ENG1060: COMPUTING FOR ENGINEERS

Lab 5 – Week 6

2020 OCT NOV

Welcome to lab 5. Remember that laboratories continuously build on previously learned concepts and lab tasks. Therefore, it is crucial that you complete all previous labs before attempting the current one.

Self-study:
Students are expected to attempt these questions during their own self-study time, prior to this lab session. There may be questions that require functions not covered in the workshops. Remember to use MATLAB's built-in help for documentation and examples.

Learning outcomes:

1. To revise input and output processes, and IF statements
2. To distinguish the usage between for loops and while loops
3. To identify processes to be repeated and construct loops to achieve the repetition
4. To apply appropriate loop structures to solve complex problems
5. To use the debugging tool and develop debugging skills while solving problems involving loops

Background:
Engineers do not want to perform repetitive tasks! Leave that to the computers. Loops are used to repeat blocks of code either for a specified number of times or until a specified condition is met. The implementation of loops greatly increases the number of calculations being performed and therefore, increases the potential for bugs to be introduced. Debugging tools are used to help resolve buggy code.

Primary workshops involved:

- Workshop 4: Input, output and IF statements
- Workshop 5: Loops and debugging

Assessment:
This laboratory comprises **2.5%** of your final grade. The questions are designed to test your recollection of the workshop material and to build upon important programming skills. You will be assessed on the quality of your programming style as well as the results produced by your programs during your laboratory session by the demonstrators. Save your work in **m-files** named ***lab1t1.m***, ***lab2t2.m,*** etc. Inability to answer the demonstrator's questions will result in zero marks, at the demonstrator's discretion.

Team tasks begin at the start of the lab session so please ensure you arrive on time to form your groups. Students who arrive late will not be able to participate in the team tasks as teams will have already formed and will therefore forfeit all associated marks. These tasks will be assessed during class.

Lab submission instructions

Follow the instructions below while submitting your lab tasks.

Team tasks:
The team tasks are designed for students to test and demonstrate their understanding of the fundamental concepts specific to that lab. These tasks will occur at the start of the lab and will be assessed on the spot. Demonstrators will advise on how these will be conducted. Most team tasks do not require the use of MATLAB but MATLAB should be used for checking purposes.

Individual tasks:
The individual tasks are designed for students to apply the fundamentals covered in the team tasks in a variety of contexts. These tasks should be completed in separate m-files. There is typically one m-file per task unless the task requires an accompanying function file (lab 3 onwards). Label the files appropriately. E.g. lab6t1.m, lab6t2.m, eridium.m, etc.

Deadline:
The lab tasks are due next Friday at 9am (MYT) or 12pm (AEDT). Late submissions will not be accepted. Students will need to apply for special consideration after this time.

Submission:
Submit your lab tasks by:
1) Answering questions in Google Form, and
2) Submitting one .zip file which includes all individual tasks.

The lab .zip file submission links can be found on Moodle under the weekly sections, namely Post-class: Lab participation & submission. The submission box ("Laboratory 5") will only accept one .zip file. Zipping instructions are dependent on the OS you are using.

Your zip file should include the separate m-files for the individual tasks including function files.

It is good practice to download your own submission and check that the files you have uploaded are correct. Test run your m-files that you download. You are able to update your submission until the deadline. Any update to the submission after the deadline will be considered late.

Grade and feedback:
The team will endeavour to grade your lab files by Tuesday of the following week. Grades and feedback can be viewed through the Moodle Gradebook, which is available on the left side pane on the ENG1060 Moodle site.

Lab 5 – Assessed questions

TASK 1

[2 MARKS – L05TE]

Note: Team tasks are designed for students to recall material that they should be familiar with through the workshops and practice of the individual questions prior to this lab session.

Students will be split into groups of 3-4 for the team tasks. Students in each group must explain aspects of the question below to receive the marks. Ensure that everyone has equal learning opportunities. Additionally, ask your table for help.

if statements:

Consider the following codes, which is supposed to return $y = 1$ if x is negative, $y = 2$ if x is positive, and $y = 3$ if $x\ == \ 5$ (in that order):

Code A:

```
x = 5;
if x < 0
    y = 1;
elseif x > 0
    y = 2;
elseif x == 5
    y = 3;
end
```

Code B:

```
x = 5;
if x < 0
    y = 1;
    if x > 0
        y = 2;
        if x ==5
            y = 3;
        end
    end
end
```

Complete the following for both Codes A and B:

1. Copy the code in the slide.
2. Identify/highlight the lines of code that are executed by MATLAB and determine the output of y.
3. Describe the differences between Code A and Code B.
4. Discuss the task and explore any misunderstandings.
5. Have a demonstrator assess your understanding.

