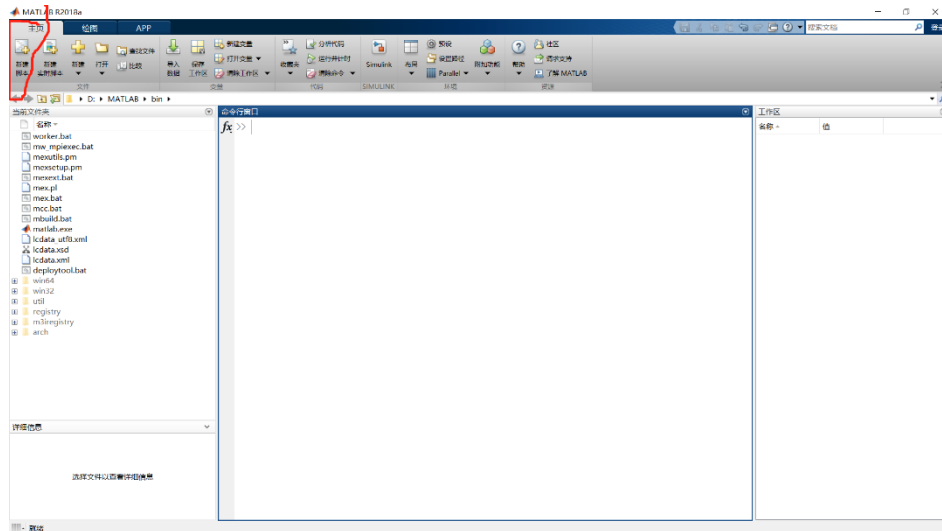
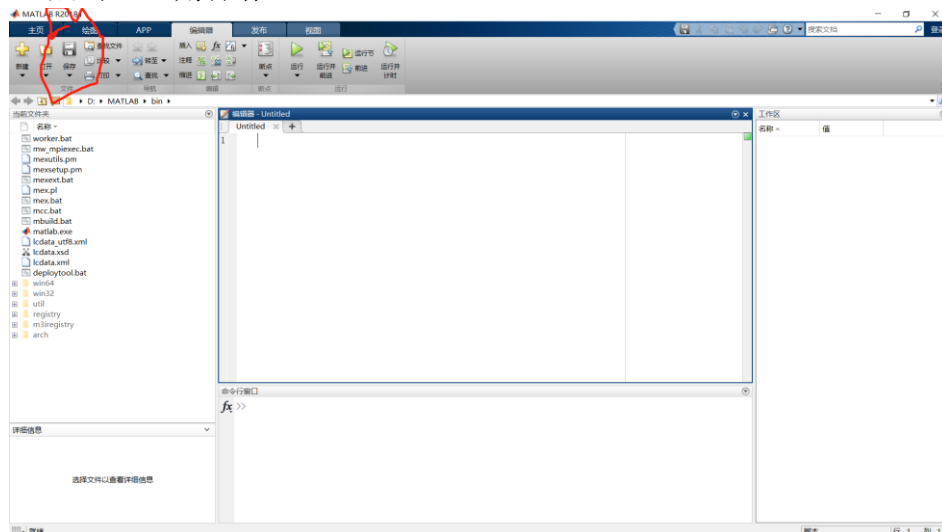


