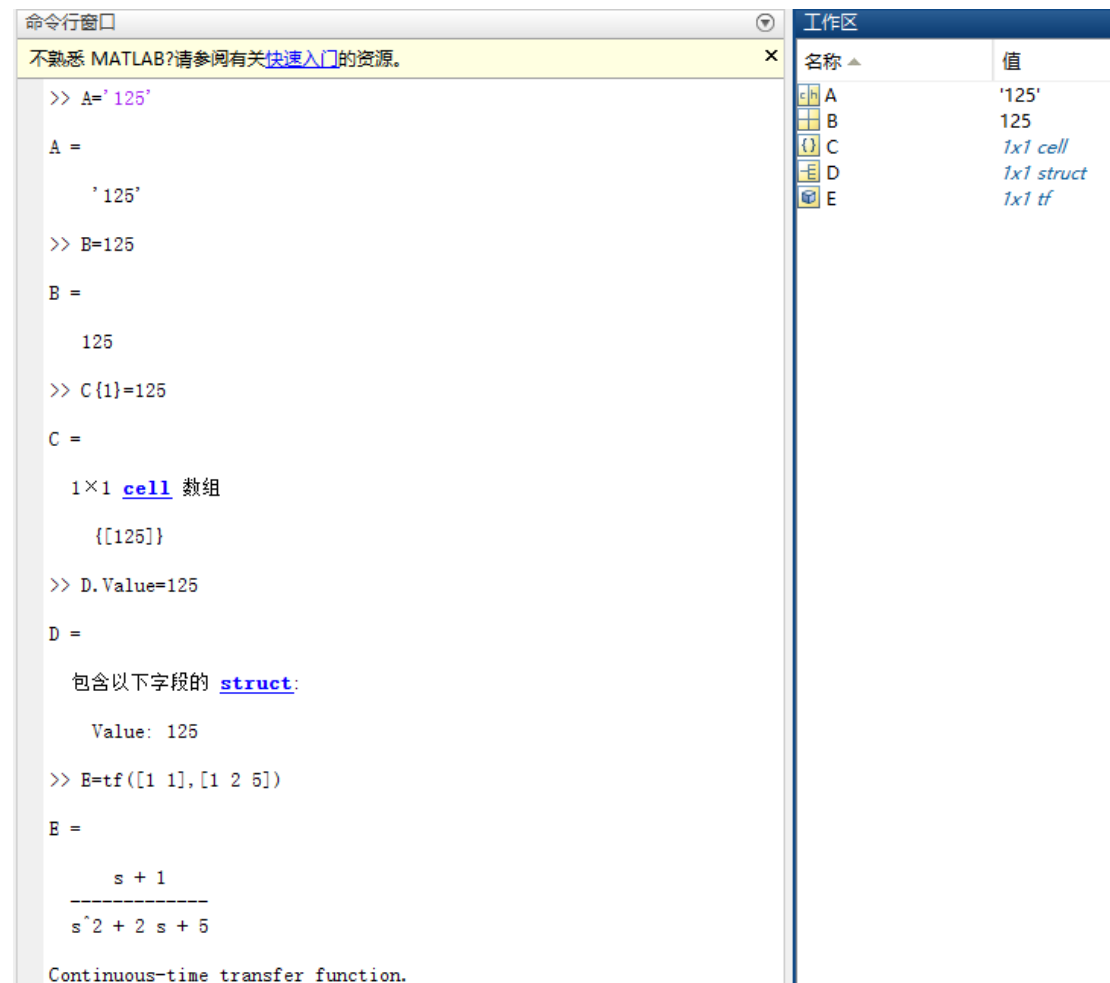


## 一、 数据类型的练习：

A 是字符型数据，值为'125'；B 是数值型数据，值为 125；C 是 1×1 元胞数组，值为[125]；D 是 1×1 的结构体，含一个域 Value，域值为 125；E 是连续系统传递函数，分子系数为 1、1，分母系数为 1、2、5。



The screenshot shows the MATLAB Command Window and Workspace. The Command Window displays the following commands and their outputs:

```
>> A='125'
A =
    '125'

>> B=125
B =
    125

>> C{1}=125
C =
    1×1 cell 数组
    {[125]}

>> D.Value=125
D =
    包含以下字段的 struct:
    Value: 125

>> E=tf([1 1],[1 2 5])
E =
    s + 1
    -----
    s^2 + 2 s + 5
Continuous-time transfer function.
```