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Loops:

Consider the following codes:

Code A:
```
data = [3, 7, 8, 1, 4];
for i = 1:length(data)
    x = data.^2;
end
```

Code C:
```
data = [3, 7, 8, 1, 4];
for i = 1:length(data)
    x = data(i).^2;
end
```

Code B:
```
data = [3, 7, 8, 1, 4];
for i = 1:length(data)
    x(i) = data(i).^2;
end
```

Code D:
```
data = [3, 7, 8, 1, 4];
for i = data
    x(i) = data(i).^2;
end
```

You will be assigned two of the above Codes.

1. Copy the code in the slide.
2. Provide the values of i and x for each iteration – use the debugger tool if required. Example table shown below.

i	x
1	??
2	??
…	…

3. Write a while loop that triples the value of x with each iteration, with x starting as 5. Create a counter to determine the number of times x is tripled before it is greater than 1000.
4. Discuss the task and explore any misunderstandings. Also, browse the work of other teams related to the other Codes and ensure that you have understood it as concepts from all sets may be required for the individual tasks.
5. Have a demonstrator assess your understanding

Remember good programming practices for all tasks even if not specifically stated. This includes, but is not limited to:

- using clc, close all, and clear all, where appropriate
- suppressing outputs where appropriate
- labelling all plots, and providing a legend where appropriate
- fprintf statements containing relevant answers

TASK 2

[4 MARKS]

Engineers often need to estimate the pressures and volumes of a gas in a container. The *van der Waals* equation is often used for this purpose. It is

$$P = \frac{RT}{V-b} - \frac{a}{V^2}$$

where the term b is a correction for the volume of the molecules and the term a/V^2 is a correction for molecular attractions. The gas constant is R, the *absolute* temperature is T, and the gas specific volume is V. The value of R is the same for all gases; it is $R = 0.08206\ L\text{-}atm/mol\text{-}K$. The values of a and b depend on the type of gas. Some values are given in the following table.

Option	Gas	$a(L^2\text{-}atm/mol^2)$	$b(L/mol)$
He	Helium	0.0341	0.0237
H2	Hydrogen	0.244	0.0266
O2	Oxygen	1.36	0.0318
CL2	Chlorine	6.49	0.0562
CO2	Carbon dioxide	3.59	0.0427

Write an m-file that achieves the following:

1. Uses fprintf to print out the available gas options to the command window
2. Prompt the user to input T, V, and a string variable containing the name of a gas listed in the table
   - Check if the input values are valid (e.g. negative values for both T and V, or incorrect gas option). Your program should continue prompting the user to enter values until they are valid.
3. Use a switch statement to compute the pressure P of carbon dioxide (CO_2) for $T = 300K$ and $V = 20L/mol$. Use fprintf to print a statement including the gas selected and the pressure.

Hint:

1. Use a while loop to ensure that the user enters valid inputs.
2. Use strcmp() to compare strings.

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 3
[2 MARKS]

The roots of the quadratic equation $ax^2 + bx + c = 0$ are given by

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

where the discriminant of the equation is defined to be $\Delta = b^2 - 4ac$.

Based on the sign of the discriminant, it is possible to characterise the nature of the roots according to the following relationships:

1. $\Delta > 0$: A pair of non-equal real valued roots
2. $\Delta = 0$: A pair of equal real valued roots
3. $\Delta < 0$: A pair of non-equal complex valued roots

A. Write a **function** that uses a, b and c as inputs to compute the discriminant and the roots of the quadratic equation.

function [x1, x2, discriminant] = my_quadratic(a,b,c)

B. Write an m-file that prompts the user to input values of a, b and c and uses the function in written in part A to output the roots and its characteristics. Print the roots to 3 decimal places. Ensure that your program checks that the user enters a **non-zero** value for the a coefficient.

Note: You may need to separate the complex variable into real and imaginary terms to display them as x = g + hi. You may want to use the real() and imag() functions.

