第3章 Matlab编程基础

3.1 Matlab的工作方式有二种

1. 交互式的指令操作方式。即用户在命令窗口中输入命令并按下回车键后，系统执行该指令并立即给出运算结果。

2. **m**文件的编程方式。**m**文件是由 **matlab**语句构成的文件,且文件名必须以**.m**为扩展名，如**ex.m**。用户可以用任何文件编辑器来对**M**文件进行编辑。

3.2 Matlab 程序设计

1、程序设计概述

- 基本概念

程序: 数据结构+算法
注解: 数据结构定义操作对象
算法定义求解过程

- 必要性

问题求解需要复杂算法
交互计算方式难以满足

2、 Matlab的基本程序设计原则

（1）设置完整的路径；

（2）参数值要集中放在程序的开始部分，便于维护；

（3）每行程序后输入分号，则执行程序行不会显示在屏幕上；如果不输入分号，则执行程序行会显示在屏幕上；

（4）符号“%”后的内容是注释行；

（5）如果语句在一行中放不下，则可以在行末键入三个点（...），指示下一行为续行；

（6）遇到不明白的命令，多使用在线帮助命令或系统演示示例；

（7）尽量使程序模块化，采用主程序调用子程序的方法，将所有子程序合并在一起来执行全部的操作。

3、MABLAB程序的基本组成部分

%说明部分
清除命令（可选）
定义变量（局部变量和全局变量）
按照顺序行执行的命令语句
控制语句开始
　　控制语句体
　　……
控制语句结束

其他命令（如绘图等）

3.3 Matlab 的文件

MATLAB语言编写的磁盘文件称为M文件，扩展名为m,格式为*.m，M文件可以互相调用，也可以调用它自己。

M文件的建立与打开

M文件是一个文本文件，它可以用任何编辑程序来建立和编辑，而一般常用且最为方便的是使用MATLAB提供的文本编辑器meditor。

1．建立新的M文件

为建立新的M文件，启动MATLAB文本编辑器有3种方法：

(1) 菜单操作。从MATLAB主窗口的File菜单中选择New菜单项，再选择M-file命令，屏幕上将出现MATLAB 文本编辑器窗口。

(2) 命令操作。在MATLAB命令窗口输入命令edit，启动MATLAB文本编辑器后，输入M文件的内容并存盘。

(3) 命令按钮操作。单击MATLAB主窗口工具栏上的New M-File命令按钮，启动MATLAB文本编辑器后，输入M文件的内容并存盘。

2．打开已有的M文件

打开已有的M文件，也有3种方法：

（1）菜单操作。从MATLAB主窗口的File菜单中选择Open命令，则屏幕出现Open对话框，在Open对话框中选中所需打开的M文件。在文档窗口可以对打开的M文件进行编辑修改，编辑完成后，将M文件存盘。

(2) 命令操作。在MATLAB命令窗口输入命令：edit 文件名，则打开指定的M文件。

(3) 命令按钮操作。单击MATLAB主窗口工具栏上的Open File命令按钮，再从弹出的对话框中选择所需打开的M文件。

M文件可以根据调用方式的不同分为两类：命令文件(Script File)和函数文件(Function File)。它们的扩展名均为m，主要区别在于：

（1）命令文件没有输入参数，也不返回输出参数，而函数文件可以带输入参数，也可以返回输出参数。

（2）命令文件对Matlab工作空间中的变量进行操作文件中所有命令的执行结果也完全返回工作空间中，而函数文件中定义的变量为局部变量，当函数文件执行完毕时，这些变量被清除。

（3）命令文件可以直接运行，在Matlab命令窗口输入命令文件的名字，就会顺序执行命令文件中的命令，而函数文件不能直接运行，要以函数调用的方式来调用。

函数文件

a. 格式：　**function [f1,f2,…]=fun(x,y,z,…)**

其中**x,y,z,…**是形式输入参数，**f1,f2,…**为返回的形式输出参数值，**fun**为形式函数名，函数名一般就是这个函数文件的文件名。

b.文件前面几行由**%**开始的语句构成了**M**文件的帮助信息，当键入：

help 文件名

可得到文件的说明信息。（**help**为在线帮助）

程序输入函数input

```
A=input('the number is');
B=input('Her name is','s');
Example
```