## 一. m 文件的建立

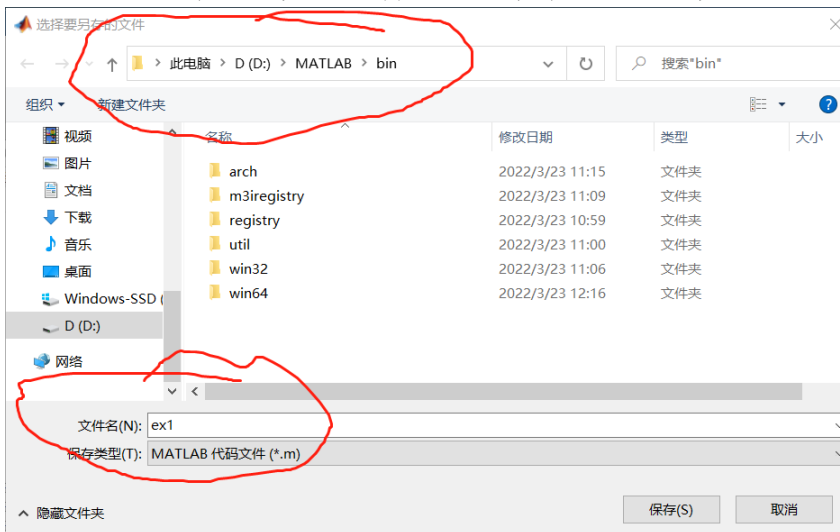
### 1. 打开 matlab，点击左上角新建脚本



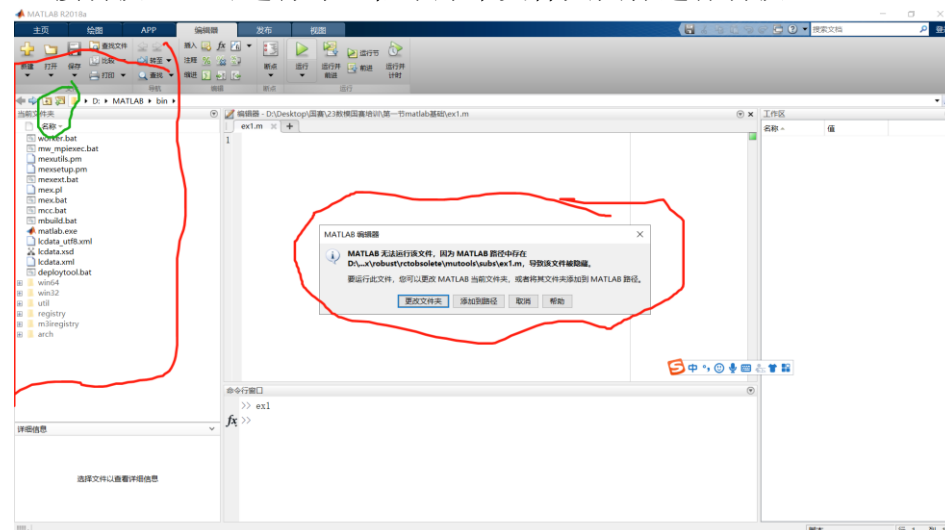
### 2. 点击左上角保存



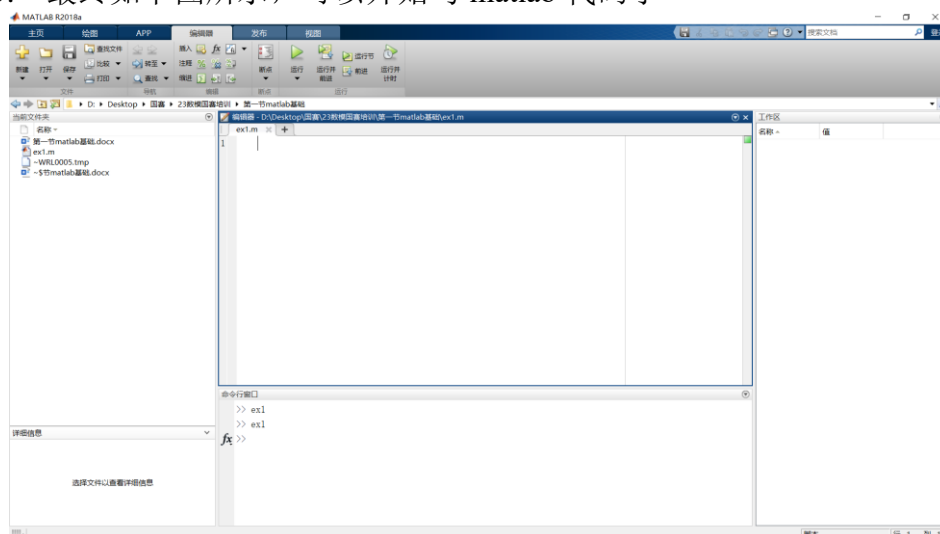
3. 修改文件名和保存路径（默认保存着 bin 中，建议更改保存路径，文件名建议以英文字母开头，不含中文，数字不能放开头）