Your output should look similar to the following:
Quadratic Program - What are the quadratic coefficients?
a = 2
b = -1
c = 1
For quadratic coefficients a=2.00, b=-1.00 and c=1.00
A pair of non-equal complex valued roots exists and are x1=0.250 + 0.661i and x2=0.250 - 0.661i

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 4

[5 MARKS]

Many real-world objects travel in a jagged manner that is best represented by piecewise functions. For example, the velocity of a rocket may change sharply when thrusters are enabled or disabled and due to various external factors. The velocity of a rocket can be modelled by ***v*(*t*)** where ***t*** represents the time.

$$v(t) = \begin{cases} 11t^2 - 5t & 0 < t \leq 10 \\ 1100 - 5t & 10 < t \leq 20 \\ 50t + 2(t-20)^{2.5} & 20 < t \leq 30 \\ 1520e^{-0.1(t-30)} & t > 30 \\ 0 & \text{otherwise} \end{cases}$$

Write a **function** that takes a start and end time as inputs and returns the vectors ***v*** and ***t***. The vector ***t*** should contain values between **t_start** and **t_end** at 0.01 increments. The vector ***v*** should contain ***v*(*t*)** for each value in ***t***.

function [t, v] = VPiecewise(t_start, t_end)

Use the function in an m-file to plot the velocities for ***t* = -5 to 80**.

Hint: Use if statements to break up the different profile conditions.

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 5

[7 MARKS]

A Fibonacci sequence is composed of elements created by adding the two previous elements. The simplest Fibonacci sequence starts with $0, 1$ and proceeds as follows:

$$0,1, 1, 2, 3, 5, 8, 13, \dots$$

However, a Fibonacci sequence can be created with any two starting numbers (a, b).

A. Write an m-file that calculates the Fibonacci sequence (10 terms long) as a vector using a for loop with $a = 5$ and $b = 8$. Use fprintf() to output the sequence. You may want to use num2str() with a %s specifier to print the Fibonacci sequence. Example output below for $a = 0$, $b = 1$ with 10 terms.

```
Fibonacci sequence with starting numbers [0, 1] is
 0    1    1    2    3    5    8   13   21   34
```

B. One interesting property of a Fibonacci sequence is that the ratio of the values of adjacent members of the sequence approaches a number called "golden ratio": $\frac{1+\sqrt{5}}{2}$.

Using the same two initial numbers specified in part A, continue calculating Fibonacci terms in the sequence until the ratio of adjacent values converges to the golden ratio within an absolute difference (error) of 10^{-5}. Plot the calculated ratios against each ratio pair number using cross markers connected with solid lines, and compare with the exact ratio (dashed line) in the same plot. Also, use fprintf() to print the number of Fibonacci terms required to achieve the absolute error. Example output below.

```
15 Fibonacci numbers needed to achieve absolute error less than 1.000000e-05
```

Use pen and paper to understand the process before coding this. Example calculations for the first few terms are shown below.

Terms = 2
Fibonacci sequence = $[5, 8]$
Ratio pair 1 = $8/5 = 1.6$
Error = $\left|1.6 - \frac{1+\sqrt{5}}{2}\right| = 0.0180$

Terms = 3
Fibonacci sequence = $[5, 8, 13]$
Ratio pair 2 = $13/8 = 1.625$
Error = $\left|1.625 - \frac{1+\sqrt{5}}{2}\right| = 0.0070$

Continue until error is below 10^{-5}.

2 marks deducted for poor programming practices (missing comments, unnecessary outputs, no axis labels, inefficient coding, etc.)

END OF ASSESSED QUESTIONS

The remainder of this document contains supplementary and exam-type questions for extended learning. Use your allocated lab time wisely!

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Lab 5 – Supplementary questions

These questions are provided for your additional learning and are not assessed in any way. You may find some of these questions challenging and may need to seek and examine functions that are not taught in this unit. Remember to use the help documentation. Coded solutions will not be provided on Moodle. Ask your demonstrators or use the discussion board to discuss any issues you are encountering.

