**BE2080 week#2**

**Submission:**

**Please copy your Matlab scripts and paste to the word document for submission.**

**1**) (10 points) Write a Matlab script that will calculate the volume of a hollow sphere,



where ri=5.1 cm is the inner radius and ro=6.8 cm is the outer radius. Assign the volume to a third variable. Include comments in the script.

%This script calculates the volume of a hollow sphere.

r\_i = 5.1; %inner radius in centimeters

r\_o = 6.8; %outer radius in centimeters

volume = (4\*pi/3)\*(r\_o^3 - r\_i^3); %calculates volume

disp(volume); %displays result

Or could do:

q1.m

%{

This script calculates the volume of a hollow sphere

using the volume function.

%}

r\_i = 5.1; %inner radius in centimeters

r\_o = 6.8; %outer radius in centimeters

volume(r\_i, r\_o)

volume.m

function [v] = volume(r\_i, r\_o)

%This function calculates the volume of a hollow sphere.

v = (4\*pi/3)\*(r\_o^3 - r\_i^3); %equation to calculate volume

end

2) (10 points) Write a script that will first use an **input** statement to prompt the user for a real number; then it will use the **fprintf** function to print the value of this variable using 3 decimal places.

%This script prompts user for a real number and then prints the result rounded to three decimal places.

user\_input = input('Enter a real number: ');

fprintf('%.3f\n', user\_input);

**3**) (10 points) Write a script called *Greeting* that will first prompt the user for a name, and then prints a greeting with the name. Your script shall work exactly like the example below:

*>> Greeting*

What’s your name? John

Hello John!

%This script prompts user for name and then prints a greeting with name.

name = input('What''s your name? ', 's');

fprintf('Hello %s!\n', name);

**4**) (10 points) In the metric system, fluid flow is measured in cubic meters per second (m3/s). A cubic foot per second (ft3/s) is equivalent to 0.028 m3/s.

Write a script titled *flowrate* that will prompt the user for flow in cubic meters per second and will print the equivalent flow rate in cubic feet per second. Here is an example of running the script. Your script must produce output in exactly the same *format* as this:

*>> flowrate*

Enter the flow in m^3/s: 15.2

15.200 m^3/s is equivalent to 542.857 ft^3/s.

%{

This script prompts user for flow in cubic meters per second

and prints equivalent flow rate in cubic feet per second.

%}

flow\_m = input('Enter the flow in m^3/s: '); %cubic meters per second

flow\_f = flow\_m/0.028; %convert to cubic feet per second

fprintf('%.3f m^3/s is equivalent to %.3f ft^3/s.\n', flow\_m, flow\_f);

Or could do:

flowrate.m

%{

This script prompts user for fluid flow in cubic meters per second,

uses function convert\_flow.m to find equivalent flow rate in

cubic feet per second, and then prints the result.

%}

flow\_m = input('Enter the flow in m^3/s: '); %cubic meters per second

new\_flow = convert\_flow(flow\_m);

fprintf('%.3f m^3/s is equivalent to %.3f ft^3/s.\n', flow\_m, new\_flow);

convert\_flow.m

function [flow\_f] = convert\_flow(flow\_m)

%{

This function converts fluid flow in cubic meters per second

to flow rate in cubic feet per second.

%}

flow\_f = flow\_m/0.028;

end

**5**) (10 points) Write a script to plot **sin(x)** for x values ranging from 0 to 2π twice in separate Figure Windows:

* In the first plot window, using 10 points in [0, 2π]. Add title 'sin(x) with 10 points'
* In the second plot window, using 100 points in [0, 2π]. Add title 'sin(x) with 100 points'

%{

This script plots sin(x) for x values ranging from 0 to 2\*pi

in two separate figure windows.

%}

x1 = linspace(0, 2\*pi, 10);

x2 = linspace(0, 2\*pi, 100);

figure(1)

plot(x1, sin(x1), '.')

title('sin(x) with 10 points')

figure(2)

plot(x2, sin(x2), '.')

A screenshot of a social media post

Description automatically generatedA picture containing boat

Description automatically generatedtitle('sin(x) with 100 points')

**6**) (10 points) In hydrology, ***hyetographs*** are used to display rainfall intensity during a storm. The intensity could be the amount of rain per hour, recorded every hour for a 24-hour period. Load the attached ‘raindata.dat’ for rain intensity in inches per hour every hour for 24 hours. Use a **bar** chart to display the intensities. Include x- and y-axis labels and a title for the graph.