4. 运行时当前文件夹必须是该 m 文件所在的文件夹，可点击更改文件夹直接切换，也可选择绿色框中那个文件夹图标进行切换



5. 最终如下图所示，可以开始写 matlab 代码了



## 二. 矩阵（向量）的建立

1. 建立矩阵  $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

matlab 代码如下

```
1 - A=[1,2,3;4,5,6;7,8,9]
```

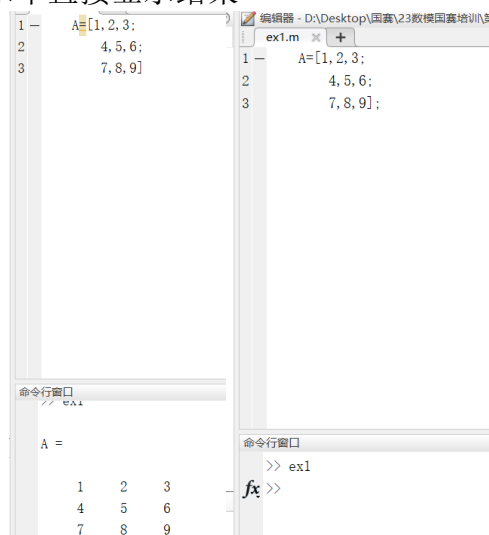
矩阵中的数置于“[]”中，相邻的数用逗号‘,’或者空格‘ ’隔开，矩阵换行时，采用分号‘;’隔开，以上所有符号都采用英文输入法，最终结果如下

```
A =  
  
     1     2     3  
     4     5     6  
     7     8     9
```

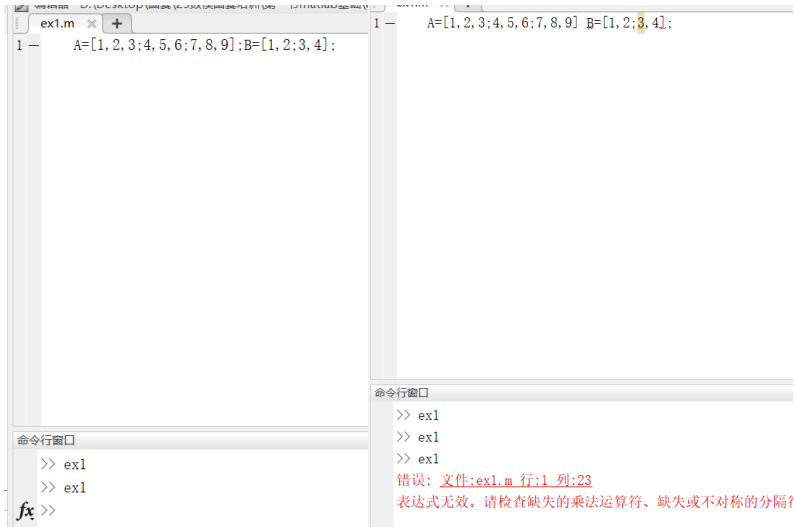
为了便于将代码中的矩阵和实际矩阵对照，一般可以在分号处换行隔开，写成如下形式，结果相同

```
x1.m +  
A=[1,2,3;  
    4,5,6;  
    7,8,9]
```

matlab 代码结尾的分号‘;’可加可不加，不加分号，则表示直接在窗口显示结果，加了分号，则表示不直接显示结果

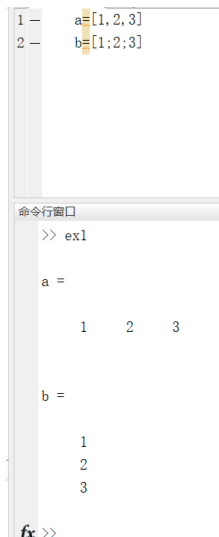


如果加了分号，下一句代码无需换行可直接写，反之，则会报错



The image shows a MATLAB editor window with a file named 'ex1.m'. The code in the editor is: `A=[1, 2, 3;4, 5, 6;7, 8, 9];B=[1, 2;3, 4];`. The command window shows the following commands and output:   
`>> ex1`  
`>> ex1`  
`>> ex1`  
The output for the third command is an error message: `错误: 文件:ex1.m 行:1 列:23`  
`表达式无效。请检查缺失的乘法运算符、缺失或不对称的分隔符`

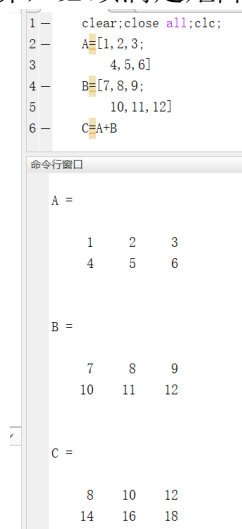
行向量和列向量的建立，行向量用逗号隔开，列向量用分号隔开



The image shows a MATLAB editor window with the following code: `a=[1, 2, 3]` and `b=[1;2;3]`. The command window shows the following commands and output:   
`>> ex1`  
`a =`  
`1 2 3`  
`b =`  
`1`  
`2`  
`3`