TASK 1S

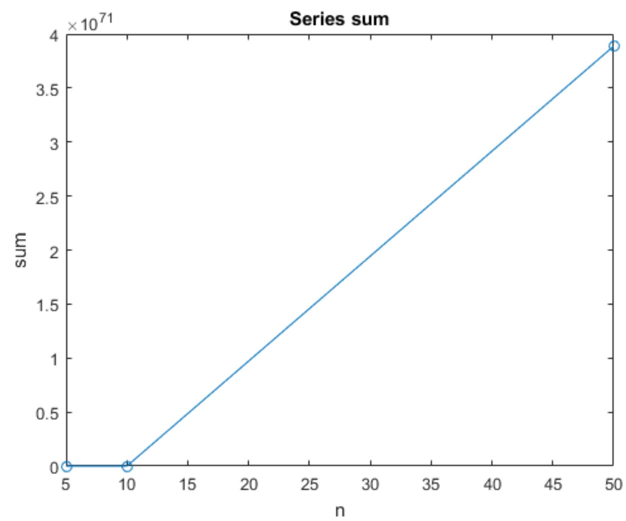
Calculate the sum of the first *n* terms of the series:

$$\sum_{k=1}^{n} \frac{(-1)^k k^{(k+1)}}{2^k}$$

For *n*=5, 10, 50, and 500. Plot the sums as a function of *n*. Are the results as expected?

SOLUTION

```
n = 5, series sum = -4.3291e+02
n = 10, series sum = 9.1329e+07
n = 50, series sum = 3.8877e+71
n = 500, series sum = NaN
```



 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 2S

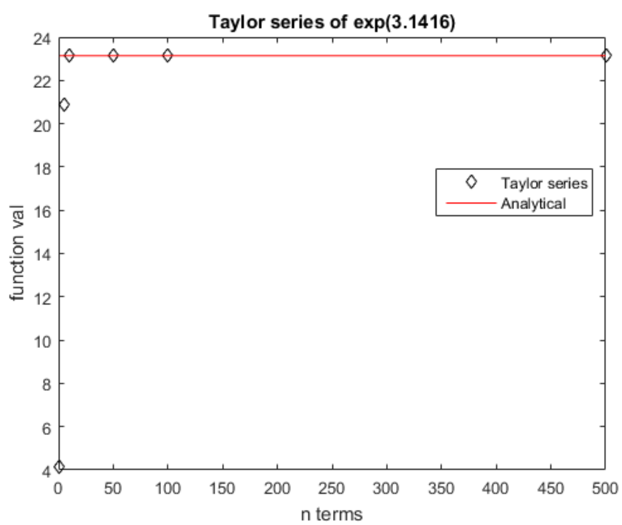
The function $f(x) = e^x$ can be represented in a Taylor series by

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

Calculate the sum of the first n terms of the series for n=1, 5, 10, 50, 100 and 500 with x being an input by the user. Plot the sums as a function of n as black diamonds and the MATLAB value of e^x as a red line to illustrate the convergence.

SOLUTION

```
Enter x value of exp(x) : pi
f(n) = 4.1416 using n = 1. Error = 8.2103e+01 %
f(n) = 20.8530 using n = 5. Error = 9.8861e+00 %
f(n) = 23.1308 using n = 10. Error = 4.2774e-02 %
f(n) = 23.1407 using n = 50. Error = 3.0705e-14 %
f(n) = 23.1407 using n = 100. Error = 3.0705e-14 %
f(n) = 23.1407 using n = 500. Error = 3.0705e-14 %
```



 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 3S

Write an m-file that finds the smallest even integer that is divisible by 13 and by 16 whose square root is greater than 120. Use fprintf to print the following statement: “The required number is XX” where XX is the appropriate number.

SOLUTION

```
The required number is 14560.
```

TASK 4S

Write an m-file which creates an n-by-m matrix with elements that have the following properties:

- The value of each element in the first row is the number of the column
- The value of each element in the first column is the number of the row
- The rest of the elements each has a value equal to the sum of the element above it and the element to its left.

The m-file should prompt the user to enter the values for n and m, the dimensions of the matrix.

For example, a 5-by-5 matrix will output the following result:

1	2	3	4	5
2	4	7	11	16
3	7	14	25	41
4	11	25	50	91
5	16	41	91	182

SOLUTION

```
Enter number of rows : 3
Enter number of cols : 5
M =
     1     2     3     4     5
     2     4     7    11    16
     3     7    14    25    41

Enter number of rows : 4
Enter number of cols : 8
M =
     1     2     3     4     5     6     7     8
     2     4     7    11    16    22    29    37
     3     7    14    25    41    63    92   129
     4    11    25    50    91   154   246   375
```

