

Introduction to Matlab programming

Class 3/4: *Logic and conditional*

The next three exercises in this guide should first be solved **without the computer**. Results can be checked afterwards in Matlab.

1. Logical comparisons. Given the vectors:

$$a = (1 \quad -2 \quad 0 \quad 4), \quad b = (-1 \quad 0 \quad -1 \quad 4),$$

determine what is the resulting vector c when

- | | |
|------------------|--------------------|
| a) $c = a == b;$ | d) $c = a >= b;$ |
| b) $c = a > b;$ | e) $c = a <= b;$ |
| c) $c = a < b;$ | f) $c = a \sim b;$ |

2. Logical operations. Given the vectors:

$$a = (1 \quad 0 \quad 1 \quad 0), \quad b = (1 \quad 0 \quad 0 \quad 1),$$

determine what is the resulting vector c when

- | | |
|----------------------------|-----------------------------------|
| a) $c = a \& b;$ | e) $c = \sim(a \& b);$ |
| b) $c = a b;$ | f) $c = a \sim b;$ |
| c) $c = \sim a;$ | g) $c = \text{any}(a \sim b);$ |
| d) $c = \text{xor}(a, b);$ | h) $c = \text{any}(b \& \sim b);$ |

3. Given the following programs:

a) if $x == 10$ disp('case 1'); end;	b) if $x == 10$ disp('case 1'); else disp('case 2'); end;
c) if $x > 0 \&\& x < 10$ disp('Range 1'); else disp('Out of range'); end;	d) if $x < 5$ disp('range 1'); elseif $x < 10$ disp('range 2'); else disp('range 3'); end;
d) if $x > 0 \parallel x < 10$ disp('Range 1'); else disp('Out of range'); end;	e) if $x < 0 \&\& x > 10$ disp('Range 1'); else disp('Out of range'); end;

What is displayed when

- | | |
|------------|-------------|
| a) $x = 0$ | d) $x = 10$ |
| b) $x = 3$ | e) $x = 12$ |
| c) $x = 8$ | f) $x = -1$ |

The next exercises are to be solved using the Matlab Editor to write scripts or functions:

4. Write a script that asks the user for a number (use the *input* function) and then displays one or more of the following messages after evaluating the number:
 - “This is a positive number”
 - “This is a negative number”
 - “This is prime number”
 - “This is an even number”
 - “This number is divisible only by 2”
 - “This number is divisible only by 3”
 - “This number is divisible by 2 and 3”
 - “This number is not divisible by 2 or 3”

Hints: Use the function *isprime* to determine if the number is prime.

The function *rem(a,b)* returns the remainder after the integer division a/b

5. Transform the previous script in a function with one input argument: the number to be analysed (the *input* function is not needed any more).
6. Logical vectors can be employed to access specific data on a matrix. Create a function with 3 input arguments (*experiment*, *cell*, *channel*) and 1 output argument. The function should automatically open the “TableExample.txt” file (this is a matrix whose columns are [Experiment | Time | Cell | GFP | mCherry | DAPI]) and return a two-columns matrix whose columns are the time and the channel (either GFP, mCherry or DAPI) intensity according to the three input parameters.
 - a) Do it **without** using *if-else-elseif* statements.
 - b) Repeat the exercise but using *if-else-elseif* statements. Which program is shorter?
7. Create a function that accepts a vector as an input argument and returns two variables as output arguments. The function should perform the following tasks:
 - Display the following text lines
 - “Choose an option:”
 - “1) Calculate mean & standard deviation”
 - “2) Calculate max and min”
 - “3) Plot the vector”
 - “4) Plot the histogram”
 - “x) Do nothing”
 - Ask the user for an option (use the *input* function)
 - Evaluate the option and perform the corresponding tasks by using *if-elseif* statements. In case of option 1 and 2, the two output variables should contain the mean and standard deviation or the min and max value respectively. For any other option, the output arguments must be empty.
8. Rewrite the previous function using *switch-case-otherwise* statements instead of *if-elseif* statements.
9. Modify the previous function to accept two input arguments. The second one is optional and corresponds to the desired option. If the user provides this argument, then no display of text lines or question to the user is performed. Use the *nargin* function to evaluate the number of input arguments.