## 2. 矩阵的运算

矩阵（向量）的加减法运算，必须满足矩阵（向量）维数相同



The image shows a MATLAB editor window with the following code: `clear;close all;clc;`, `A=[1, 2, 3;`, `4, 5, 6]`, `B=[7, 8, 9;`, `10, 11, 12]`, and `C=A+B`. The command window shows the following commands and output:   
`A =`  
`1 2 3`  
`4 5 6`  
`B =`  
`7 8 9`  
`10 11 12`  
`C =`  
`8 10 12`  
`14 16 18`

写代码前最好第一行写 `clear`（清空工作空间），`close all`（关闭所有窗口），`clc`（清空命令窗口）

矩阵的乘法运算分为线性代数中的矩阵乘法‘\*’和同维度矩阵一一对应元素相乘的矩阵乘法‘.\*’

```
1 clear;close all;clc;
2 A=[1,2,3;
3     4,5,6]
4 B=[7,8,9;
5     10,11,12]
6 C=A*B
```

命令行窗口

A =

1	2	3
4	5	6

B =

7	8	9
10	11	12

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:

出错 ex1 (line 6)  
C=A\*B

错误原因 A 和 B 均为  $2 \times 3$  的矩阵，是无法相乘的

```
1 clear;close all;clc;
2 A=[1,2,3;
3     4,5,6]
4 B=[7,8,9;
5     10,11,12]
6 D=B'
7 C=A*D
```

命令行窗口

7	8	9
10	11	12

D =

7	10
8	11
9	12

C =

50	68
122	167

单引号‘表示将矩阵进行转置运算， $D=B'$ 就是定义矩阵 D 为矩阵 B 的转置，这样 A 是  $2 \times 3$  的矩阵，而 D 则是  $3 \times 2$  的矩阵，那么 A 和 D 就能相乘了。

如果 A 和 B 采用'.\*'，则两者是可以相乘的，其结果仍是  $2 \times 3$  的矩阵，A 和 B 对应位置的元素相乘。

```
1— clear;close all;clc;
2— A=[1,2,3;
3—     4,5,6]
4— B=[7,8,9;
5—     10,11,12]
6— C=A.*B
```

命令行窗口

A =

1	2	3
4	5	6

B =

7	8	9
10	11	12

C =

7	16	27
40	55	72

矩阵的乘方运算也分为'^'和'.^'，两者区别也是和乘法运算一样

```
1— clear;close all;clc;
2— A=[1,2,3;
3—     4,5,6;
4—     7,8,9]
5— B=A^3
6— C=A.^3
```

命令行窗口

A =

1	2	3
4	5	6
7	8	9

B =

468	576	684
1062	1305	1548
1656	2034	2412

C =

1	8	27
64	125	216
343	512	729

矩阵的逆可以用'^(-1)''（注意不是点乘方-1次）和函数 inv

```
1— clear;close all;clc;
2— A=[1,2,4;
3—     2,5,4;
4—     3,6,7]
5— B=A^(-1)
6— C=inv(A)
```

命令行窗口

A =

1	2	4
2	5	4
3	6	7

B =

-2.2000	-2.0000	2.4000
0.4000	1.0000	-0.8000
0.6000	-0.0000	-0.2000

C =

-2.2000	-2.0000	2.4000
0.4000	1.0000	-0.8000
0.6000	-0.0000	-0.2000

提取矩阵  $A$  中的元素。提取单个元素采用  $A(i,j)$  的形式，提取多个元素采用  $A(i1:di:i2,j1:dj:j2)$ ,  $i1$  表示提取元素的首行， $di$  表示提取元素的行间隔， $i2$  表示提取元素的末行， $j1$  表示提取元素的首列， $dj$  表示提取元素的列间隔， $j2$  表示提取元素的末列， $di$  和  $dj=1$  可省略。