TASK 5S

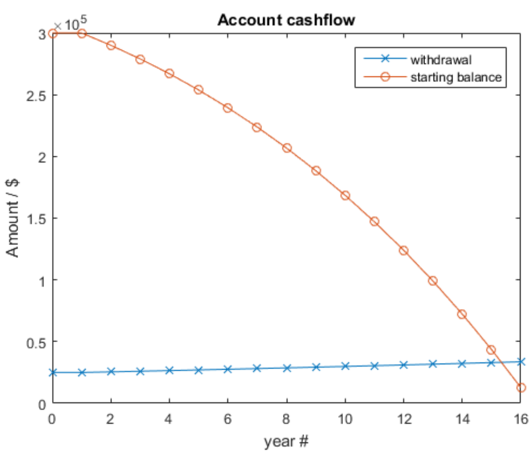
 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

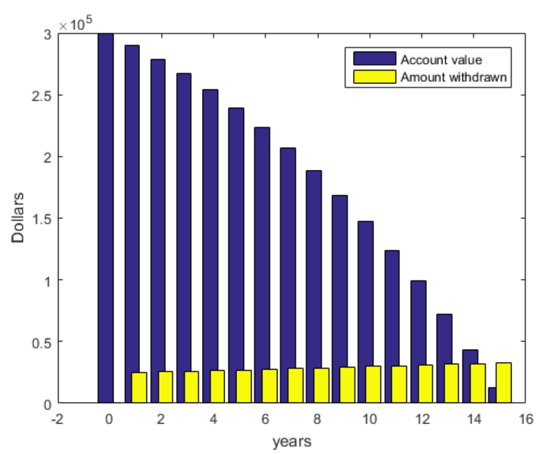
A person in retirement is depositing $300,000 in a saving account that pays 5% interest per year. The person plans to withdraw money from the account once a year. He starts by withdrawing $25,000 after the first year, and in future years he increases the amount he withdraws according to the inflation rate. For example, if the inflation rate is 3%, he withdraws $25,750 after the second year. Calculate the number of years the money in the account will last assuming a constant yearly inflation rate of 2%. Make a plot that shows the yearly withdrawals and the balance of the account over the years.

SOLUTION

Year	Start/$	Interest/$	Inflation/$	Out/$	End/$
1	300000.00	15000.00	0.00	25000.00	290000.00
2	290000.00	14500.00	500.00	25500.00	279000.00
3	279000.00	13950.00	510.00	26010.00	266940.00
4	266940.00	13347.00	520.20	26530.20	253756.80
5	253756.80	12687.84	530.60	27060.80	239383.84
6	239383.84	11969.19	541.22	27602.02	223751.01
7	223751.01	11187.55	552.04	28154.06	206784.50
8	206784.50	10339.22	563.08	28717.14	188406.58
9	188406.58	9420.33	574.34	29291.48	168535.43
10	168535.43	8426.77	585.83	29877.31	147084.88
11	147084.88	7354.24	597.55	30474.86	123964.27
12	123964.27	6198.21	609.50	31084.36	99078.12
13	99078.12	4953.91	621.69	31706.04	72325.98
14	72325.98	3616.30	634.12	32340.17	43602.12
15	43602.12	2180.11	646.80	32986.97	12795.25
16	12795.25	639.76	659.74	33646.71	-20211.69

```
Hooman will not survive year #16.
They’ve got bills they've gotta pay
So they gon' work, work, work every day! (Bills - LunchMoney Lewis)
```





 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Lab 5 – Exam-type questions

These questions are provided for your additional learning and are not assessed in any way. You may find these type of questions on ENG1060 exams. Solutions will not be provided on Moodle. Ask your demonstrators or use the discussion board to discuss any issues you are encountering. Additionally, you may use the exam collaboration document on Moodle (under the exam section) to share your answers.

1. A) Explain how short-circuit operators work in if-statements.

B) Consider the code below. Provide the first two values of **ii** that satisfy the if conditions and the corresponding output of **log_check**.

```
Data = 2:20
for ii = 1:length(Data)
      if (Data(ii) == 3) || (Data(ii) >= 7) && mod(Data(ii),3) == 1
           log_check = Data(ii)/(5*ii);
      end
end
```

ii =	log_check =
ii =	log_check =

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

2. An engineer is designing a machine that predicts heads or tails when a coin is flipped by using a camera. There are many physical considerations, and the final part of the program is to report the predicted state. i.e. "heads" or "tails".

