + 2 \times (5.1 + 1.9 + 0.4) + 0]$$ **0.5 marks**

$= 49.2$

$\int_0^D A(z)\,\mathrm{d}z =$ **49.2000 (1 mark)**

(c) Hence, calculate the average concentration to 4 decimal places.

$\bar{c} =$ **8.3425 (1 mark)**

Consider now that the substance will be transferred to another location via a channel. The average **flow rate *Q*** can be calculated as:

$$Q = \int_0^B U(y)D(y)\,\mathrm{d}y$$

where ***B* is the total channel width (11m)**, *D* is the depth (m) and *y* is the distance from the bank (m). The distance and corresponding velocity-depth product is provided in the following table.

***y* (m)**	0	1	2	5	7	9	11
***U*(*y*)*D*(*y*) (m²/s)**	0.015	0.03	0.04	0.065	0.25	0.11	0.005

(d) Use a combination of the **Trapezoidal** rule, **Simpson's 1/3** rule, **Simpson's 3/8** rule to calculate the **average flow rate *Q*** over the total width of the channel. Show ALL your working and provide answers to 4 decimal places.

$$I = \frac{h}{3}\left[f(x_1) + 4\sum_{\substack{i=2,4,6,\ldots\\ i,\text{even}}}^{n-1} f(x_i) + 2\sum_{\substack{j=3,5,7,\ldots\\ j,\text{odd}}}^{n-2} f(x_j) + f(x_n) \right]$$

h=1 **0.5 marks**

$$Ia = \frac{1}{3}[0.015 + 4 * (0.03) + 0.04]$$ **0.5 marks**

$= 0.0583$ **1 mark**

$$I = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n-1} f(x_i) + f(x_n) \right]$$

h=3 **0.5 marks**

$$Ib = \frac{3}{2}[0.04 + 0.065]$$ **0.5 marks**

$=0.1575$ **1 mark**

$$I = \frac{3h}{8}[f(x_1) + 3f(x_2) + 3f(x_3) + f(x_4)]$$

h=2 **0.5 marks**

$$Ic = \frac{3 * 2}{8}[0.065 + 3 * (0.25 + 0.11) + 0.005]$$ **0.5 marks**

$=0.8625$ **1 mark**

I = Ia + Ib + Ic = 1.0783

$Q =$ **1.0783 (1 mark)**

(e) A MATLAB function that is supposed to perform composite Simpson's 1/3 rule is given below. **However, it contains errors.** Identify and correct the errors by providing the line number and correct code in the table below the code. There are 6 errors in total.

```
1  function I = composite_simpson_one_third(func,a,b,n)
2  % inputs
3  % func = name of function to be integrated
4  % a, b = integral limits
5  % n = number of segments
6  % output
7  % I = Integral estimate
8
9  h=(b-a)/n+1;
10
11 % Evaluating f(a)
12 s=f(a);
13
14 %Evaluating the even terms (add to first term)
15 x = a+h;
16 for j = 1:2:(n-1)
17     s= s+4*(func(x));
18     x = x+h;
19 end
20
21 %Evaluating the odd terms (add to first and even terms)
22 x = a+(2*h);
23 for i = 1:2:(n-2)
24     s = s+2*(func(x));
25     x = x+(2*h);
26 end
27
28 %Evaluating sum of terms including last term
29 s = s;
30
31 %Evaluating integral
32 Integral = h*s/3;
```

Line: 9	h = (b-a)/n
Line: 12	s = func(a)
Line: 18	x = x + (2*h)
Line: 23	i = 2:2:(n-2) or equivalent (e.g. i = 1:2:(n-3))
Line: 29	s = s + func(b)
Line: 32	I = h*s/3

0.5 marks for each line (3 marks total)

Question B2 (15 marks)

Solve the following ODE over the interval from *x* = 0 to 1 using a step size of 0.5 where *y*(0)=1.