如果  $A$  是一个  $5 \times 5$  的矩阵， $A(1:2:5,1:3)$  就是提取  $A(1,1), A(1,2), A(1,3); A(3,1), A(3,2), A(3,3); A(5,1), A(5,2), A(5,3)$  这些元素按原来的顺序排列成一个  $3 \times 3$  的矩阵

```
1 clear; close all; clc;
2 A=[1,2,3,4,5;2,3,4,5,6;3,4,5,6,7;4,5,6,7,8;5,6,7,8,9]
3 b1=A(2,3)%矩阵A第2行第3列元素
4 b2=A(1,1:5)%矩阵A第1行的元素
5 b3=A(1:5,2)%矩阵A第2列的元素
6 b4=A(1:2:5,1:3)%矩阵A第1,3,5行,第1,2,3列元素构成的矩阵
```

命令行窗口

A =

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

b1 =

4

b2 =

1	2	3	4	5
---	---	---	---	---

b3 =

2
3
4
5
6

b4 =

1	2	3
3	4	5
5	6	7

矩阵某些元素变化  $A(i1:di:i2,j1:dj:j2)=[\text{矩阵或向量}]$ ，其中要删除某行要写成  $A(:,2)=[]$ ；不能写成  $A(1:5,2)=[]$ ；单一一个 ':' 就表示一整行或一整列

```
1 clear; close all; clc;
2 A=[1,2,3,4,5;2,3,4,5,6;3,4,5,6,7;4,5,6,7,8;5,6,7,8,9]
3 A(2,3)=5%将矩阵A第2行第3列元素变为5
4 A(1,1:5)=[0,0,0,0,0]%矩阵A第1行的元素全部置0
5 A(:,2)=[]%删除矩阵A第2列的元素
```

命令行窗口

A =

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

A =

1	2	3	4	5
2	3	5	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

A =

0	0	0	0	0
2	3	5	5	6
3	4	5	6	7
4	5	6	7	8
5	6	7	8	9

A =

0	0	0	0
2	5	5	6
3	5	6	7
4	6	7	8
5	7	8	9

### 三. 一些常见的内置函数

`length` 是计算矩阵（向量）的维数中较大的那个数，比如 `A` 是  $2 \times 3$  的矩阵，`length(A)` 的结果就是 3，一般常用于计算向量的维数。

`size` 是计算矩阵的两个维数。

```
1 - clear;close all;clc;
2 - A=[1,2,3;2,3,4];
3 - b=length(A)
4 - c=size(A)
```

命令行窗口

A =

1	2	3
2	3	4

b =

3

c =

2	3
---	---

`max/min` 函数分别是计算矩阵（向量）最大值/最小值的函数，注意 `max(A)`，`min(A)` 是按照矩阵每一列来算最大、小值的，如果要算出整个矩阵的最大、小值，则需要写成 `max(max(A))`，`min(min(A))`

```
1 - clear;close all;clc;
2 - A=[1,5,3;2,3,4];
3 - b=max(A)
4 - c=max(max(A))
```

命令行窗口

A =

1	5	3
2	3	4

b =

2	5	4
---	---	---

c =

5

`ones/zeros/eye` 分别是生成全 1/全 0/单位阵，具体用法 `ones(m,n)`、`zeros(m,n)`、`eye(n)`

```
1 - clear;close all;clc;
2 - A=ones(2,3);
3 - B=zeros(3,2);
4 - C=eye(4)
```

命令行窗口

B =

0	0
0	0
0	0

C =

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



sum/diff 函数是对矩阵（向量）中的元素进行求和或差分，针对矩阵情况，sum(A)、diff(A)也是按列进行求和或差分，要对整个矩阵进行求和，需要写成sum(sum(A))

```
1 - clear;close all;clc;
2 - A=[1,2,3;4,5,6]
3 - s1=sum(A)
4 - d=diff(A)
5 - s2=sum(sum(A))
```

命令行窗口

A =

1	2	3
4	5	6

s1 =

5	7	9
---	---	---

d =

3	3	3
---	---	---

s2 =

21
----

find 函数是查找矩阵（向量）中元素的位置。find(A==2)，“==”在程序中表示相等的意思，“=”表示赋值，它是将矩阵按照行展开排序，比如下图矩阵 A 第 2 行第 3 列的那个 2 排序就是第 6

```
ex1.m x +
1 - clear;close all;clc;
2 - A=[1,2,3;4,5,2]
3 - m=find(A==2)
4
```

命令行窗口

A =

1	2	3
4	5	2

m =

3
6

fx >>

rand 函数是生成 0-1 上均匀分布的随机数（矩阵），rand(m,n)就是产生 m 行 n 列的随机矩阵

```
ex1.m x +
1 - %clear;close all;clc;
2 - rand(2,3)
3
```

