第5章

MATLAB程序设计及应用

在MATLAB中,用户可以使用两种方法执行运算:一种是交互式的命令执行方式;另一种是M文件的程序执行方式。命令执行方式是在命令窗口中输入命令,MATLAB逐句解释执行。这种方式虽然简单、直观,但速度较慢,不保留执行过程。程序执行方式是将有关命令编成程序存储在一个文件中(称为M文件),当运行该程序后,MATLAB就会自动依次执行该文件中的命令,直至全部命令执行完毕。以后需要这些命令时,只要再次运行该程序。程序执行方式为实际应用中的重要执行方式。

MATLAB的程序设计既有传统高级语言的特征,又有自己独特的优点。在MATLAB程序设计时,充分利用MATLAB数据结构的特点,可以使程序结构简单,编程效率高。本章介绍有关MATLAB程序控制结构以及程序设计的基本方法。

本章要点:

(1) M文件。

(2) 程序控制结构。

(3) 函数文件。

(4) 程序调试。

(5) 程序设计应用。

学习目标:

(1) 理解MATLAB的编程流程。

(2) 掌握MATLAB的3种程序控制结构。

(3) 掌握MATLAB函数文件的编写方法。

(4) 了解MATLAB的变量类型。

(5) 了解MATLAB中的错误处理与程序调试方法。

(6) 掌握MATLAB的程序设计应用。

5.1 M文件

5.1.1 M文件的分类

用MATLAB语言编写的程序,称为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)。命令文件也称为脚本文件,通常用

于执行一系列简单的 MATLAB 命令，运行时只需输入文件名字，MATLAB 就会自动按顺序执行文件中的命令。函数文件和命令文件不同，它可以接收参数，也可以返回参数，在一般情况下，用户不能靠单独输入其文件名来运行函数文件，而必须由其他语句来调用，MATLAB 的大多数应用程序都以函数文件的形式给出。它们的扩展名均为.m，主要区别如下。

(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到工作空间中，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，这些变量被消除。

(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，而要以函数调用的方式来调用它。

【例 5-1】 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。

程序 1：

首先建立命令文件并以文件名 li3.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 li3.m，将会执行该命令文件，输出为：

```
li3
a =
    11    12    13    14
    15    16    17    18
b =
     1     2     3     4     5     6     7     8     9    10
```

调用该命令文件时，不用输入参数，也没有输出参数，文件自身建立需要的变量。当文件执行完毕后，可以用命令 whos 查看工作空间中的变量。这时会发现 a、b、c 仍然保留在工作空间中。

程序 2：

首先建立函数文件 fli3.m。

```
function [a,b] = fli3(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] = fli3(x,y)
```

输出结果为：

```
x =
    11    12    13    14
    15    16    17    18
y =
     1     2     3     4     5     6     7     8     9    10
```

调用该函数文件时，既有输入参数，又有输出参数。当函数调用完毕后，可以用命令whos查看工作空间中的变量。这时会发现函数参数a、b、c未被保留在工作空间中，而x、y保留在工作空间中。

5.1.2 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。

1. 建立新的M文件

为建立新的M文件，应先启动MATLAB文本编辑器，具体有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。

注意：M文件存放的位置一般是MATLAB默认的工作目录work，当然也可以是其他目录。如果是其他目录，则应该将该命令设定为当前命令或将其加到搜索路径中。

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需要打开的M文件。

5.2 程序控制结构

MATLAB程序结构一般可分为顺序结构、选择结构、循环结构3种。任何复杂的程序都可以由这3种结构构成。

5.2.1 顺序结构

顺序结构是指按照程序中语句的排列顺序依次执行，直到程序的最后一个语句。这是最简单的一种程序结构。一般涉及数据的输入、数据的计算或处理、数据的输出等内容。

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。例如，从键盘输入 A 矩阵，可以采用下面的命令来完成。

```
A = input('输入 A 矩阵')
```