   The program creates a map of possible outcomes based on various factors (such as the initial launch speed, aerodynamics of coin, bounce off the table etc.) and creates an 8 x 10 matrix, A, of possible outcomes between 0 and 1 (where 1 = "heads"). Values of 0.5 and greater are to be regarded as heads, and less than 0.5 as tails.

   To predict the final outcome, you realize that the probability can be either 0, 0.25, 0.75 or 1 in a process that is known as "binning". You decide to use "for" loops as you will use a second confidence predictor matrix (given to you) to decide what to do. Your supervisor decides that you cannot use the round() function.

Provide the code in the box below that does the following:

- An 8 x 10 matrix, **A**, of random camera predictions between 0 and 1 is provided. A predictor confidence matrix, **Conf**, is also provided (an 8 x 10 matrix of random numbers between 0 and 1)
- Using two for loops to go through the values of **A** and **Conf** and use if statements with the following conditions
  - If the value in **A** is less than 0.5 and the value in **Conf** is less than 0.6 then set **A** to be 0.25
  - If the value in **A** is greater than or equal to 0.5 and the value in **Conf** is less than 0.6 then set **A** to be 0.75
  - If the value in **A** is less than 0.5 and the value in **Conf** is greater than or equal to 0.6 then set **A** to be 0
  - If the value in **A** is greater than or equal to 0.5 and the value in **Conf** is greater than or equal to 0.6 then set **A** to be 1
- Calculate the sum of the modified **A** matrix *across the rows.*

```
A = rand(8,10);
Conf = rand(8,10);
```

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

3. Write a MATLAB program that will smooth an array of noisy data using a two-point average. Assume the array of data is a row vector and that a backwards average will be used for the smoothing.

   For a backwards two-point average, $X_{avg,k} = \frac{x_k + x_{k-1}}{2}$, except for k starts off at 2. Use a loop rather than array operations. The first line of code is provided to create a 1-by-100 row vector containing noisy data based on a normal distribution with a mean of 10.0 and a standard deviation of 3. Complete the code to calculate all of 99 $X_{avg,k}$ values.

```
%normal dist noise - randn(maxval, length of vector)
noisyData = 10 + 3*randn(1,100);

% Pre-allocate avgData



% Loop to average data
```

4. Write code using a *switch* structure that provides the 8-bit CMYK color value for the three primary light colours red, green and blue (see table below). The code to prompt the user for an input is provided. The input should be 'red', 'green', or 'blue'. If the input is neither 'red', 'green', or 'blue', output the 'black' colour spectrum vector. Store the output colour vectors in the variable named **CMYKcol**.

Colour:	**Red**	**Green**	**Blue**	**Black**
CMYK vector	[0 255 255 0]	[255 0 255 0]	[255 255 0 0]	[0 0 0 255]

```
Colour_input = input(Please enter the text red, green, or blue: ', 's')

%switch structure
```

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

5. An engineer wants to create a matrix, **X**, with m-by-n elements. There is a corresponding **Y** matrix with dimensions n-by-m. The size of **Y** is the size of the transposed **X** and therefore, the corresponding elements will require a flipped index.

   The conditions and operations required are:
   - check if the value of Y is greater than 5, and if so,
   - add this value to the flipped index element of X.

   An incomplete code is shown below, which uses **rand()** to create the **Y** vector and **randi()** for the **X** vector.

```
R = randi(IMAX,N) returns an N-by-N matrix containing pseudorandom
integer values drawn from the discrete uniform distribution on 1:IMAX.
    randi(IMAX,M,N) or randi(IMAX,[M,N]) returns an M-by-N matrix.
```

```
%creating X and Y
m = 20;
n = 25;
X = rand(m,n);
Y = 1+ randi(15,n,m);


for r=1:n
  for k = 1:m
     if ____
        ____ = ____
  end
end
```

   Select the correct syntax that achieves the check and operation outlined above

   a) if Y(r,k) > 5
      X(r,k) = X(r,k) + Y(r,k)

   b) if Y(k,r) >= 5
      X(r,k) = X(r,k) + Y(k,r)

   c) if Y(r,k) > 5
      X(r,k) = X(k,r) + Y(r,k)

   d) if Y(r,k) > 5
      X(k,r) = X(k,r) + Y(r,k)

   e) if Y(k,r) >= 5
      X(k,r) = X(k,r) + Y(r,k)

 Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.