例子：

```
Year=input('请输入年份：');
N=weekday([int2str(Year),'-4-30']);
Day=1+14-N;
fprintf('%d年的母亲节日期：%d-%d-%d\n',Year,Year,5,Day)
```

例 分别建立命令文件和函数文件，将华氏温度f转换为摄氏温度c。

程序1：
首先建立命令文件并以文件名f2c.m存盘。

```
clear;          %清除工作空间中的变量
f=input('Input Fahrenheit temperature：');
c=5*(f-32)/9
```

然后在MATLAB的命令窗口中输入f2c，将会执行该命令文件，
执行情况为：

```
Input Fahrenheit temperature：73
c =
   22.7778
```

程序2：

首先建立函数文件f2c.m。

```
function c=f2c(f)
c=5*(f-32)/9
```

然后在MATLAB的命令窗口调用该函数文件。

```
clear;
y=input('Input Fahrenheit temperature：');
x=f2c(y)
```

输出情况为：

```
Input Fahrenheit temperature：70
c =
   21.1111
x =
   21.1111
```

例 建立一个命令文件将变量**a,b**的值互换，然后运行该命令文件。

首先建立命令文件并以文件名**exch.m**存盘：

```
clear;
a=1:10; b=[11,12,13,14;15,16,17,18];
c=a;a=b;b=c;
a
b
```

然后在**MATLAB**的命令窗口中输入**exch**，将会执行该命令文件。

程序2：

建立一个函数文件将变量a,b的值互换，然后在命令窗口调用该函数文件。

首先建立函数文件**fexch.m**：

```
function [a,b]=exch(a,b)
c=a;a=b;b=c;
```

然后在**MATLAB**的命令窗口调用该函数文件：

```
clear;
x=1:10; y=[11,12,13,14;15,16,17,18];
[x,y]=exch(x,y)
```

- 创建一个命令（底稿）文件

```
%plot a circle
t=0:0.01:10;
a=1;
x=sin(t)+a;
y=cos(t);
plot(x,y)
axis equal
```

- 函数文件实例

```
function [mean,stdev] = stat(x)
% STAT Interesting statistics.
n = length(x);
mean = avg(x,n);
stdev = sqrt(sum((x-avg(x,n)).^2)/n);
%----------------------
function mean = avg(x,n)
%AVG subfunction
mean = sum(x)/n;
```

建议：**matlab**中一个函数保存为一个文件，**matlab**系统是根据文件名查找函数的

函数文件和命令文件的区别

A、形式上，函数文件第一行必须包含关键字**function**，且有函数名和输入输出形式参数，而底稿文件没有；

B、函数文件可以传递参数，底稿文件不具备参数传递功能；

C、函数文件中定义及使用的变量都是局部变量，只在本函数内有效，底稿文件中定义及使用的变量都是全局变量，在退出文件后仍有效。

程序控制结构

Matlab语言的程序结构与其它高级语言是一致的，分为顺序结构，选择结构，循环结构。

顺序结构

1．数据的输入

从键盘输入数据，则可以使用**input**函数来进行，该函数的调用格式为：

A=input(提示信息，选项**);**

其中提示信息为一个字符串，用于提示用户输入什么样的数据。

如果在**input**函数调用时采用‘**s'**选项，则允许用户输入一个字符串。例如，想输入一个人的姓名，可采用命令：

```
xm=input('What''s your name?','s');
```

2．数据的输出

MATLAB提供的命令窗口输出函数主要有**disp**函数、**fprintf**函数。 **disp**调用格式为

disp(输出项**)**

其中输出项既可以为字符串，也可以为矩阵。

例 求一元二次方程**ax2 +bx+c=0**的根。

程序如下：

```
a=input('a=?');
b=input('b=?');
c=input('c=?');
d=b*b-4*a*c;
x=[(-b+sqrt(d))/(2*a),(-b-sqrt(d))/(2*a)];
disp(['x1=',num2str(x(1)),',x2=',num2str(x(2))]);
```

- **fprintf**函数最常见的使用方式用以下例子说明.