The Workspace window shows the following variables and their values:

| 名称 | 值          |
|----|------------|
| A  | '125'      |
| B  | 125        |
| C  | 1×1 cell   |
| D  | 1×1 struct |
| E  | 1×1 tf     |

## 二、 数组和矩阵的生成、扩展和引用

1. A(起始下标:间隔:终止下标)。A(1:3:end)表示从第一个开始、间隔为+3、直到最后一个，引用对应下标的数值组成新的数组；A(end:-1:1)表示从最后一个开始、间隔为-1、直到第一个，引用对应下标的数值组成新的数组；A(3)=[]表示删除第三个数据；A=[A;A]表示扩展 A 矩阵为两个 A 行矩阵的并联（纵向扩展）。

```

>> A=rand(1,5)

A =

    0.0975    0.2785    0.5469    0.9575    0.9649

>> A(1:3:end)

ans =

    0.0975    0.9575

>> A(end:-1:1)

ans =

    0.9649    0.9575    0.5469    0.2785    0.0975

>> A(3)=[]

A =
|
    0.0975    0.2785    0.9575    0.9649

>> A=[A;A]

A =

    0.0975    0.2785    0.9575    0.9649
    0.0975    0.2785    0.9575    0.9649

```

2.  $C=[A\ B]$ 运行结果错误，说明 A、B 分别为  $1 \times 5$ 、 $5 \times 1$  矩阵，维度不一致无法串联； $C=[A'\ B]$ 运行结果为 A 转置后与 B 以列扩展的形式并联。

```

>> C=[A' B]

C =|
    0.1576    0.1419
    0.9706    0.4218
    0.9572    0.9157
    0.4854    0.7922
    0.8003    0.9595

>> C=[A B]
错误使用 horzcat
要串联的数组的维度不一致。

```

3.  $D(:,3)=[]$ 表示删除 D 矩阵第三列所有元素； $D(2,2)=[]$ 运行错误，提示空赋值只能具有一个非冒号索引；结合上面  $A(3)=[]$ 的运行结果，说明**赋空值只能消**

除整行或整列的元素，无法消除单个元素，若执行 `D(2)=[]` 会消除第 2 行第 1 列的元素，但矩阵维度也会被改变成单行矩阵。

```
D =
    0.1419    0.7922    0.0357
    0.4218    0.9595    0.8491
    0.9157    0.6557    0.9340

>> D(:,3)=[]

D =
    0.1419    0.7922
    0.4218    0.9595
    0.9157    0.6557

>> D(2,2)=[]
空赋值只能具有一个非冒号索引。
```

```
>> D=rand(3,3)
D =
    0.0462    0.6948    0.0344
    0.0971    0.3171    0.4387
    0.8235    0.9502    0.3816

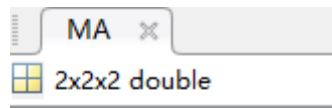
>> D(2)=[]
D =
1 至 6 列
    0.0462    0.8235    0.6948    0.3171    0.9502    0.0344
7 至 8 列
    0.4387    0.3816
```

#### 4. 多维数组练习：

##### (1) 命令如下：

```
MA=rand(2,2,2);
MA(:,:,1)=rand(2,2);
MA(:,:,2)=zeros(2,2);
```

结果：



```
val(:,:,1) =
    0.2238    0.2551
    0.7513    0.5060

val(:,:,2) =
    0     0
    0     0
```

(2) 第一组结果：无法进行赋值；第二组结果：生成  $3 \times 3 \times 2$  的多维数组；如下图：

```
>> MM(:, :, 1)=randn(3, 3);MM(:, :, 2)=rand(2, 2);
无法执行赋值，因为左侧的大小为 3-by-3，右侧的大小为 2-by-2。
```

```
>> MM(:, :, 1)=randn(3, 3);MM(:, :, 2)=rand(3, 3)
```

```
MM(:, :, 1) =
```

```
    0.4900    -0.1941     1.3546
    0.7394    -2.1384    -1.0722
    1.7119    -0.8396     0.9610
```

```
MM(:, :, 2) =
```

```
    0.5383     0.4427     0.0046
    0.9961     0.1067     0.7749
    0.0782     0.9619     0.8173
```

