Functions in MATLAB - Further Details - 2

Lecture 11

Dr. Görkem Saygılı

Department of Biomedical Engineering
Ankara University

Introduction to MATLAB, 2017-2018 Spring

Persistent Variable:

Persistent variables are in fact somewhere in the middle between local and global variables.

Persistent variables are defined in functions, they seem to be local, however their value is global in between their function calls.

Persistent Variable Example:

```
function persistent_var()

persistent inc;

if isempty(inc)
    inc = 0;
else
    inc = inc + 1;
end
disp(inc);
```

Persistent Output:

```
>> persistent_var
     0

>> persistent_var
     1

>> persistent_var
     2

>> persistent_var
     3
```

How to Remove Persistent Variables:

Persistent variables ṕersist ín the workspace until:

- re-saving of the function,
- using clear built-in function of MATLAB to remove the variable,
- closing or restarting MATLAB.

Recursion:

Recursion is a programming method that is simply using the function call inside the function itself. This might seem a bit weird but this method allows complex problems in a simple/short manner.

First of all, recursion is not necessarily supported by every programming language but lucily for us, it is supported by MATLAB.

Greatest Common Divisor:

Let's understand the logic behind recursion through an example.

In order to find the greatest common devider (GCD) of two numbers, we can find the GCD of the smaller number and the remainder from the division of larger number by the smaller number, recursively:

$$gcd(a,b) = \begin{cases} a, if b = 0 \\ gcd(b, \text{remainder}(a,b)), \text{ if } a \geq b \ \& \ b > 0 \end{cases}$$

GCD Function:

```
function res = my_gcd(a,b)

if a>b
    if b==0
        res = a;
    else
        res = my_gcd(b, rem(a,b));
    end
else
    if a==0
        res = b;
    else
        res = my_gcd(a, rem(b,a));
    end
end
```

GCD Output:

```
>> my_gcd(10, 2)

ans =

     2

>> my_gcd(10, 6)

ans =

     2

>> my_gcd(14, 6)

ans =

     2

>> my_gcd(15, 6)

ans =

     3
```

Factorial:

Recursion method can also be applied to calculate the factorial:

$$n! = 1 \times 2 \times 3... \times (n-1) \times n$$

This can be also written as:

$$fact(n) = \begin{cases} 1, \text{if } n = 1 \\ n \times \text{fact}(n-1), \text{if } n > 1 \end{cases}$$

Recursive Factorial Function:

```
function res = my_fact(n)

if n < 1
    disp('input should be larger or equal to 1');
    return;
end

if n==1
    res = 1;
else
    res = n*my_fact(n-1);
end
```

Recursive Factorial Function Output:

```
>> my_fact(5)

ans =

   120

>> my_fact(6)

ans =

   720
```