若键入命令

fprintf（‘圆周率pi=%10.9f’，pi）

则会按浮点型输出含**9**位小数，**1**位整数的圆周率近似值，其输出结果为

圆周率**pi=3.141592654**

若键入命令

n=23;

fprintf(‘n=%d’,n)

则会按整型数输出**n**值，其输出结果为

n=23

选择结构

1．if语句

在MATLAB中，if语句有3种格式。

(1) 单分支if语句：

```
if 条件
   语句组
end
```

当条件成立时，则执行语句组，执行完之后继续执行if语句的后继语句，若条件不成立，则直接执行if语句的后继语句。

(2) 双分支**if**语句：

if 条件

语句组**1**

else

语句组**2**

end

当条件成立时，执行语句组**1**，否则执行语句组**2**，语句组**1**或语句组**2**执行后，再执行**if**语句的后继语句。

例 计算分段函数的值。

程序如下：

```
x=input('请输入x的值:');
if x<=0
  y= (x+sqrt(pi))/exp(2);
else
  y=log(x+sqrt(1+x*x))/2;
end
y
```

练习：计算以下分段函数的值

$$y = \begin{cases} \dfrac{\sin x}{x} & x \neq 0 \\ 1 & x = 0 \end{cases}$$

(3) 多分支if语句：

```
if  条件1
    语句组1
elseif  条件2
    语句组2
   ……
elseif  条件m
     语句组m
else
     语句组n
end
```

语句用于实现多分支选择结构。

例 输入一个字符，若为大写字母，则输出其对应的小写字母；若为小写字母，则输出其对应的大写字母；若为数字字符则输出其对应的数值，若为其他字符则原样输出。

```
c=input('请输入一个字符','s');
if c>='A' & c<='Z'
   disp(char(abs(c)+abs('a')-abs('A')));
elseif c>='a'& c<='z'
    disp(char(abs(c)- abs('a')+abs('A')));
elseif c>='0'& c<='9'
    disp(abs(c)-abs('0'));
else
    disp(c);
end
```

switch语句

switch语句根据表达式的取值不同，分别执行不同的语句，其语句格式为：

```
switch  expr                （计算的表达式）
  case  表达式1             （表达式i可以是标量值或单元数组，把expr
  case  表达式2                  和单元数组中的所有元素比较）
    语句组2
    ……
  case  表达式m
    语句组m
  otherwise
    语句组n
end
```

当**expr**的值等于表达式**1**的值时，执行语句组**1**，当表达式的值等于表达式**2**的值时，执行语句组**2**，**…**，当表达式的值等于表达式**m**的值时，执行语句组**m**，当表达式的值不等于**case**所列的表达式的值时，执行语句组**n**。当任意一个分支的语句执行完后，直接执行**switch**语句的下一句。

例 某商场对顾客所购买的商品实行打折销售，标准如下(商品价格用price来表示)：

price<200　　没有折扣

200≤price<500　　3%折扣

500≤price<1000　　5%折扣

1000≤price<2500　　8%折扣

2500≤price<5000　　10%折扣

5000≤price　　14%折扣

输入所售商品的价格，求其实际销售价格。

程序如下：

```
price=input('请输入商品价格');
switch fix(price/100)
  case {0,1}            %价格小于200
    rate=0;
  case {2,3,4}          %价格大于等于200但小于500
    rate=3/100;
  case num2cell(5:9)      %价格大于等于500但小于1000
    rate=5/100;
  case num2cell(10:24)    %价格大于等于1000但小于2500
    rate=8/100;
  case num2cell(25:49)    %价格大于等于2500但小于5000
    rate=10/100;
  otherwise             %价格大于等于5000
    rate=14/100;
end
price=price*(1-rate)            %输出商品实际销售价格
```