因为生成随机数矩阵的大小要和所赋值矩阵的大小相对应，否则无法执行。

(3) 创建了  $1 \times 2 \times 3 \times 4 \times 5 \times 6$  的六维元胞数组，其(1,2,3,4,5,6)处的元素为  $2 \times 2$  随机数组，元胞数组其余处为空。其他创建六维数组方法如下：

MMM2=zeros(1,2,3,4,5,6)，创建  $1 \times 2 \times 3 \times 4 \times 5 \times 6$  的六维数组，全部为 0 元素；

MMM2=rand(1,2,3,4,5,6)，创建  $1 \times 2 \times 3 \times 4 \times 5 \times 6$  的六维数组，全部为随机元素。

5. 执行 save Excise1 A C MM 命令保存 A、C、MM 变量到 Excise1.mat 文件中；

6. 执行 load Excise1 加载 A、C、MM 变量到工作区。

### 三、 取整和取余运算练习

| 命令\SS       | -0.1 | -0.6 | 1.5 | -1.5 |
|-------------|------|------|-----|------|
| A=ceil(SS)  | 0    | 0    | 2   | -1   |
| B=floor(SS) | -1   | -1   | 1   | -2   |
| C=fix(SS)   | 0    | 0    | 1   | -1   |
| D=round(SS) | 0    | -1   | 2   | -2   |

**ceil:** 向上 ( $+\infty$ ) 取整；**floor:** 向下 ( $-\infty$ ) 取整；**fix:** 向中间 (0) 取整；**round:** 向靠近整数取整 (四舍五入)。

```
>> SS

SS =

    -0.1000    -0.6000     1.5000    -1.5000

>> M=mod(SS,2)

M =

     1.9000     1.4000     1.5000     0.5000

>> R=rem(SS,2)

R =

    -0.1000    -0.6000     1.5000    -1.5000
```

**rem(x,y)**命令表示求余数，返回的是  $x-n.*y$ ，若  $y \neq 0$ ，其中的  $n=\text{fix}(x./y)$ ；  
**mod(x,y)**命令表示模除求余，返回的是  $x-n.*y$ ，若  $y \neq 0$ ， $n=\text{floor}(x./y)$ 。

#### 四、 查找和排序的练习

%% 查找和排序的练习

```
clc,clear;
```

```
SS=rand(101,1);
```

```
[SC,Pos]=sort(SS,'descend'); %按降序排列
```

```
SC_mid=SC((1+101)/2); %返回排序后中间值
```

```
SC_mid_Pos=Pos((1+101)/2); %返回该值在SS数组中的位置
```

```
Index=find(~(SS-SC_mid)); %find找非0值，所以取反使用，找0值，也返回该值在SS数组中的位置
```

结果如下：

|              |  |        |
|--------------|--|--------|
| SC_mid =     |  | 0.4468 |
| SC_mid_Pos = |  | 94     |
| Index =      |  | 94     |

| SC |        |
|----|--------|
| 43 | 0.5132 |
| 44 | 0.5108 |
| 45 | 0.5085 |
| 46 | 0.4909 |
| 47 | 0.4893 |
| 48 | 0.4868 |
| 49 | 0.4509 |
| 50 | 0.4505 |
| 51 | 0.4468 |
| 52 | 0.4427 |
| 53 | 0.4359 |
| 54 | 0.4314 |
| 55 | 0.4173 |
| 56 | 0.4039 |
| 57 | 0.4018 |
| 58 | 0.3998 |
| 59 | 0.3897 |
| 60 | 0.3786 |

## 五、 结构体练习

### 1. 命令如下:

```
S_exce(:, :, 1)=[struct('Name', 'Glaz', 'Weight', 79, 'BloodType', 'A'), ...
    struct('Name', 'Fuze', 'Weight', 80, 'BloodType', 'A'); ...
    struct('Name', 'Kapkan', 'Weight', 80, 'BloodType', 'O'), ...
    struct('Name', 'Tachanka', 'Weight', 70, 'BloodType', 'B')];
S_exce(:, :, 2)=[struct('Name', 'Ash', 'Weight', 63, 'BloodType', 'B'), ...
    struct('Name', 'Thermite', 'Weight', 80, 'BloodType', 'O'); ...
    struct('Name', 'Castle', 'Weight', 86, 'BloodType', 'A'), ...
    struct('Name', 'Pulse', 'Weight', 85, 'BloodType', 'AB')];
fieldnames(S_exce)
getfield(S_exce(1, 2, 1), 'Name')
S_exce(1, 2, 1)=setfield(S_exce(1, 2, 1), 'Name', 'Hostage Killer')
getfield(S_exce(1, 2, 1), 'Name')
getfield(S_exce(1, 1, 2), 'Name')
S_exce(1, 1, 2)=setfield(S_exce(1, 1, 2), 'Name', 'Fivespeed Headless Monster')
getfield(S_exce(1, 1, 2), 'Name')
Russian_bomb=rmfield(S_exce(1, 2, 1), 'Name')
```

