


Practical 1 - data, variables and operators

- Due 29 Jul by 23:59
- Points 12
- Submitting an external tool

This practical assesses your basic programming skills in MATLAB. After you have completed the assignment upload your ".m" files to this assignment by the due date.

This practical is to be done **individually** you should be able to complete this practical based on the work you have done with the weekly videos, seminar, readings and workshop. You are encouraged to complete the work during your practical session and your tutor can help clarify any concepts you are finding confusing. If you are not able to complete this practical within 2 hours, you will benefit from additional practice. There are additional examples and practice exercises in the textbook and the Peer Assisted Study Sessions (PASS) offers further opportunities for practice (watch course announcements for details on PASS).

CBOK and SFIA mapping

This exercise assesses your skills in the CBOK knowledge area of technology building: Programming and the SFIA Skill of (PROG) - program development. [See the ACS Core Body of Knowledge \(CBOK\) document for reference.](https://www.acs.org.au/content/dam/acs/acs-accreditation/CBoK%20V3.2.pdf) 
(<https://www.acs.org.au/content/dam/acs/acs-accreditation/CBoK%20V3.2.pdf>).

How to work at this Practical - and all Practicals.

Most software is refined through a process of design, coding, testing and refinement. When building even very small programs, to make things go smoothly consider the following process:

1. Quickly break your problems into rough steps to solve. For example if your problem is:

Read the heights of three people and then print a nicely formatted message that displays their average height.

The above might be broken down into the steps:

1. Read in the heights of the three people.
2. Calculate the average height - broken down into:
 - sum the heights and assign to a variable.
 - divide the sum of heights by three and assign to an average variable.
3. Print out the value of the average variable in a nicely formatted message.

These 'steps' will usually be comments in your code, identifying what each section of code's role is in the overall program. Your program should also begin with a comment

stating what the overall program does. In this instance, 'Calculates the average of 3 height inputs'

2. Think about the **ingredients** you are going to need to make the above plan work. Two key ingredients are how to read in input and how to print a nicely formatted output. Write a very short and simple script to see if you can get these things to work. For example, for input, you should write a very small script that just displays a prompt and reads in a single value. If this works **save your script** and put it aside so you can refer to it later.
3. Work on each section of the code in turn - thinking carefully about what you expect it to do, how you can test it, and what you expect its behaviour or output to be. Try to **minimise the time between tests**. This means you should be thinking of how you will write the next piece of code so it can be tested to see if it does what you expected. Write **comments** in your code to help annotate your design and how you expect it to behave. Note, if you can write and test a piece of code separately, it is a good idea to do so (and save a copy in a script of its own). If you have tested a part of the code (e.g. the part of the code to read in three values) make sure you save the script and take a copy to work on - this way if you wreck your old version of the code you will be able to go back to the correct version.
4. Test the whole program with different inputs. Think about valid inputs that may cause problems. Can your code handle these?

Questions

If you have not done so already, make a folder in your U: drive called week1, even if you complete these questions on your own device make sure you put a copy of your MATLAB scripts in this folder. Details of accessing your U: drive on non-university computers can be found at <https://www.adelaide.edu.au/technology/your-tools/academic/adapt>

Before you begin coding, be sure you have read the sections on variable naming conventions and indentation in [MatlabStyle2 book.pdf](#)

<https://myuni.adelaide.edu.au/courses/95033/files/15473409?wrap=1> ↓

https://myuni.adelaide.edu.au/courses/95033/files/15473409/download?download_frd=1

Question 1:

Write a MATLAB script called **question1.m** that does the following:

Read the height, depth, and length of a rectangular-shaped cardboard container and then print a nicely formatted message that displays its volume. Recall the volume of any rectangular-shaped container is the length times the height times the depth.

Note, that you will probably have a number of versions of your MATLAB scripts that you have create as you work towards your final solution. Please call each of these something like **question1-1.m** or **question1InputTest.m** or any reasonable name starting with "question1". We are interested in these files so we can give you feedback on your process. You can submit

these along with your final working version which must be named as specified in each question. Only the final version with the specified name will be tested.

For testing, consider different types of values. What values of height, depth, and length would you want to test to give you confidence that your program is working correctly? Are there any particular values for height, depth, and length that give interesting volumes that you can test for? In a text file, write down the test inputs and expected outputs you used to test your code works. You should include **at least 3** tests. For example, one test might be:

```
test case: height, depth and length all 1

input:
1
1
1
expected output: 1
```

Submit your test cases as **tests1.txt**

Question 2:

Write a MATLAB script called **question2.m** that takes in, as input, the population of a city and makes an estimate about the total daily water use of that city measured in kilolitres. The input is only the total population of the city. If you need other information to estimate the total daily water use of the city you will need to define variables* and set these to appropriate values**.

*Note that you will want to define variables with meaningful names representing things like the number of people per household and the average household water use.

**Note that there is no one correct solution to this program. Look to see what information is available online from real data from published sources for real cities and store that information in variables in your program. Make sure your comments state your assumptions about your estimate.

Make sure that you save and submit all of the files that you created while developing your solution.

In a text file, write down the test inputs and expected outputs you used to test your code works. Submit this as **tests2.txt**. For example, one test case you would want to check is a population of 0. The total water use of such a city should be 0.

```
test case: 0 population
input: 0
expected output: 0
```

Think of at least 3 other cases you want to include to ensure your program is working correctly.

Question 3:

Take your solution to question 2 above and save it in **question3.m** modify your code so that it can estimate the percentage of daily use for a city that can be covered by the desalination plant

(daily capacity 300,000 kilolitres). For example, if the daily water use of the city is 300,000 kilolitres, then you should output: desalination can cover 100% of the daily water use.

In a text file, write down the test inputs and expected outputs you used to test your code works. You should include at least 3 tests. Submit this as **tests3.txt**

Question 4:

Write a MATLAB script called **question4.m** that reads in the population of a city and estimates the total daily cost of transport for those people driving a car. There are multiple ways you could get the other values you need into your program. You could set them as constants or you could choose to ask the user to input them.

Your calculation should include the population. You can make up reasonable numbers for other values that you need.

Note you should build up from a very simple model e.g. start with the assumption that everyone owns a car, drives 10kms a day, and only pays the cost of fuel. You can add details as you go (e.g. including registration and insurance costs, getting statistics from user input: average distance traveled per day, percentage of people owning cars). Each time you modify the script you should save a copy. Each time you test you should have a rough estimate of what you expect the result should be (write this estimate in the comments before you run the script). As before, hand in all of your intermediate files.

In a text file, write down the test inputs and expected outputs you used to test your code works. You should include at least 3 tests. Submit this as **tests4.txt**

Assessment

Marks are awarded for

- **functionality** (the code runs and produces expected results) - 4 marks
- **style** (the code adheres to [MATLAB Style Guidelines](https://myuni.adelaide.edu.au/courses/95033/files/15473409?wrap=1) (<https://myuni.adelaide.edu.au/courses/95033/files/15473409?wrap=1>) ↓ (https://myuni.adelaide.edu.au/courses/95033/files/15473409/download?download_frd=1) for this practical we will be assessing adherence to naming conventions for variables and comments) - 4 marks
- **testing** for this practical we are looking for the presence of at least 3 test cases for each question - 4 marks

This tool needs to be loaded in a new browser window

Load Practical 1 - data, variables and operators in a new window