已知学生的名字和百分制分数，采用“满分”，“优秀”“良好”“及格”“不及格”显示。

```
a=cell(1,10);b=cell(1,10);c=cell(1,10);d=cell(1,10);
for k=1:10
   a(k)={89+k};   b(k)={79+k};
   c(k)={69+k};   d(k)={59+k};
end
A=cell(3,5)
A(1,:)={'zhang','wang','li','zhao','qian'};
A(2,:)={70,74,89,67,92};
for k=1:5
   switch A{2,k}
      case 100
         r='满分'
      case a
         r='优秀'
      case b
         r='良好'
      case c
         r='及格'
      otherwise
         r='不及格'
   end
   A(3,k)={r}
end
A
```

try语句

语句格式为：

try

　语句组1

catch

　语句组2

end

try语句先试探性执行语句组1，如果语句组1在执行过程中出现错误，则将错误信息赋给保留的**lasterr**变量，并转去执行语句组2。

例　矩阵乘法运算要求两矩阵的维数相容，否则会出错。
先求两矩阵的乘积，若出错，则自动转去求两矩阵的点乘。
程序如下：

```
A=[1,2,3;4,5,6]; B=[7,8,9;10,11,12];
try
  C=A*B;
catch
  C=A.*B;
end
C
lasterr              %显示出错原因
```

循环结构

循环是指按照给定的条件，重复执行指定的语句，这是一种十分重要的程序结构。

1. for-end 循环

循环次数事先确定

格式为：

```
for i=n:s:m    (初值:步长:终值)
  语句体
end
```

s 为步长，可以为正数，负数或小数。

注：变量x通常称为循环变量，但也可以是一个数组。

- 如果循环变量是数组，则按数组中的每一列执行一次。
- 在每一次循环中，x被指定为数组的下一列，即在第n次循环中，x=array(:, n)。

```
x=[0 2 3;4 7 9]
for a=x              //把矩阵x的每一列元素依次赋给变量a
b=a+4
end
```

运行结果：

```
x =
    0    2    3
    4    7    9
b =
    4
    8
b =
    6
   11
b =
    7
   13
```

与For循环相关的规定：

A、For循环内不能对循环变量重新赋值。

B、For循环可以嵌套。

C、为了节省时间，可用下列2种方法

a、当有一个等效的数组方法来解给定的问题时，应避免用For循环。

例如，» n=1:10;

» x=sin(n*pi/10)

b、在For循环(While循环)被执行之前，应预先分配数组。例如，在For循环内每执行一次命令，变量x的大小增加1。迫使MATLAB每通过一次循环要花费时间对x分配更多的内存。为了消去这个步骤，For循环的例子应重写为

```
x=zeros(1,10);    %  preallocated memory for x
for n=1:10
  x(n)=sin(n*pi/10);
end
```

要想在任何时候终止循环（**for**或**while**循环），可利用**break**语句。

```
x=[0 2 3;4 7 9]
for a=x
  b=a+4
  break
  %return
end
```

运行结果

```
x =
     0     2     3
     4     7     9
b =
     4
     8
```

```
x=[0 2 3;4 7 9]
while x
  b=x+4
 end
```

例 一个三位整数各位数字的立方和等于该数本身则称该数为水仙花数。输出全部水仙花数。

程序如下：

```
for m=100:999
  m1=fix(m/100);          %求m的百位数字
  m2=rem(fix(m/10),10);   %求m的十位数字
  m3=rem(m,10);           %求m的个位数字
  if m==m1*m1*m1+m2*m2*m2+m3*m3*m3
    disp(m)
  end
end
```