$$\frac{dy}{dx} = (1 + 7x)\sqrt{2y}$$

For each method below, show all calculations for $y(x)$, including all intermediate variables for all iterations. Show your working in obtaining the *y* solution using

a) **Euler's method**

Iteration 1	
dy/dx(0) = (1+7(0))*sqrt(2*1) = 1.4142	**0.5+0.5 marks**
y(0.5) = 1 + 1.4142(0.5) = 1.7071	**0.5+0.5 marks**
iteration 2	
dy/dx(0.5) = (1+7(0.5))*sqrt(2*1.7071) = 8.3149	**0.5+0.5 marks**
y(1) = 1.7071 + 8.3149(0.5) = 5.8646	**0.5+0.5 marks**

b) **Heun's method**

iteration 1	
dy/dx(0) = (1+7(0))*sqrt(2*1) = 1.4142	
y(0.5*) = 1 + 1.4142(0.5) = 1.7071	
dy/dx(0.5*) = (1+7(0.5))*sqrt(2*1.7071) = 8.3149	**0.5+0.5 marks**
y(0.5) = 1+ 0.5*(1.4142+8.3149)(0.5) = 3.4323	**0.5+0.5 marks**
iteration 2	
dy/dx(0.5) = (1+7(0.5))*sqrt(2*3.4323) = 11.7902	
y(1*) = 3.4323 + 11.7902(0.5) = 9.3274	**0.5+0.5 marks**
dy/dx(1*) = (1+7(1))*sqrt(2*9.3274) = 34.5530	**0.5+0.5 marks**
y(0.5) = 3.4323+ 0.5*(11.7902+34.5530)(0.5) = 15.0181	**0.5+0.5 marks**

Place your results for $y(x)$ from both methods into the following table:

x	$y(x)$ Euler's method	$y(x)$ Heun's method
0	1	1
0.5	**1.7071**	**3.4323**
1	**5.8646**	**15.0181**

1 mark for correct table

c) The analytical solution to the ODE provided at the start of this question is given by:

$$y = \left(\frac{7x^2 + 2x}{2\sqrt{2}} + 1\right)^2$$

Calculate the percentage error in your predicted solutions **from parts (a) and (b) at *x*=1** and write them in the box below (use 1 decimal point in the %). Is this the result you expect? Why or why not? Provide your explanation in the box below.

Note: $\text{Error} = \left|\frac{\text{predicted value} - \text{actual value}}{\text{actual value}}\right| \times 100\%$

Error (Euler's method) at *x*=1	Error (Heun's method) at *x*=1
66.5% (+/- 0.5%)	**14.1% (+/- 0.5%)**

0.5 marks each error (1 mark total)

Yes, it is what I expect – Heun's method is 2nd order (i.e. more accurate) than Euler's method (that is 1st order).

1 mark

d) The function file for the midpoint method shown on the next page is **incomplete as lines 29-31 are missing code**. Complete the function file by writing the complete code in the box below.

Line: 29	`yhalf = y(i) + dydt(t(i),y(i))*(t(i+1)-t(i))/2;`
Line: 30	`thalf = t(i)+(t(i+1)-t(i))/2;`
Line: 31	`y(i+1) = y(i) + dydt(thalf,yhalf)*(t(i+1)-t(i));`

1 mark for each line

```
function [t,y] = midpoint(dydt,tspan,y0,h)
% [t,y] = midpoint(dydt,tspan,y0,h):
% uses midpoint method to solve an ODE
% input:
% dydt = function handle of the ODE, f(t,y)
% tspan = [<initial value>, <final value>] of independent variable
% y0 = initial value of dependent variable
% h = step size
% output:
% t = vector of independent variable
% y = vector of solution for dependent variable
% Input Validation: tspan

% Create all independant values, t
t = (tspan(1):h:tspan(2))';
n = length(t);

% if necessary, add an additional t so that range goes up to tspan(2)
if t(n)<tspan(2)
    t(n+1) = tspan(2);
    n = n+1;
end

% Implement Euler's method
y = y0*ones(n,1); % Preallocate y to improve efficiency

for i = 1:n-1
    % midpoint method
    yhalf =
    thalf =
    y(i+1) =
end
```

Question B3 (15 marks)

Consider the following equation:

$$f(x) = x^{10} - 1$$

(a) Perform 3 iterations of the bisection method to locate the root of *f*(*x*) using xl=0 (lower bound) and xu=1.3 (upper bound). Show your working for the **1st and 2nd iterations only** in the answer box **BELOW** the table. Then complete the following table using numbers to 4 decimal places.