执行该语句，屏幕显示提示信息：

```
输入 A 矩阵
```

等待用户从键盘按 MATLAB 规定的格式输入 A 矩阵的值。

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，其调用格式为：

```
disp(输出项)
```

其中，输出项既可以为字符串，也可以为矩阵。

注意：和前面介绍的矩阵显示方式不同，用 disp 函数显示矩阵时将不显示矩阵的名字，而且其输出格式更紧凑，且不留任何没有意义的空行。

【例 5-2】 求一元二次方程 $ax^2+bx+c=0$ 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))]);
```

输出结果为：

```
a = ?5
b = ?37
```

```
c = ?49
x1 = -1.7277,x2 = -5.6723
```

3. 程序的暂停

在运行时,为了查看程序中间结果或者查看输出的图形,有时需要暂停程序的执行。这时可以使用 pause 函数,其调用格式为:

```
pause(延迟秒数)
```

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

若要强行中止程序的运行可使用快捷键 Ctrl+C。

5.2.2 选择结构

选择结构是根据给定的条件成立与否,分别指向不同的语句。MATLAB 用于实现选择结构的语句有 if 语句、switch 语句和 try 语句。

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 5-3】 计算分段函数。

$$y=\begin{cases}\cos(x+1)+\sqrt{x^2+1}, & x=10\\ x\sqrt{x+\sqrt{x}}, & x\neq 10\end{cases}$$

程序如下:

```
x = input('请输入 x 的值:');
if x == 10
```

```
  y = cos(x + 1) + sqrt(x * x + 1);
else
  y = x * sqrt(x + sqrt(x));
end
y
```

也可以用单分支 if 语句来实现：

```
x = input('请输入 x 的值:');
y = cos(x + 1) + sqrt(x * x + 1);
if x~ = 10
   y = x * sqrt(x + sqrt(x));
end
y
```

或用以下程序：

```
x = input('请输入 x 的值:');
if x == 10
   y = cos(x + 1) + sqrt(x * x + 1);
end
if x~ = 10
   y = x * sqrt(x + sqrt(x));
end
y
```

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
```

注意：

（1）每条 if 语句都尾随有一条 end 语句。

（2）如果都要使用 else 和 elseif 语句，则必须将 else 语句放在 elseif 语句的后面，其用于处理未加说明的所有条件。

（3）elseif 语句并不需要单独的 end 语句。

（4）if 语句可以相互嵌套，可根据实际需要将各个 if 语句进行嵌套来解决比较复杂的实际问题。

【例 5-4】 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


微视频 5-1

输出。

程序如下：

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

输出结果为：

```
请输入一个字符 G
g
```

2. 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  表达式
   case  表达式 1
      语句组 1
   case  表达式 2
      语句组 2
      …
   case  表达式 m
      语句组 m
   otherwise
      语句组 n
end
```

当表达式的值等于表达式 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 语句的下一条语句。


微视频 5-2

【例 5-5】 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}                        % 价格大于或等于 200 但小于 500
      rate = 3/100;
   case num2cell(5:9)                  % 价格大于或等于 500 但小于 1000
      rate = 5/100;
   case num2cell(10:24)                % 价格大于或等于 1000 但小于 2500
      rate = 8/100;
   case num2cell(25:49)                % 价格大于或等于 2500 但小于 5000
      rate = 10/100;
   otherwise                           % 价格大于或等于 5000
      rate = 14/100;
end
price = price * (1 - rate)             % 输出商品实际销售价格
```

输出结果为：

```
请输入商品价格 2576
price =
  2.3184e + 003
```

num2cell 函数是将数值矩阵转换为单元矩阵，num2cell(5:9)等价于{5,6,7,8,9}。

3. 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。

在 MATLAB 中，常出现的一些警告信息，如表 5-1 所示。

表 5-1 警告信息的命令

命 令	说 明
error('message')	显示出错信息 message，终止程序
errordlg('errorstring','dlgname')	显示出错信息的对话框，对话框的标题为 dlgname
warning('message')	显示警告信息 message，程序继续进行

【例 5-6】 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
```

输出结果为：

```
C =
     7    16    27
    40    55    72
ans =
Error using ==> mtimes
Inner matrix dimensions must agree.
```

5.2.3 循环结构

1. for 语句

for 语句的格式为：