%{

This bar graph shows rain intensity in inches per hour every

hour for 24 hours.

%}

d = importdata('c:\Users\agp4tq\Documents\MATLAB\raindata.dat');

x = 1:length(d);

bar(x, d)

xlabel('Hour of 24-hr period')

ylabel('Inches of rain')

title('Rainfall intensity')

A screenshot of a social media post

Description automatically generated

**7**) (10 points) A file called “hightemp.dat” was created some time ago which stores, on every line, a year followed by the high temperature at a specific site for each month of that year. For example, the file might look like this:

89 42 49 55 72 63 68 77 82 76 67

90 45 50 56 59 62 68 75 77 75 66

91 44 43 60 60 60 65 69 74 70 70

etc.

As can be seen, only two digits were used for the year (which was common in the last century). Write a script that will read this file into a matrix, create a new matrix which stores the years correctly as 19xx, and then write this to a new file called “y2ktemp.dat”. (Hint: add 1900 to the entire first column of the matrix.) Such a file, for example, would look like this:

1989 42 49 55 72 63 68 77 82 76 67

1990 45 50 56 59 62 68 75 77 75 66

1991 44 43 60 60 60 65 69 74 70 70

etc.

%{

This script uses data from the hightemp.dat file to create a

new matrix which stores the years in the correct format and

writes this matrix to a new file called y2ktemp.dat.

%}

temps = importdata('c:\Users\agp4tq\Documents\MATLAB\hightemp.dat');

modified\_temps = temps;

modified\_temps(:, 1) = 1900 + temps(:, 1);

save('y2ktemp.data', 'modified\_temps', '-ascii');

**8)** (10 points) In chemistry, the pH of an aqueous solution is a measure of its acidity. The pH scale ranges from 0 to 14, inclusive. A solution with a pH of 7 is said to be *neutral*, a solution with a pH greater than 7 is *basic*, and a solution with a pH less than 7 is *acidic*.

Write a script that will prompt the user for the pH of a solution, and will print whether it is neutral, basic, or acidic. If the user enters an invalid pH, an error message will be printed.

%{

This script prompts user for pH of a solution and prints

whether the solution is neutral, basic or acidic.

%}

pH = input('Enter the pH of a solution: ');

if pH >= 0 && pH < 7

disp('acidic')

elseif pH == 7

disp('neutral')

elseif pH > 7 && pH <= 14

disp('basic')

else

disp('Error')

end

**\*9)** (10 points) Rewrite the following **switch** statement as one nested **if-else** statement (**elseif** clauses may be used). Assume that there is a variable *letter* and that it has been initialized.

switch letter

case 'x'

disp('Hello')

case 'y'

disp('Yes')

case 'Q'

disp('Quit')

otherwise

disp('Error')

end

if letter == 'x'

disp('Hello')

else

if letter == 'y'

disp('Yes')

else

if letter == 'Q'

disp('Quit')

else

disp('Error')

end

end

end

Or with elseif clauses:

if letter == 'x'

disp('Hello')

elseif letter == 'y'

disp('Yes')

elseif letter == 'Q'

disp('Quit')

else

disp('Error')

end

**10**) (10 points) Write a script *CalBMI* to calculate a person’s body-mass-index (BMI) using the equation:

Your script should work like the example below:

>>CalBMI

What’s your weight in [lbs]: 170

What’s your height in [in]: 72

Your BMI is 23.1.

>>CalBMI

What’s your weight in [lbs]: -170

BMI cannot be calculated with negative measures.

CalBMI.m

%{

This script calculates a person's body-mass-index using

BMIcalculator function.

%}

weight = input('What''s your weight in [lbs]: ');

height = input('What''s your height in [in]: ');

if weight < 0 || weight < 0

disp('BMI cannot be calculated with negative measures.')

else

BMI = BMIcalculator(height, weight);

fprintf('Your BMI is %.1f.\n', BMI)

end

BMIcalculator.m

function [bmi] = BMIcalculator(height,weight)

%Calculate BMI based on person's height and weight.

bmi = weight/(height\*height) \* 703;

end