### 2. 结果如下:

ans =

3×1 cell 数组

```
{'Name'      }
{'Weight'    }
{'BloodType'}
```

ans =

'Fuze'

S\_exce =

包含以下字段的 2×2×2 struct 数组:

```
Name
Weight
BloodType
```

ans =

'Hostage Killer'

ans =

'Ash'

S\_exce =

包含以下字段的 2×2×2 struct 数组:

Name  
Weight  
BloodType

ans =

'Fivespeed Headless Monster'

Russian\_bomb =

包含以下字段的 struct:

Weight: 80  
BloodType: 'A'

fieldnames 命令: 给出结构体的所有域名; getfield 命令: 得到对应域名的域值;  
setfield 命令: 设置对应域名的域值; rmfield 命令: 去除对应的域

**注:** 执行 setfield 命令改变原结构体域值时需要再进行赋值操作, 否则只是 ans 的结果 setfield 了; 执行 rmfield 命令无法赋值回原结构体, 会显示在不同结构体之间进行下标赋值, 因为已经删除了一个域, 所以与原结构体的域不同, 要想使用 rmfield 命令得到的结果, 需要赋给新的结构体。

六、 单元/元胞数组练习

代码如下:

```
Cell_exce(:, :, 1) = {'赵敏琨', S_exce; '2018302068', []};  
Cell_exce(:, :, 2) = {'赵敏琨', S_exce; '2018302068', []};  
Cell_exce
```

```

S=0;
for n=1:2
    for k=1:2
        for l=1:2
            for m=1:2
                S=S+getfield(Cell_exce{1,2,n}(k,l,m),'Weight');
            end
        end
    end
end
M=S/(n*k*l*m)

```

运行结果如下：

2×2×2 cell 数组

Cell\_exce(:,:,1) =

```

    {'赵敏琨'      }    {2×2×2 struct}
    {'2018302068'}    {0×0    double}

```

Cell\_exce(:,:,2) =

```

    {'赵敏琨'      }    {2×2×2 struct}
    {'2018302068'}    {0×0    double}

```

M =

77.8750

## 七、 字符串的创建和使用

1. 结果：str 是字符型，A 是数值型，B 是字符型  
str2num 命令，字符转数字；num2str 命令，数字转字符；
2. 代码如下：（用 eval 命令将字符串转换成命令行）

```

for index=0:2:100
    eval(['A' num2str(index) '=(sqrt(index+1)-5)^2;'])
end

```

## 八、 矩阵运算和矩阵函数

1. reshape 命令表示矩阵重组（按列遍历，比如原 B 矩阵第二列第一行是第三个），矩阵运算遵循矩阵行列规则，点运算只是每个元素运算结果以矩阵形式显示



```
>> C=A+A'
```

```
C =
```

```
    0.9418    1.0748
```

```
    1.0748    0.3895
```

```
>> E=A*B
```

```
E =
```

```
    0.2505    0.4751    0.9261
```

```
    0.0853    0.1373    0.2515
```

```
>> D=reshape(B,[3,2])
```

```
D =
```

```
    0.2259    0.4357
```

```
    0.1707    0.3111
```

```
    0.2277    0.9234
```

```
>> F=E*B
```

```
错误使用 *  
用于矩阵乘法的维度不正确。请检查并确保第一个矩阵中的列数与第二个矩阵中的行数匹配。要  
执行按元素相乘，请使用 '.*'。
```

```
>> F=E.*B
```

```
F =
```

```
    0.0566    0.1082    0.2881
```

```
    0.0146    0.0598    0.2323
```

