Matlab 函数

Matlab 函数用法

在脚本文件中编程技巧：

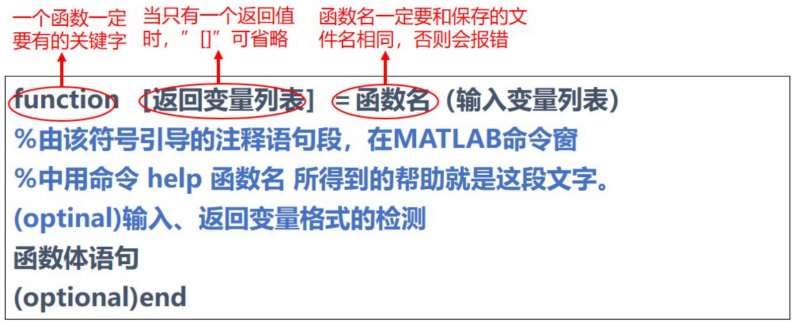
- 在脚本程序开始前都使用
  - — clear, clear all: 删除工作区中的所有储存变量
  - — clc: 清除命令行窗口内的所有内容
  - — close all:关闭所有绘图窗口
- 在命令行结尾使用“；”来避免显示中间结果。
  注意：流控制语句后没有
- 当程序进入死循环时，用“ctrl+c”强行终止运行

✓ 确保工作区没有其他变量干扰，同时回避中间变量显示

函数的基本定义

函数的基本定义：

- 函数是由 function 语句引导的，其基本格式如下




```
myFunction(1);
```

函数举例：

- 求1到10, 3到10, 7到10之前所有数字之和

```
start_num =[1 3 7];
end_num = 10;
for i=1:length(start_num)
    final_sum(i) = Cal_sum(start_num(i)
    , end_num);
end
```

```
>> disp(final_sum)
    55    52    34
```

```
function sum = Cal_sum(s_num,e_num)
% This function is used for calculating
    the summation of a range of
    numbers
 sum=0;
 for i=s_num : e_num
     sum=sum+i;
 end
end
```

收藏 不懂

✓ 函数被调用三次只占用一行代码

```
function sum = Cal_sum(s_num,e_num)
↑
错误：此上下文中不支持函数定义。函数只能作为代码文件中的局部函数或嵌套函数创建。
```

✓ 注意函数的定义只能在脚本文件中进行

7

```
% 这样用矩阵(i) -- 当参数 -- 很方便
start_num = [1,3,7];
end_num = 10;
for i = 1:length(start_num)
    final_sum(i) = Cal_sum(start_num(i),end_num);
end
disp(final_sum);
```

```
    55    52    34
```

变量的介绍：

- MATLAB工作区中的变量和函数内部的变量都是局部变量，且互不干扰
  - — 工作区中的变量需要通过参数传递进入函数进行计算
  - — 函数中的变量如不通过输出传递，则在函数执行完后不显示在工作区中
- 通过 “global variable”,可将variable定义为全局变量
  - —需在工作区和函数内部同时声明该变量为全局变量
  - —为保证函数的独立性，一般情况下不建议使用全局变量

✓ 这里体现了封装的概念，规避了同名变量间潜在的干扰

✓ 不同函数的作用域范围也得以限定

9

```
% eg1
global A;
A = 3;
Test();
```

```
    3
```

```
% eg2
```

```
% 全局变量需要在工作区和函数内部同时声明！
global p;
p = 0;
for i =1:5
    res = global_p(0); % 调用函数
end
p
```

```
p = 5
```

```
% eg3
% 请编写一个 fun_xy.m 函数使其没有返回值，完成以下的计算，当(x,y) = (2,3)
```

$$f(x,y) = \begin{cases} x+y & x \geq 0 \ and \ y \geq 0 \\ x+y^2 & x \geq 0 \ and \ y < 0 \\ x^2+y & x < 0 \ and \ y \geq 0 \\ x^2+y^2 & x < 0 \ and \ y < 0 \end{cases}$$

```
f = fun_xy(2,3);
disp(f);
```

```
    5
```

```
f = fun_xy(5,-1);
disp(f);
```

```
    6
```

调试与其它

调试：

- 设置断点，运行命令时程序会自动停留在断点处
  - — 可以设置多个断点
  - — 可以从断点处逐行运行(F10)，也可调至下一个断点处(F5)
- 调试状态下，命令窗的命令提示符变为 “K>>” 。这时可输入任何MATLAB命令，变量为函数内部的局部变量，而不是工作区中的变量
- 调试状态下，鼠标指针指向想查询的局部变量上将立即能显示出该变量值或者查看工作区，直接输入该变量名字也可展示其值

- 设置断点，运行命令时程序会自动停留在断点处
  - — 可以设置多个断点
  - — 可以从断点处逐行运行(F10)，也可调至下一个断点处(F5)
- 调试状态下，命令窗的命令提示符变为 “K>>” 。这时可输入任何MATLAB命令，变量为函数内部的局部变量，而不是工作区中的变量
- 调试状态下，鼠标指针指向想查询的局部变量上将立即能显示出该变量值或者查看工作区，直接输入该变量名字也可展示其值

- MATLAB常见的报错语句
  - — Index must be a positive integer or logical.
  - — Undefined function or variable “B”.
  - — Inner matrix dimensions must agree.
  - — In an assignment A(I) = B, the number of elements in B and I must be the same.
  - — Expression or statement is incorrect--possibly unbalanced (, {, or [.
  - — Too many input arguments.
  - — .....

