Workshop 2 - Programming Practice in MATLAB

- Due 2 Aug by 23:59
- Points 1

Workshop 2 - Programming Practice in MATLAB

This workshop is designed to give you some more practice at programming in MATLAB and working through some key concepts from this week's video content.

If you are sharing a terminal make sure that take turns at working during this practical so every group member has a chance to save some working code.

When you have finished your workshop you should make sure that you show the work that you have done to the laboratory supervisor so your participation can be marked.

Question 1a.

On paper, or in a text file, write three new tests for a program that takes in a width and height of a rectangle as input and displays the area of that rectangle. The tests should be in the form of a table with two columns for input values (*width* and *height*) and one column for the ***expected*** result: *areaRect*. As an example, two tests for this program might be.

```
width      height    areaRect
3          4        12.00
6.0        3.5      21.00
```

Question 1b:

Now we have some tests, write a MATLAB script, called **area.m** that reads in two values, representing the width and height of a rectangle and displays the area of the rectangle. Your program should prompt the user for the input values and print out the area with a message that starts "The area is: ". Run your program with your test data.

As an example a run of your code might look like:

```
enter the width 3.5
enter the height 8
The area is: 28.00
```

Check that your code produces the expected result.

Question 2a.

On paper, or in a text file, in your group, write three new test cases for a new program which reads in the length and width of a rectangle and calculates the total length (perimeter) of the four sides of that rectangle. For reference, the perimeter of a rectangle is is 2 times the length plus 2 times the width of

the rectangle. As before, your test should be written as a table with two columns for input variables and one column for ***expected*** output values. As an example, two tests for this program might be.

```
width      height    perimeterRect
1          1        4.0
2.0        3.0      10.0
```

Question 2b.

If you are sharing a computer, swap a new group member to the terminal. Save your old copy of **area.m** as **perimeter.m**. Now, modify the code so that it calculates the *perimeter* of the rectangle. Run the tests you prepared in 2a and check the results match your expected results.

As an example a run of your code might look like:

```
enter the width 3
enter the height 5
The perimeter is: 16.00
```

Question 2c.

Create a script called **shape_perimeter.m** that prompts the user for the number of sides that a shape has and then prompts for the length of each side and calculates the perimeter (hint: you will need to use a loop for this - as covered in video 7 - Repetition). An example run:

```
enter the number of sides: 3
enter the length of side 1: 4
enter the length of side 2: 12.2
enter the length of side 3: 8.6
The perimeter is: 24.80
```

Again, devise at least 3 different test cases to ensure your code works.

Question 2d.

If you are sharing a computer, swap a new group member to the terminal. Extend your script called **shape_perimeter.m** to also store the lengths and print them out. An example run:

```
enter the number of sides: 3
enter the length of side 1: 4
enter the length of side 2: 12.2
enter the length of side 3: 8.6
The perimeter is: 24.80
The lengths of the 3 sides were: 4.0, 12.2, 8.6
```

Again, devise at least 3 different test cases to ensure your code works.

Question 3a.

For this problem you have to calculate how long your money will last. On paper, or in a text file, with your group, write five new test cases for a program that reads in your *current bank balance* and your

weekly expenditure and prints out the number of weeks you have before the money runs out.

As an example, some two test cases for your program might be:

```
balance expenditure weeksLeft
100     10          10.0
25      10          2.5
```

Question 3b.

Again, if you are sharing a computer, swap the person who is typing. Write a MATLAB script called **weeks.m** which reads in your current bank balance and weekly expenditure and, as described above, prints out the number of weeks you have before your money runs out. Again, run your program with your test data.

As an example a run of your code might look like:

```
Enter your balance: 25
Enter your weekly expenditure: 10
You have 2.5 weeks left before your money runs out.
```

Question 4a.

For this problem you have to read in text representing a person's name and print out a string saying:

Your name is:

Followed by the name that was read. An example run might be:

```
Please enter your name: Trudy
Your name is: Trudy
```

Some example test cases might be:

```
name      output
Eddy      Your name is: Eddy
Susan     Your name is: Susan
```

Write two new test cases for this program.

Question 4b.

If you are sharing a computer, swap the person you are typing and write a MATLAB script called **greeting.m** that reads in a name and prints out the message as specified in question 4a above. For reference, some example MATLAB code to read in and print a formatted text string is:

```
insect=input('name an insect ','s');
fprintf('your insect is a %s\n',insect);
```

(think of what the 's' and %s symbols mean)

Try saving the code above in a script and run. Feel free to modify to answer this question.

Question 5a.

For this problem you have to read text representing:

- A type of animal e.g. cat
- A name for an animal e.g. tibby
- An activity for that animal e.g. running

On paper or in a text file write test cases for a program that reads the input above and prints out text of the format:

I saw your <animal>, <name>, <activity> in my garden.

An example run of this program might be:

```
Enter a type of animal: dog
Enter a name: Biggles
Enter an activity: digging
I saw your dog, Biggles, digging in my garden.
```

Write two new test cases for this program.

Question 5b.

Write a new MATLAB script called animalsVerb.m that reads in text and prints out the messages as specified in 5a. above. Run your program with your test data.

Finishing up.

Make sure you let a practical supervisor see your work for this session. If you have free time at the end of these questions then feel free to extend your code for question 5b into a program that generates more detailed text output (perhaps a formal letter).

End of Workshop.