```
for 循环变量 = 表达式 1:表达式 2:表达式 3
      循环体语句
end
```

其中表达式 1 的值为循环变量的初值，表达式 2 的值为步长，表达式 3 的值为循环变量的终值。步长为 1 时，表达式 2 可以省略。

执行过程为：首先计算 3 个表达式的值，再将表达式 1 的值赋给循环变量，如果此时循环变量的值介于表达式 1 和表达式 3 的值之间，则执行循环体语句，否则结束循环的执行。执行完一次循环之后，循环变量自增一个表达式 2 的值，然后再判断循环变量的值是否介于表达式 1 和表达式 3 结果的值之间，如果是，仍然执行循环体，直至条件不满足。这时将结束 for 语句的执行，而继续执行 for 语句后面的语句。

注意：for 语句需要伴随一个 end 语句。end 语句标志着所要执行语句的结束。


微视频 5-3

【例 5-7】 若一个 3 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
```

输出结果为：

```
153
370
371
407
```

【例 5-8】 已知 $y=\frac{1}{1^2}+\frac{1}{2^2}+\frac{1}{3^2}+\cdots+\frac{1}{n^2}$，当 $n=100$ 时，求 y 的值。

程序如下：

```
y = 0;n = 100;
for i = 1:n
y = y + 1/i/i;
end
y
```

输出结果为：

```
y =
    1.6350
```

在实际 MATLAB 编程中，为提高程序的执行速度，常用向量运算来代替循环操作，所以上述程序通常用下面的程序来代替。

```
n = 100;
i = 1:n;
f = 1./i.^2;
y = sum(f)
```

【例 5-9】 设 $f(x)=e^{-0.5x}\sin\left(x+\frac{\pi}{6}\right)$，求 $\int_0^{3\pi} f(x)\mathrm{d}x$。

以梯形法为例，程序如下：

```
a = 0;b = 3 * pi;
n = 1000; h = (b - a)/n;
x = a; s = 0;
f0 = exp( - 0.5 * x) * sin(x + pi/6);
for i = 1:n
    x = x + h;
    f1 = exp( - 0.5 * x) * sin(x + pi/6);
    s = s + (f0 + f1) * h/2;
    f0 = f1;
end
s
```

输出结果为：

```
s =
    0.9008
```

上述程序来源于传统的编程思想。也可以利用向量运算，从而使得程序更加简洁，更有MATLAB的特点。程序如下：

```
a = 0;b = 3 * pi;
n = 1000; h = (b - a)/n;
x = a:h:b;
f = exp( - 0.5 * x). * sin(x + pi/6);
for i = 1:n
    s(i) = (f(i) + f(i + 1)) * h/2;
end
s = sum(s)
```

程序中x、f、s均为向量，f的元素为各个x点的函数值，s的元素分别为n个梯形的面积，s各元素之和即定积分近似值。

事实上，MATLAB提供了有关数值积分的标准函数，实际应用中直接调用这些函数求数值积分，这些内容已在第4章中做了介绍。

按照MATLAB的定义，for语句的循环变量可以是一个列向量。for语句更一般的格式为：

```
for 循环变量 = 矩阵表达式
   循环体语句
end
```

执行过程是依次将矩阵的各列元素赋给循环变量，然后执行循环体语句，直至各列元素处理完毕。

2. while 语句

while语句的一般格式为：

```
while (条件)
    循环体语句
end
```

其执行过程为：若条件成立，则执行循环体语句，执行后再判断条件是否成立，若不成立则跳出循环。


微视频 5-4

【例 5-10】 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的和。

程序如下：

```
sum = 0;
n = 0;
val = input('Enter a number (end in 0):');
while (val~ = 0)
```

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

输出结果为：

```
Enter a number (end in 0):63
Enter a number (end in 0):78
Enter a number (end in 0):15
Enter a number (end in 0):24
Enter a number (end in 0):0
sum =
   180
mean =
    45
```