✓ 展示信息与出错位置，方便进行查找和错误修改

✓ 语法错误容易发现，但是逻辑错误编译器基本无法发现！

18

匿名函数

- MATLAB间接调用函数的数据类型

可传递给其它函数，以便该函数句柄所代表的函数可以被调用；
也可以被储存起来，以便以后利用

✓ 函数句柄可以用符号@后面加上函数名进行表示

```
>> f1 = @sin

f1 =

  包含以下值的 function_handle:

    @sin
```

定义一个函数句柄

```
>> t = 0:pi/6:pi

t =

         0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416
```

```
>> f1(t)

ans =

         0    0.5000    0.8660    1.0000    0.8660    0.5000    0.0000
```

特别是当函数名字较为复杂时，定义句柄是一种可选方法

```
f1 = @sin;
t = 0:pi/6:pi;
disp(t);
```

```
  列 1 至 3

         0    0.5236    1.0472

  列 4 至 6

    1.5708    2.0944    2.6180

  列 7

    3.1416
```

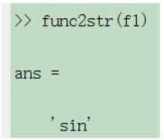
```
f1(t)
```

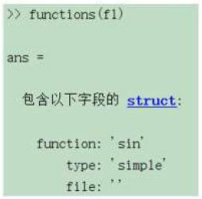
```
ans = 1×7
         0    0.5000    0.8660    1.0000    0.8660    0.5000    0.0000
```

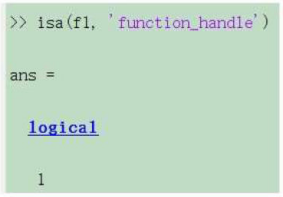
函数句柄相关常用函数

- 函数句柄相关常用函数

函数	说明
func2str(fhandle)	将函数句柄转换为字符串
str2func(str)	将字符串转换为函数句柄
functions(fhandle)	返回包含函数信息的结构体变量
isa(a, 'function_handle')	判断是否为函数句柄
isequal(fhandle1, fhandle2)	检测两个函数句柄是否对应同一函数








```
func2str(f1)
```

```
ans =
'sin'
```

```
functions(f1)
```

```
ans = 包含以下字段的 struct:
    function: 'sin'
        type: 'simple'
        file: ''
```

函数的书写规范

```
edit(which('mean.m'));
```

- "%% XYZ"，注释用于说明模块功能
- "% XYZ"，注释用于该命令功能，一般在命令后方。若太长，可放置该命令上方
- 太简单的命令可以不注释, e.g.,

  Var = 2; % make the variable Var equal 2

运行效率问题

```
% 一些建议
% 1.提前预留变量的内存空间
% 2.在不影响运算时，尽可能将大循环放在嵌套循环内部
% 3.尽量避免使用循环结构

% 3.eg
```

```
tic;
i = 0;
for n = 0:10000000
    i = i + 1;
    y(i) = cos(n);
end
toc;
```

历时 1.082540 秒。

```
tic;
n = 0:1:10000000;
y = cos(n);
toc;
```

历时 0.382431 秒。

```
% 更应该选择后者避免循环

% 4.尽量使用 matlab 内置函数
```

```
function output = myFunction(input)% function 1
    switch input
        case -1
            output = 'nagative one';
        case 0
            output = 'zero';
        case 1
            output = 'positive one';
        otherwise
            output = 'other value';
    end
end


% 计算 1~10，3~10，7~10 之前所有数字之和
function sum = Cal_sum(s_num,e_num)
    sum = 0;
    for i = s_num:e_num
        sum = sum+i; % 这里的 i 和外面的 i 是互不影响的！
    end
end

% 关于变量的介绍
function a = Test()
    global A;
    disp(A); % 如果直接用 A -- A 是无法识别的
end

%
function y = global_p(y)
```

```
    global p;
    p = p+1;
end

%
% A&B：首先判断 A 的逻辑值，然后判断 B 的值，然后进行逻辑与的计算。
% A&&B：首先判断 A 的逻辑值，如果 A 的值为假，就可以判断整个表达式的值为假，就不需要再判断 B 的值。
function f = fun_xy(x,y)
    if(x>=0)&&(y>=0)
        f = x+y;
    elseif(x>=0)&&(y<0)
        f = x+y^2;
    elseif(x<0)&&(y>=0)
        f = x^2+y;
    else
        f = x^2+y^y;
    end
end
```