Download the supporting files from the relevant section of the VLE and save to your working directory.

Exercise 2 – Using decision making and logic

In this first exercise you'll work on an m-file that contains a few if/else structures and adapt it to add further logic to the program. You will then look at making a simple program that uses the switch/case structure to select some song lyrics to display based on a user input. Finally you'll have the bigger challenge of using the find logic to process a matrix containing some patient data.

If/Elseif/Else

1. Ensure the ifsnippet.m file from the Lab2.zip is in your working directory and remember to set the current directory in the MATLAB window to the same directory.

   a. Open the m-file in Editor and read through it, note how well commented it is, all your code should have the same level of documentation.

   b. Run the program by either typing ifsnippet in the Command Window or pressing the run button in editor. Note in the script, the pause command awaits a key press by the user before continuing.

   c. Does it work as expected? Run it a few times using a range of inputs.

2. You will now look into adding further logic statements to display further messages depending on input value.

   a. Delete everything in the .m file after the first if structure. Now change the first if structure in ifsnippet.m so it displays a message stating the following outcomes if your number is:

      exactly equal to 5
      less than 5
      is more than 5 but less than 10
      or is greater than or equal to 10

   You can do this by building a more complicated structure using if/elseif/else. Clear the if/else structures from ifsnippet.m except for the first one and adapt that structure to complete it all in one go. (QG 8) The new structure should have the following format:

```
if logic
    [code to display message]
elseif further logical expression
    [code to display message]
elseif further logic
    [code to display message]
else
    [code to display message]
end
```

It's worth pointing out that once one of the conditions has been met then the rest of the statements are ignored... so if your number is less than 5 it displays a message and jumps to the end.

b. Run the code and check it's doing what you think it should.

Graduation predictor

3. Create a new .m file in editor called graduation.m. In this program you are going to ask the user what their average mark for the year is, you will then use some if/elseif/else statements to tell the user what class of degree they are currently predicted.

   a. Open a new .m file and save as graduation.m. Add some opening comments containing the program name and a description.

   b. Clear the command window and workspace using ***clc*** and ***clear*** respectively. (QG 0)

   c. Use the ***input*** function to ask the user their current module average is and assign it to a variable *average_mark*. (QG 12.1)

   d. Now you need to create an if structure that has cases for each degree class and use the else case for the fail condition. (QG 8) You will essentially be implementing the table below. Use the ***disp*** function to tell the user their predicted grade i.e. 'You are predicted a third class degree.' (QG 12.2)

Average mark	Degree Classification
Greater than or equal to 70	1st class
Greater than or equal to 60 and less than 70	2(1) class
Greater than or equal to 50 and less than 60	2(2) class
Greater than or equal to 40 and less than 50	3rd class
Less than 40	Fail

   e. Save your code and run it. Does it work as expected? Could you simplify the conditional statements based on the order they are evaluated?

One for the money

4. There is a second way to pick from a list (best with a continuous list e.g. 1, 2,3, 4 etc) using the switch/case structure.

a. Create a new script and save it as the_king.m

b. Add comment at the start of the m-file:
% the_king.m uses the switch case structure to display some the lyrics from Blue Suede Shoes by Elvis Presley

c. Use the input function to request a number between 1 and 3 and assign it to the variable song_line.

d. Use the switch/case structure to display the following messages depending on the number they select:

1 -> One for the money
2 -> Two for the show
3 -> Three to get ready now go cat go

Note you should also include an otherwise statement if none of the cases match. The structure will be of the form (QG 5):

```
switch variable_name
case value
        [code to display string here]
case value
        [code to display string here]
case value
        [code to display string here]
otherwise
        [code to display string here]
end
```

e. Save your program and run it. Could this have been done with if/elseif/else statements?

Find a patient challenge

5. Now for a more involved problem. We'll now look at the find function; it is used to select values in vectors or matrices that agree with a conditional statement, returning the indices of those statements that do. You'll be looking at data for a number of patients to a clinic. As part of a new clinical trial, we need to identify suitable patients.
Patient_data.mat contains a matrix of *patient_data*. Each column in patient data represents a different patient. There are 20 sets of patient data in total.

Each row of data contains pieces of patient information in the following order:

- Patient ID Number
- Age in years
- Height in m
- Weight in kg
- BMI (Body mass index)
- Smoker (0 if no, 1 if yes)

Your job is to create an automated program to identify suitable patients for a clinical trial. In order to be eligible for the trial the patient must be aged between 25 and 40, have a BMI of over 39kg/m^2 and be a smoker.

a. Create a new .m file and save it as patient_recruiter.m, also provide a brief description in the opening comments.

b. Clear the workspace and command window programmatically. (QG 0)

c. Load in patient_data.mat to the workspace. (QG 11.1)

d. We now need to find the patients who meet the first requirement, that is those that are 25 or over and under the age of 40. We will use the find function to look at just the data in the second row of *patient_data* so you'll have to do some indexing too. The find function looks for values that agree with the conditional statement and returns the indices of those that do. (QG 6) Assign the result to a variable *patient_index*. Your code should look something like:

```
patient_index = find(patient_data(i,j) ?? 25 ?? patient_data(i,j) ?? 40);
```

Replace i and j and the question marks with the correct indexing and logical operators respectively. When complete this will return the index of each patient that meets the first requirement. Run the code to check.

e. Now we want to create a new matrix that contains just the data for the eligible patients. To do this we can index *patient_data*, selecting all the rows but just the columns given by *patient_index*. Assign the new matrix to a variable *eligible_patients*.

f. We now want to look at the second criteria, which is a BMI of over 39kg/m2. We can go through the same process as d. but now we want to use the find function on the 5th row of the *eligible_patients* matrix looking for values over 39. Again, assign the output of the find function to the variable *patient_index*.

g. Now update *eligible_patients* to remove the unsuitable patients. Index *eligible_patients*, selecting all the rows but just the columns given by the new *patient_index*. (QG 2.2)

h. Finally we want to check if any of the remaining eligible patients smoke. Again use the find function to check in any values in the 6th row of the *eligible_patients* are 1. There should only be one index value returned, assign it to the variable *chosen_patient*.

i. Finally index the patient number given in row 1 of the data and display a line of test: "The chosen patient ID number is ??." (QG 12.2)

j. As an extra challenge, try saving just this patient's data to a file called recruited_patient.mat. (QG 11.1)

End of exercise 2