例 已知 $y=\sum_{i=1}^{n}\frac{1}{2i-1}$ ，当**n=100**时，求**y**的值。

程序如下：

```
y=0;
n=100;
for i=1:n
  y=y+1/(2*i-1);
end
y
```

在实际MATLAB编程中，采用循环语句会降低其执行速度，所以前面的程序通常由下面的程序来代替：

```
n=100;
i=1:2:2*n-1;
y=sum(1./i);
y
```

例 写出下列程序的执行结果。

```
s=0;
a=[12,13,14;15,16,17;18,19,20;21,22,23];
for k=a
   s=s+k;
end
disp(s');
```

- 练习：编程输入范德蒙型的矩阵

$$A=\begin{bmatrix}1 & -1 & 1 & -1 & 1\\ 1 & 0 & 0 & 0 & 0\\ 1 & 1 & 1 & 1 & 1\\ 1 & 2 & 4 & 8 & 16\\ 1 & 3 & 9 & 27 & 81\end{bmatrix}$$

```
for i=1:5
    for j=1:5
        a(i,j)=(i-2)^(j-1)
    end
end
```

2. while-end 循环

用于循环次数不能事先确定的，

格式为

```
while 表达式
  语句体
end
```

只有表达式为真，就执行语句体，表达式为假，终止该循环。

注：表达式可以是一个矩阵，且矩阵中的所有元素都为非0时，才执行循环体中的内容。如果表达式为一空矩阵，则循环体中的内容永远不会被执行。

```
num=1;a=0;
while num<=100
   a=a+num;
   num=num+1;
end
a
```

例 从键盘输入若干个数，当输入0时结束输入，求这些数的平均值和它们之和。

程序如下：

```
sum=0;
cnt=0;
val=input('Enter a number (end in 0):');
while (val~=0)
    sum=sum+val;
    cnt=cnt+1;
    val=input('Enter a number (end in 0):');
end
if (cnt > 0)
   sum
   mean=sum/cnt
end
```

3．break语句、continue语句和return语句

与循环结构相关的语句还有**break**语句和**continue**语句。它们一般与**if**语句配合使用。

break语句用于终止循环的执行。当在循环体内执行到该语句时，程序将跳出循环，继续执行循环语句的下一语句。

continue语句控制跳过循环体中的某些语句。当在循环体内执行到该语句时，程序将跳过循环体中所有剩下的语句，继续下一次循环。

return语句是终止程序的运行。

程序暂停语句

暂停程序的执行可以使用pause函数，其调用格式为：

pause（n） （n为延迟秒数）

如果省略延迟时间，直接使用pause，则将暂停程序，直到用户按任一键后程序继续执行。

若要强行中止程序的运行可使用Ctrl+C命令。

例 求[100，200]之间第一个能被21整除的整数。

程序如下：

```
for n=100:200
if rem(n,21)~=0
    continue
end
break
end
n
```

例：**从1到n的任何一个自然数, 只要对n反复进行下列两种运算:**

1）如果n是偶数, 就除以2 ;

2）如果n是奇数, 就乘以3加1，

最后的结果总是1。

这个问题大约是在二十世纪五十年代被提出来的。在西方它常被称为西拉古斯Syracuse)猜想，因为据说这个问题首先是在美国的西拉古斯大学被研究的；而在东方，这个问题由将它带到日本的日本数学家角谷静夫的名字命名，被称作角谷猜想。

角谷静夫曾用计算机验算到7×1011，并未出现反例。1992年李文斯(G.T.Leavens)和孚门南(M.Vermeulen)也以计算机对小于5.6×1013的正整数进行验证，也未发现反例。

```
n=input('请输入一个大于1的正整数 n=');
if n<=0
    disp('输入的数为负数或零，程序中断')
    break
end
while n>1
   if  rem(n,2)==0
          n=n/2
   else
          n=n*3+1
   end
end
```