Iteration	xl	xu	xr	f(xr)
1	0	1.3	0.65	-0.9865
2	0.65	1.3	0.975	-0.2237
3	0.975	1.3	1.1375	2.6267

1 mark each for 2nd and 3rd row (2 marks total)

Show working for the 1st and 2nd iterations here

1st iteration

xr = 0.5*(0+1.3) = 0.65 — 0.5 marks

f(xr) = 0.65^10 – 1 = -0.9865 — 0.5 marks

2nd iteration

f(xl) = -1 or f(xu) = 12.7858 (interval check) — 0.5 marks

Therefore xl=xr

xr = 0.5*(0.65+1.3) = 0.9750

f(xr) = -0.2237 — 0.5 marks

(b) The **bisection method appears to lose its convergence at the 3rd iteration** based on the value of f(xr). Provide an explanation for this.

f(x) increases rapidly after x=1. Since xr > 1 at the 3rd iteration, f(xr) appears to diverge rather than converge on zero. (Or significant curvature, or similar) — 1 mark

(c) Describe the difference between the bisection method and the false-position method in **how they estimate xr, using xl and xu**. Do not just refer to the equations.

Bisection method estimates xr by bisecting the interval of xu and xl — **1 mark**

False-position method estimates xr by drawing a chord between f(xu) and f(xl) and seeing where it intersects on the horizontal axis. — **1 mark**

(d) Write an m-file that uses the **Newton-Raphson method to determine the value of *x* which satisfies** $f(x) = x^{10} - 1 = 99$. Use a precision of 10^{-5} and an initial guess of x=0.5. Provide the appropriate code in the following parts to complete the m-file.

```
% define the anonymous functions to be solved when f(x)=99
g = @(x) x.^10 – 100;                                        0.5 marks
dg = @(x) 10*x.^9;                                           0.5 marks
```

```
% define the initial guess and precision
xi = 0.5;                                                    0.5 marks
precision = 1e-5;                                            0.5 marks
```

```
% calculate the initial values for the functions
gxi = g(xi);                                                 0.5 marks
dgxi = dg(xi);                                               0.5 marks
```

```
% jump start the while loop
gxr = 1; (or any value greater than precision)               0.5 marks
```

```
% iteration for Newton-Raphson method starts
while abs(gxr) > precision                                   0.5 marks
   xr = xi - gxi/dgxi;                                       0.5 marks
   gxr = g(xr);                                              0.5 marks
   xi = xr;                                                  0.5 marks
   gxi = g(xi);                                              0.5 marks
   dgxi = dg(xi);                                            0.5 marks
or equivalent code.
end
```

```
% return the root value
root = xr;                                                   0.5 marks
```

```
% print the root to 7 decimal places with a width of 10
fprintf('%10.7f', root)                                      1 mark
```

Question B4 (15 marks)

The rate of an enzyme-catalyzed reaction is represented by the Michaelis-Menten equation as follows:

$$v = \frac{v_m S}{K + S}$$

where v (dependent variable) is the rate of the enzyme catalyzed reaction and v_m is the maximum reaction rate. S (independent variable) represents the substrate concentration and K is a constant related to the substrate concentration. Below is a set of experimentally measured kinetic data for an enzyme catalyzed reaction.

S	1.3	1.8	3	6	9
v	0.08	0.125	0.2	0.3	0.333

(a) **Linearise this non-linear model**. Ensure you show **ALL** steps and working.

v = (vm*S)/(K+S)

Take reciprocal of both sides:

1/v = (K+S)/(vm*S) **1 mark**

Simplify:

1/v = K/vm*(1/S) + 1/vm (Or equivalent linearised equations) **1 mark**

(b) Relate the non-linear variables (S, v, v_m and K) to the linear variables (y, a_0, a_1 and x) below.

	y	=	a_0	+	a_1	x
Linearised model:	**1/v**	=	**1/vm**	+	**K/vm**	**1/S**

2 marks for correct relationships (0.5 marks each)