2. **det** 命令：计算矩阵的行列式；**eig** 命令：计算矩阵的特征值（结果矩阵的对角元）和特征向量（另一结果矩阵的列向量）；**norm** 命令：计算矩阵的范数；**inv** 命令：计算矩阵的逆矩阵；**rank** 命令：计算矩阵的秩。

```
>> A=rand(2,2)
```

```
A =
```

```
    0.4302    0.9049
```

```
    0.1848    0.9797
```

```
>> [V,L]=eig(A)
```

```
V =
```

```
   -0.9722   -0.7626
```

```
    0.2341   -0.6468
```

```
L =
```

```
    0.2123         0
```

```
         0    1.1977
```

## 九、 数据文件的读写

在导入纯数值型数据时，**load** 命令和 **importdata** 命令效果一样；但若数据中存在非数值型，**load** 命令无法使用，**importdata** 命令运行结果会生成结构体，包含纯数值的矩阵和包含字符的元胞矩阵。下面举例说明：

1. 导入纯数值型数据运行结果：

```
>> Data=importdata('grades.txt')
```

```
Data =
```

|         |         |         |         |         |         |         |
|---------|---------|---------|---------|---------|---------|---------|
| 95.0100 | 76.2100 | 61.5400 | 40.5700 | 5.7900  | 20.2800 | 1.5300  |
| 23.1100 | 45.6500 | 79.1900 | 93.5500 | 35.2900 | 19.8700 | 74.6800 |
| 60.6800 | 1.8500  | 92.1800 | 91.6900 | 81.3200 | 60.3800 | 44.5100 |
| 48.6000 | 82.1400 | 73.8200 | 41.0300 | 0.9900  | 27.2200 | 93.1800 |
| 89.1300 | 44.4700 | 17.6300 | 89.3600 | 13.8900 | 19.8800 | 46.6000 |

```
>> Data=load('grades.txt')
```

```
Data =
```

|         |         |         |         |         |         |         |
|---------|---------|---------|---------|---------|---------|---------|
| 95.0100 | 76.2100 | 61.5400 | 40.5700 | 5.7900  | 20.2800 | 1.5300  |
| 23.1100 | 45.6500 | 79.1900 | 93.5500 | 35.2900 | 19.8700 | 74.6800 |
| 60.6800 | 1.8500  | 92.1800 | 91.6900 | 81.3200 | 60.3800 | 44.5100 |
| 48.6000 | 82.1400 | 73.8200 | 41.0300 | 0.9900  | 27.2200 | 93.1800 |
| 89.1300 | 44.4700 | 17.6300 | 89.3600 | 13.8900 | 19.8800 | 46.6000 |

2. 导入非纯数值型数据运行结果:

| Data.data |         |         |         |         |         |         |         |  |
|-----------|---------|---------|---------|---------|---------|---------|---------|--|
| Data.data |         |         |         |         |         |         |         |  |
|           | 1       | 2       | 3       | 4       | 5       | 6       | 7       |  |
| 1         | 95.0100 | 76.2100 | 61.5400 | 40.5700 | 5.7900  | 20.2800 | 1.5300  |  |
| 2         | 23.1100 | 45.6500 | 79.1900 | 93.5500 | 35.2900 | 19.8700 | 74.6800 |  |
| 3         | 60.6800 | 1.8500  | 92.1800 | 91.6900 | 81.3200 | 60.3800 | 44.5100 |  |
| 4         | 48.6000 | 82.1400 | 73.8200 | 41.0300 | 0.9900  | 27.2200 | 93.1800 |  |
| 5         | 89.1300 | 44.4700 | 17.6300 | 89.3600 | 13.8900 | 19.8800 | 46.6000 |  |
| 6         |         |         |         |         |         |         |         |  |

| Data          |             |   |   |   |   |   |   |   |
|---------------|-------------|---|---|---|---|---|---|---|
| Data.textdata |             |   |   |   |   |   |   |   |
|               | 1           | 2 | 3 | 4 | 5 | 6 | 7 |   |
| 1             | class\group | A | B | C | D | E | F | G |
| 2             | '1'         |   |   |   |   |   |   |   |
| 3             | '2'         |   |   |   |   |   |   |   |
| 4             | '3'         |   |   |   |   |   |   |   |
| 5             | '4'         |   |   |   |   |   |   |   |
| 6             | '5'         |   |   |   |   |   |   |   |

#### 命令行窗口

```
>> Data=load('grades.txt')
错误使用 load
ASCII 文件 grades.txt 的行号 1 中的文本未知
"class\group"。
```

```
>> Data=importdata('grades.txt')
```

Data =

包含以下字段的 **struct**:

```
data: [5×7 double]
textdata: {6×8 cell}
```