循环的嵌套

如果一个循环结构的循环体又包括一个循环结构，就称为循环的嵌套，或称为多重循环结构。

例 若一个数等于它的各个真因子之和，则称该数为完数，如6=1+2+3，所以6是完数。求[1,500]之间的全部完数。

程序如下：

```
for m=1:500
s=0;
for k=1:m/2
if rem(m,k)==0
s=s+k;
end
end
if m==s
   disp(m);
end
end
```

作曲线y=x(1-x)在0到1区间上的转动切线，从几何上说明水平切线的存在性。

```
axis([0,1,0,1])
hold on
x=0:0.005:1;
 y=x.*(1-x);
 plot(x,y,'r')
 x0=0:0.05:1;
 y0=x0.*(1-x0);
 n=length(x0);
 ybar=1-2*x0;
for i=1:n
for x1=0:0.01:1
y1=y0(i)+ybar(i)*(x1-x0(i));
plot(x1,y1,'k')
end
pause(0.1)
end
plot([0,1],[1/4,1/4],'k')
xlabel('x轴')
ylabel('y轴')
title('水平切线的存在性演示')
text(0.4,0.2,'y=x(1-x)')
hold off
```

程序举例

猜数游戏。首先由计算机产生[1,100]之间的随机整数，
然后由用户猜测所产生的随机数。根据用户猜测的情况给出不同提示，如猜测的数大于产生的数，则显示“您的数字较大!”，小于则显示“您的数字较小!'”，等于则显示“YOU WIN! ”，同时退出游戏。用户最多可以猜6次。

```
y=round(10+89*rand());
for k=1:6;
x=input(['第',num2str(k),'次输入一个两位数(输完请按回车):']);
if(x<y)
    '您的数字较小!'
else if(x==y)
     msgbox('YOU WIN!');
     return;
    else  '您的数字较大!'
    end
end
 if(k==6)
     msgbox('YOU LOSE!  GAME OVER!')
 end
end
```

2、用筛选法求某自然数范围内的全部素数。

素数是大于**1**，且除了**1**和它本身以外，不能被其他任何整数所整除的整数。用筛选法求素数的基本思想是：要找出**2~m**之间的全部素数，首先在**2~m**中划去**2**的倍数**(**不包括**2)**，然后划去**3**的倍数**(**不包括**3)**，由于**4**已被划去，再找**5**的倍数**(**不包括**5)**，**…**，直到再划去不超过的数的倍数，剩下的数都是素数。

```
m=input('m=');
    p=2:m;
    for i=2:sqrt(m)
       n=find(rem(p,i)==0&p~=i);
       p(n)=[];
    end
    p
```

例、猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个，第二天又将剩下的桃子吃了一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到到第十天早上想再吃，只剩一个桃子。求第一天共摘多少桃子？

```
number=1;
x(10)=number;
fprintf('计算结果如下：\n');
for i=9:-1:1
    x(i)=(x(i+1)+1)*2;
    fprintf('    第%d天有%d个桃子\n',i,x(i));
end
```

例、统计一个字符串里有几个字母？

程序如下：

```
function k=f(s)
[m,n]=size(s); x=isletter(s);
if x(1)+x(n)==2 %若该字符串首尾均是字母
   k=1;
elseif x(1)+x(n)==1  %若该字符串首或尾有一个且仅有一个是字母
   k=0;
elseif x(1)+x(n)==0  %若该字符串首尾均不是字母（比如标点符号或空格）
   k=-1;
End
k=0
for i=1:n
   if x(i)==1
      k=k+1;
end
end
```

函数变量及其变量作用域

在**MATLAB**语言中，变量可以分为输入变量、输出变量和函数内使用的变量。

输入变量相当于函数的入口数据，也是一个函数操作的主要对象，从某种意义上说，函数的功能在于对输入变量进行一定的操作从而实现一定的功能。函数的输入变量为局部变量，函数对输入变量的一切操作和修改如果不依靠输出变量的话，将不会影响工作区间中该变量的值。