(c) You will be required to fit a straight line to the linearized data using Least Squares Regression to obtain an equation of the form y = a_0 + a_1*x. **Show the values you need to first calculate by filling in the table below**. (Do **NOT** show the arithmetic needed to calculate the sums/mean).

i	x_i	y_i	$x_i y_i$	x_i^2
1	0.7692	12.5	9.6154	0.5917
2	0.5555	8	4.4444	0.3086
3	0.3333	5	1.6667	0.1111
4	0.1667	3.3333	0.5556	0.0278
5	0.1111	3.0030	0.3337	0.0123
SUM	**1.9359**	**31.8363**	**16.6157**	**1.0516**
MEAN	**0.3872**	**6.3673**		

3 marks for correct sums and averages (0.5 each)

(d) **ASSUME you obtained the values in the table below** (instead of the values you calculated above in part (c)) and then **calculate the linear coefficients a_0 and a_1. Show your working**.

i	x_i	y_i	$x_i y_i$	x_i^2
SUM	**2**	**32**	**16**	**1**
MEAN	**0.4**	**6.5**		

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_1 = \frac{5 * 16 - 2 * 32}{5 * 1 - 2^2} = 16$$ **1 mark**

$$a_0 = \bar{y} - a_1 \bar{x}$$

$$a_0 = 6.5 - 16 * 0.4 = 0.1000$$ **1 mark**

(e) From your results in part (d), **calculate the non-linear coefficients (v_m and K)**. Finally, show the non-linear equation in the box as requested.

a0 = 1/vm **0.5 marks**

vm = 1/a0 = 1/0.1 = 10 **0.5 marks**

a1 = K/vm **0.5 marks**

K = a1*vm = 16*10 = 160 **0.5 marks**

Equation of fitted curve:

$$v = \frac{10S}{160 + S}$$

1 mark for correct equation

(f) Write a function file which accepts vectors *x* and *y* and **performs the least-squares linear regression on a set of linear data stored in *x* and *y***. Complete the following code:

```
function  [slope, intercept] = linreg(x,y)
% [slope, intercept] = linreg(x,y)
% inputs
% x and y are vectors containing linearised data
% outputs
% slope is the gradient of the fitted line
% intercept is the value of the intercept on the vertical axis

n = length(x)
sx = sum(x)                                            0.5 marks
sy =  sum(y)                                           0.5 marks
sx2 = sum(x.^2)                                        0.5 marks
sxy = sum(x.*y)                                        0.5 marks
slope = (n*sxy – sx*sy)/(n*sx2 – sx^2)                 0.5 marks
intercept = mean(y) – slope*mean(x)                    0.5 marks
```

END of EXAM

Blank page for workings (will not be marked)

Blank page for workings (will not be marked)

Blank page for workings (will not be marked)

MATLAB Information and Formulas

OPERATOR PRECEDENCE

1	()	Parentheses
2	.' ' .^ ^	Transpose, Matrix Transpose, Power, Matrix Power
3	~	Logical Negation
4	.* * ./ / .\ \	Multiplication, Matrix Multiplication, Right Division, Matrix Right Division, Left Division, Matrix Left Division
5	+ -	Addition Subtraction
6	:	Colon Operator
7	< <= > >= == ~=	Less Than, Less Than Or Equal To, Greater Than, Greater Than Or Equal To, Equal To, Not Equal To
8	&	Element-wise AND
9	|	Element-wise OR
10	&&	Short-circuit AND
11	||	Short-circuit OR

fprintf SPECIFIER

%d	Integer
%f	Fixed-Point Notation
%e	Exponential Notation
%s	String of Characters
%c	Single Character
\t	Horizontal Tab
\n	New Line
%%	Percent Character
\''	Single Quote Mark
\\	Backslash
\b	Backspace

Fixed-Point Notation Syntax
%<field_width>.<precision>f

COLOR SPECIFIER

r	Red
g	Green
b	Blue
c	Cyan
m	Magenta
y	Yellow
k	Black
w	White

LINE STYLE SPECIFIER

-	Solid Line
--	Dashed Line
:	Dotted Line
-.	Dash-dot Line

MARKER TYPE SPECIFIER

+	Plus Sign
o	Circle
*	Asterisk
.	Point
x	Cross
s	Square
d	Diamond
^	Triangle (Up)
v	Triangle (Down)
>	Triangle (Right)
<	Triangle (Left)