命令行窗口

ans =

0.9572	0.8003	0.4218
0.4854	0.1419	0.9157

>> ex1

ans =

0.7922	0.6557	0.8491
0.9595	0.0357	0.9340

>> ex1

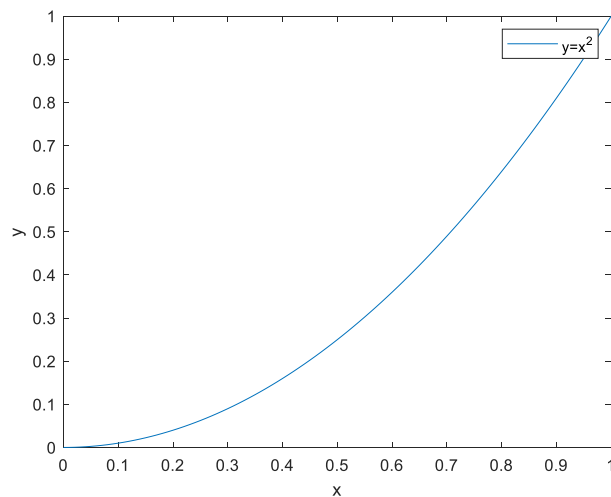
ans =

0.6787	0.7431	0.6555
0.7577	0.3922	0.1712

plot,xlabel,ylabel,legend 分别是画图，坐标轴和图例的函数

```
编辑器 - D:\Desktop\国赛\23数模国赛培训\第一节mat
ex1.m x +
1 — clear;close all;clc;
2 — x=0:0.01:1;%产生自变量
3 — y=x.^2;%计算函数值
4 — plot(x,y)%画图
5 — xlabel('x')
6 — ylabel('y')%x、y坐标
7 — legend('y=x^2')%图例

命令行窗口
```



plot 采用的是描点画图，x 是 0 到 1 上每间隔 0.01 取一个点

如果不懂函数的用法，可以在命令窗口输入 doc 函数名，如果不清楚函数的名称，可以百度描述一下函数的功能，比如百度 matlab 求行列式的函数，就能查到 det 是求行列式的函数，最后在命令窗口输入 doc det 就可以查看 det 函数的用法了





#### 四. 循环和条件结构

for 循环一般用于循环次数已知的情況下

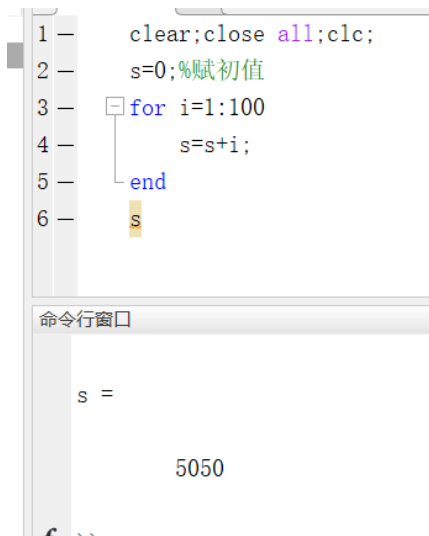
matlab 的 for 循环结构为

for i=a:d:b

end

其中 a 是循环起始值，d 是步长（也就是间隔），b 是循环终止值

例：计算  $S = 1 + 2 + \dots + 100$



当然这个例子比较简单，matlab 中 sum 函数也可以解决

while 循环是循环次数未知的情况下使用

matlab 的 while 循环结构

while（循环结束条件）

end

例：计算 n 的最大值，使得  $S_n < 10000$ ，其中  $S_n = \sum_{i=1}^n 2^i$

```

1 - clear;close all;clc;
2 - n=0;s=0;%赋初值
3 - while(s<10000)%当s<10000时,计算
4 -     n=n+1;
5 -     s=s+2^n;%累加
6 - end
7 - s=s-2^n
8 - n=n-1;%最后一次循环s已经大于10000,所以n需要减1
9 - %matlab内置函数验证一下
10 - x=1:n;y=2.^x;
11 - s1=sum(y)

```

命令行窗口

```

8190

n =

    12

s1 =

    8190

```

## 五. 函数模块的编写

在解决大型数学建模问题时,对于重复使用的部分代码或者特别冗长的代码选择性地写成函数模块,便于调用和简化主函数

函数文件的编写方式

function [输出 1,输出 2,...]=函数名(输入 1,输入 2)