5.3.1 变量的输入和输出规则

MATLAB可以有任意数量的输入和输出变量。这些参数的特性和规则如下：

- 函数式**M**文件可以没有输入和输出变量。
- 函数可以用比**M**文件中的函数定义行所规定的输入输出变量更少的变量进行调用。但是不能用比规定的输入输出变量更多的变量进行调用。
- 在一次调用中所用到的输入和输出变量的个数可以通过分别调用函数**nargin**和**nargout**来确定。因为**nargin**和**nargout**是函数而不是变量，所以用户不能用诸如**nargin=nargin+pi**之类的语句对它们进行重新赋值。
- 当一个函数被调用时，输入变量并没有被复制到函数的工作区间中，但是它们的值在这个函数是可读的。应当注意的是，如果输入变量的任何值被改变了，这个输入变量组就被复制到了函数的工作区。

- 如果一个函数声明了一个或者多个输出变量，但是用户在使用的时候又不想要输出参数，则只要不把输出变量赋值给任何变量就可以了；或者在函数结束之前用函数**clear**删除这些变量。

- 函数可以通过在函数声明中将**varargin**作为最后的输入参数，接受可变的任意个数的输入参数。**Varargin**是一个预先定义的单元数组，这个单元数组的第**i**个单元就是**varargin**程序的位置算起的第**i**个变量。

- 通过函数声明行中将**varargout**作为最后的输出变量，函数可以接受任意个数的变量形式的输出参数。**Varargout**也是一个预先定义的单元数组，这个单元数组的第**i**个单元就是从**varargout**的出现位置算起的第**i**个变量。

- 函数**nargchk**和**nargoutch**分别提供了对有效地输入和输出变量个数的简单错误校验，因为如果函数调用的输入或者输出变量的个数多于函数定义中出现的个数，函数都自动地返回一个错误，因此虽然这些函数的作用有限，但是在一个函数定义声明了任意数目的输入变量和输出变量的时候却是非常有用的。

例、**nargin**函数的初级使用方法。

解：本程序实现如下功能，当调用过程时小于或等于一个变量时，系统提示错误的输入，当有两个变量时，程序将两个数相加，当有**3**个变量时，将前两个数相加并减去第**3**个。程序如下：

```
function d=nargintest(a,b,c)
if nargin<=1
error('Not enough input arguments')
elseif nargin==2
d=a+b;
elseif nargin==3
d=a+b+c
end
```

此外，使用**nargin**可以查找函数输入变量的个数，例如，想要查找上述**nargintest**函数的参数个数，可以使用命令 **nargin('nargintest')**

5.3.2 局部变量

局部变量是在函数内部使用的变量，其影响范围只能在本函数内，每个函数在运行时，都占有独立的函数工作空间，此工作空间和**MATLAB**的工作空间时相互独立的，局部变量仅存在于函数的工作空间内。当函数执行完毕之后，该变量即自行消失。

5.3.3 全局变量

在**Matlab**中，函数内部定义的变量都是局部变量，它们不被加载到工作区间中。有时，用户需要使用全局变量，这时要使用**global**函数来进行定义，而且在任何使用该全局变量的函数中都应加以定义，即使是在命令窗口也不例外。

例 全局变量应用示例。

先建立函数文件**wadd.m**，该函数将输入的参数加权相加。

```
function f=wadd(x,y)
global ALPHA BETA
f=ALPHA*x+BETA*y;
```

在命令窗口中输入：

```
global ALPHA BETA
ALPHA=1;
BETA=2;
s=wadd(1,2)
```

5.3.4 永久变量

除了通过全局变量共享数据外，函数式M文件还可以通过声明一个变量persistent来对函数中重复使用和递归调用的变量的访问进行限制，使用格式形如persistent (X Y Z)。

永久变量与全局变量类似，但是它的范围被限制在声明这些变量的函数内部，不允许在其他的函数中对它们进行改变。只要M文件还在MATLAB 7的内存中，永久变量就存在。