Root Finding

Bisection Method

$$x_r = \frac{x_l + x_u}{2}$$

False Position Method

$$x_r = x_u - \frac{f(x_u)(x_l - x_u)}{f(x_l) - f(x_u)}$$

Newton-Raphson Method

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Secant Method

$$x_{i+1} = x_i - \frac{f(x_i)(x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}$$

Modified Secant Method

$$x_{i+1} = x_i - \frac{\delta x_i f(x_i)}{f(x_i + \delta x_i) - f(x_i)}$$

Curve Fitting

Linear Regression:

$$y = a_o + a_1 x$$

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_0 = \bar{y} - a_1 \bar{x}$$

Coefficient of Determination

$$r^2 = \frac{S_t - S_r}{S_t}$$

Linearizing Nonlinear Models

Nonlinear	Linearized
$y = \alpha_1 e^{\beta_1 x}$	$\ln y = \ln \alpha_1 + \beta_1 x$
$y = \alpha_2 x^{\beta_2}$	$\log y = \log \alpha_2 + \beta_2 \log x$
$y = \alpha_3 \dfrac{x}{\beta_3 + x}$	$\dfrac{1}{y} = \dfrac{1}{\alpha_3} + \dfrac{\beta_3}{\alpha_3}\dfrac{1}{x}$

Standard Deviation

$$S_t = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$s_y = \sqrt{\frac{S_t}{n-1}}$$

Standard Error of the Regression Estimate

$$S_r = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)^2$$

$$s_{y/x} = \sqrt{\frac{S_r}{n-2}}$$

Numerical Integration (n is the number of points)

Trapezoidal Rule:

$$I = (b-a)\frac{f(b)+f(a)}{2}$$

$$E_t = -\frac{1}{12} f''(\xi)(b-a)^3$$

Composite Trapezoidal Rule

$$I = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n-1} f(x_i) + f(x_n) \right]$$

where $\quad h = \dfrac{(b-a)}{n-1}$

Composite Trapezoidal Rule with Unequal Segments

$$I = (x_2 - x_1)\frac{f(x_2)+f(x_1)}{2} + (x_3 - x_2)\frac{f(x_3)+f(x_2)}{2} + \cdots + (x_n - x_{n-1})\frac{f(x_{n-1})+f(x_n)}{2}$$

Simpson's 1/3 Rule

$$I = \frac{h}{3}[f(x_1) + 4f(x_2) + f(x_3)]$$

$$E_t = -\frac{1}{2880} f^{(4)}(\xi)(b-a)^5$$

Simpson's 3/8 Rule

$$I = \frac{3h}{8}[f(x_1) + 3f(x_2) + 3f(x_3) + f(x_4)]$$

$$E_t = -\frac{1}{6480} f^{(4)}(\xi)(b-a)^5$$

Composite Simpson's 1/3 Rule:

$$I = \frac{h}{3}\left[f(x_1) + 4\sum_{\substack{i=2,4,6,\ldots \\ i,\ \text{even}}}^{n-1} f(x_i) + 2\sum_{\substack{j=3,5,7,\ldots \\ j,\ \text{odd}}}^{n-2} f(x_j) + f(x_n) \right]$$

ODE: Initial Value Problems

Euler's Method

$$y_{i+1} = y_i + f(t_i, y_i)h$$

Heun's Method

$$y_{i+1}^0 = y_i + f(t_i, y_i)h$$

$$y_{i+1} = y_i + \frac{f(t_i, y_i) + f(t_{i+1}, y_{i+1}^0)}{2}h$$

Midpoint Method

$$y_{i+1/2} = y_i + f(t_i, y_i)\frac{h}{2}$$

$$t_{i+1/2} = t_i + \frac{h}{2}$$

$$y_{i+1} = y_i + f(t_{i+1/2}, y_{i+1/2})h$$