代码

end

注意: 文件的名称需要与函数的名称一致

例: 给定  $N$ , 计算  $n$  的最大值, 使得  $S_n < N$ , 其中  $S_n = \sum_{i=1}^n 2^i$

sum2n.m

```

1 - function [s,n]=sum2n(N)
2 -     n=0;s=0;%赋初值
3 -     while(s<N)%当s<n时,计算
4 -         n=n+1;
5 -         s=s+2^n;%累加
6 -     end
7 -     s=s-2^n;
8 -     n=n-1;
9 - end

```

```

1 function [s,n]=s(N)
2     n=0;s=0;%赋初值
3     while(s<N)%当s<n时,计算
4         n=n+1;
5         s=s+2^n;%累加
6     end
7     s=s-2^n;
8     n=n-1;
9     end

```

命令窗口

```

>> [s,n]=s(1)
未定义函数或变量 's'。

```

如果函数名和文件名不一致，就无法调用  
 函数调用格式，在另一个 m 文件（两个文件需要在同一路径）如图调用，需要给定输入变量

```

1 clear;close all;clc;
2 N=10000;
3 [s,n]=sum2n(N)

```

命令窗口

```

s =
    8190

n =
    12

```

例：a 取不同值时，观察递推数列  $x_{n+1} = \frac{2}{x_n - 1}, x_1 = a$  的极限

如果递推数列极限存在，那么令  $\lim_{n \rightarrow \infty} x_n = X$ ，则  $X = \frac{2}{X-1}$ （单调有界准则证明略），则  $X = -1$  或  $2$

```

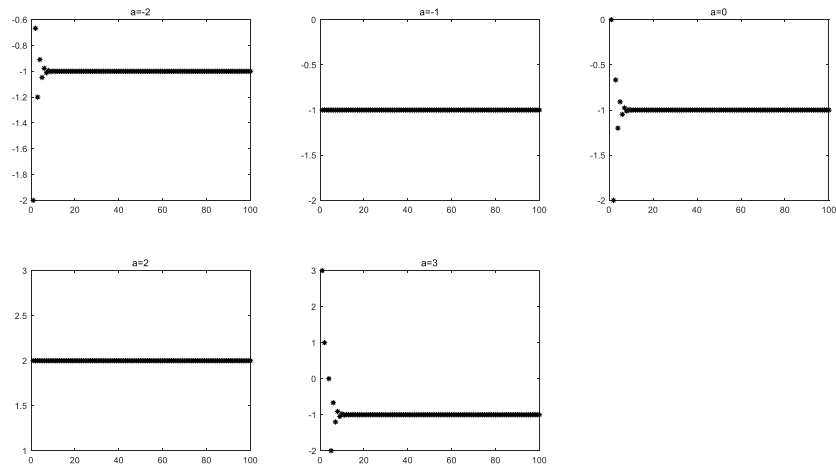
1 function x=xn(a,n)
2     x=zeros(n,1);%生产n行1列的0向量初始化
3     x(1)=a;%赋初值
4     for i=1:n-1
5         x(i+1)=2/(x(i)-1);
6     end%迭代计算递推数列
7     end

```

```

1  clear;close all;clc;
2  a=[-2,-1,0,2,3];
3  %计算出的极限是-1和2,所以分别在这两个值的两侧取值
4  n=100;%计算前100项
5  for i=1:length(a)
6      x=xn(a(i),n);
7      subplot(2,3,i)
8      plot(x,'k*')
9      title(['a=',num2str(a(i))])
10 end

```



例:编写  $n$  维矩阵  $A_n = \begin{pmatrix} 1 & 2 & 0 & \cdots & 0 \\ 1 & 2 & 3 & \cdots & 0 \\ 0 & 2 & \ddots & \ddots & \vdots \\ \vdots & 0 & \ddots & n-1 & n \\ 0 & 0 & \cdots & n-1 & n \end{pmatrix}$  (除了 3 条主对角线上有元素,

其余都是 0)

```

1  function A=An(n)
2  A=zeros(n);
3  for i=1:n
4      A(i,i)=i;%主对角线上元素
5  end
6  for i=2:n
7      A(i-1,i)=i;%上对角线
8      A(i,i-1)=i-1;%下对角线
9  end
10 end

```

命令窗口

```

>> A=An(5)

A =

     1     2     0     0     0
     1     2     3     0     0
     0     2     3     4     0
     0     0     3     4     5
     0     0     0     4     5

```