

Introduction to Matlab programming

Class 6: *'While' loops*

1. **Without using a computer**, please predict how many times will the following programs print "Hello World"?:

a) <pre>n = 1; while n < 100 disp(num2str(n)); disp('Hello World'); n = n + 1; end;</pre>	b) <pre>n = 100; while n > 0 disp(num2str(n)); disp('Hello World'); n = n - 2; end;</pre>
c) <pre>n = 1; while n > 1 && n < 10 disp(num2str(n)); disp('Hello World'); n = n + 1; end;</pre>	d) <pre>n = 1; while n > 1 n < 10 disp(num2str(n)); disp('Hello World'); n = n + 1; end;</pre>
f) <pre>n = 0; while n < 100 if n == 50 break; end; disp(num2str(n)); disp('Hello World'); n = n + 2; end;</pre>	g) <pre>n = 0; while n < 100 if n == 50 continue; end; disp(num2str(n)); disp('Hello World'); n = n + 2; end;</pre>

2. Given the following **for** loops, rewrite them using **while** loops. **Please, try to do it without using the computer.**

a) <pre>for a = 1:50 disp('Hello World'); end;</pre>	b) <pre>for a = 20:-1:5 disp(num2str(a)); end;</pre>
c) <pre>a = input('Write a number:'); for n = 1:a if rem(n,5) == 0 disp(num2str(n)); end; end;</pre>	d) <pre>for ii = 1:10 for jj = 1:20 disp(['Row=',num2str(ii)]); disp(['Col',num2str(jj)]); end; end;</pre>

3. Given the following **while** loop, analyze it and then try to rewrite it using **for** loops.

```
a)  n = -1;
    while n <= 0
        n = input('Please, write a positive number: ');
        if ~isnumeric(n)
            n = -1;
            continue;
        end;
    end;
```

4. Write a program that asks the user to respond by 'yes' or 'no'. The program must keep on asking until the user enters the correct information. Remember the functions *strcmp* and *strcmpi* to compare strings.
5. Write a function with a vector *V* as input argument that finds and returns the index to the first negative value.

Your function should return the same result as typing in the command window:

find(V < 0 , 1, 'first')

6. The value $\pi^2/8$ can be approximated by an infinite sum of terms

$$1 + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{9^2} + \dots$$

Create a function to perform and return this approximation, with 2 input arguments:

- a) *TermError*: minimum value of each term in order to be considered in the series. For example, if *TermError* = 0.1, the term $1/3^2 = 0.1111$ will be considered, but $1/5^2 = 0.04$ and the next ones will be omitted.
- b) *MaxIterations*: maximum number of iterations.

Therefore, the loop will stop after reaching either the minimum term error or the maximum number of allowed iterations.

7. Modify the previous script by changing the stopping criterion for the term error. Let the sum of the first *n* terms be *T_n*. As *n* increases the ratio *T_n/T_{n+1}* approaches 1. Now let *TermError* be the maximum allowed value for this ratio, whose values are between 0 and 1. Hint: You need to keep track of the previous and current sum in the loop.