3. break 语句和 continue 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。

【例 5-11】 求[100,200]区间第一个能被 21 整除的整数。

程序如下：

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

输出结果为：

```
n =
   105
```

4. 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，则称之为循环的嵌套，或多重循环结

构。实现多重循环结构仍用前面介绍的循环语句。多重循环的嵌套层数可以是任意的。在设计多重循环时,要注意内、外循环之间的关系,以及各语句放置的位置。

【例 5-12】 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]区间的全部完数。

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
```

输出结果为:

```
  6
 28
496
```

5.3 函数文件

函数文件是另一种形式的 M 文件,每一个函数文件都定义一个函数。事实上,MATLAB 提供的标准函数大部分都是函数文件定义的。

5.3.1 函数文件的基本结构

函数文件由 function 语句引导,其基本结构为:

```
function 输出形参表 = 函数名(输入形参表)
  注释说明部分
  函数体语句
```

其中,以 function 开头的一行为引导行,表示该 M 文件是一个函数文件。函数名的命名规则与变量名相同。输入形参为函数的输入参数,输出形参为函数的输出参数。当输出形参多于一个时,则应该用方括号括起来。

说明:

(1) 关于函数文件名。

函数文件名通常由函数名再加上扩展名.m 组成,不过函数文件名与函数名也可以不相同。当两者不同时,MATLAB 将忽略函数名而确认函数文件名,因此调用时使用函数文件名。不过最好把文件名和函数名统一,以免出错。

(2) 关于注释说明。

注释说明包括 3 部分内容：

① 紧随函数文件引导行之后以%开头的第一注释行。这一行一般包括大写的函数文件名和函数功能简要描述，供 lookfor 关键词查询和 help 在线帮助时使用。

② 第一注释行及之后连续的注释行。通常包括函数输入/输出参数的含义及调用格式说明等信息，构成全部在线帮助文本。

③ 与在线帮助文本相隔一空行的注释行。包括函数文件编写和修改的信息等。

(3) 关于 return 语句。

如果在函数文件中插入了 return 语句，则执行到该语句就结束函数的执行，程序流程转至调用该函数的位置。通常，在函数文件中也可不使用 return 语句，这时在被调用函数执行完成后自动返回。

【例 5-13】 编写函数文件求半径为 r 的圆的面积和周长。


微视频 5-5

函数文件如下：

```
function [s,p] = fcircle(r)
% CIRCLE  calculate the area and perimeter of a circle of radii r
% r          圆半径
% s          圆面积
% p          圆周长

% 2020 年 2 月 30 日编
s = pi * r * r;
p = 2 * pi * r;
```

将以上函数文件以文件名 fcircle.m 存盘，然后在 MATLAB 命令窗口调用该函数：

```
[s,p] = fcircle(10)
```

输出结果为：

```
s =
  314.1593
p =
   62.8319
```

采用 lookfor 命令和 help 命令可以显示出注释说明部分的内容，其功能和一般 MATLAB 函数的帮助信息是一致的。

5.3.2 函数调用

函数文件编制好以后就可以调用了。函数调用的一般格式是：

```
[输出实参表] = 函数名(输入实参表)
```

注意：函数调用时各实参出现的顺序、个数，应与函数定义时形参的顺序、个数一致，否则会出错。函数调用时，先将实参传递给相应的形参，从而实现参数传递，然后再执行函数

的功能。


微视频 5-6

【例 5-14】 利用函数文件,实现直角坐标(x,y)与极坐标(ρ,θ)之间的转换。

已知转换公式为:

极坐标的极径:

$$\rho=\sqrt{x^2+y^2}$$

极坐标的极角:

$$\theta=\arctan\left(\frac{y}{x}\right)$$

函数文件 tran. m:

```
function [rho,theta] = tran(x,y)
rho = sqrt(x * x + y * y);
theta = atan(y/x);
```

调用 tran. m 的命令为:

```
x = input('Please input x = :');
y = input('Please input y = :');
[rho,the] = tran(x,y);
rho
the
```

输出结果为:

```
Please input x = :5
Please input y = :7
rho =
    8.6023
the =
0.9505
```

在 MATLAB 中,函数可以嵌套调用,即一个函数可以调用别的函数,甚至调用它自身。一个函数调用它自身称为函数的递归调用。

【例 5-15】 利用函数的递归调用,求 $n!$。

$n!$ 本身就是以递归的形式定义的:

$$n!=\begin{cases}1, & n\leqslant 1\\ n(n-1)!, & n>1\end{cases}$$

显然,求 $n!$ 需要求$(n-1)!$,这时可采用递归调用。递归调用函数文件 factor. m 如下:

```
function f = factor(n)
if n < = 1
   f = 1;
else
   f = factor(n - 1) * n;          % 递归调用求(n - 1)!
end
```

在命令窗口中调用函数文件 factor.m 求 $s=1!+2!+3!+4!+5!$。

```
s = 0;
for i = 1:5
   s = s + factor(i);
end
s
```

输出结果为：

```
s =
   153
```

5.3.3 函数参数

MATLAB 的函数参数包括函数的输入参数和输出参数。函数提供输入参数接收数据，经过函数执行后由输出参数给出结果，因此，MATLAB 的函数调用就是输入参数和输出参数传递的过程。

1. 参数的传递

函数的参数传递是将主函数中的变量值传送给被调函数的输入参数，被调函数执行后，将结果通过被调函数的输出参数传送给主函数的变量。被调函数的输入参数和输出参数都存放在函数的工作空间中，与 MATLAB 的工作空间是独立的，当调用结束后，函数的工作空间数据被清除，被调函数的输入参数和输出参数也被清除。

2. 参数的个数

MATLAB 函数的输入参数和输出参数在使用时，不用事先声明和定义，参数的个数可以改变。在调用函数时，MATLAB 用两个永久变量 nargin 和 nargout 分别记录调用该函数时的输入实参和输出实参的个数。只要在函数文件中包含这两个变量，就可以准确地知道该函数文件被调用时的输入输出参数个数，从而决定函数如何进行处理。

5.3.4 函数变量

MATLAB 的函数变量根据作用范围，可以分为局部变量和全局变量。

1. 局部变量

局部变量的作用范围是函数的内部。如果没有特别声明，函数内部的变量都是局部变量，都有自己的函数工作空间，与 MATLAB 的工作空间是独立的。局部变量仅在函数内部执行时存在，函数执行完毕，变量就消失了。

2. 全局变量

全局变量的作用范围是全局数的，可以在不同的函数和 MATLAB 工作空间中共享。使用全局变量可以减少参数的传递，有效地提高程序的执行效率。

全局变量在使用前必须用 global 命令定义，格式为：

```
global 变量名
```

要清除全局变量可以用 clear 命令,格式为:

```
clear global 变量名                    % 清除某个全局变量
clear global                           % 清除所有的全局变量
```

在函数文件里,全局变量的定义语句应放在变量使用之前,一般都放在文件的前面,用大写字符命名,以防重复定义。

5.4 程序调试

程序调试是程序设计的重要环节,也是程序设计人员必须掌握的重要技能。MATLAB 提供了相应的程序调试功能,既可以提供文本编辑器对程序进行调试,又可以在命令窗口结合具体的命令进行。

5.4.1 程序调试概述

程序调试的目的是检查程序是否正确,即程序能否顺利运行并得到预期结果。在运行程序之前,应先设想到程序运行的各种情况,测试在各种情况下程序是否能正常运行。

一般来说,应用程序的错误有两类:一类是语法错误,另一类是运行时的错误。语法错误包括词法或文法的错误,例如,函数名的拼写错误、表达式书写错误等。MATLAB 能够检查出大部分的语法错误,给出相应的错误信息,并标出错误在程序中的行号。

程序运行时的错误是指程序的运行结果有错误,这类错误也称为程序逻辑错误。在程序设计中逻辑错误是较为常见的一类错误,这类错误往往隐蔽性较强、不易查找。产生逻辑错误的原因通常是算法设计有误,这时需要对算法进行修改。程序的运行错误通常包括不能正常运行和运行结果不正确,出错的原因一般有:

(1) 数据不对,即输入的数据不符合算法要求。

(2) 输入的矩阵大小不对,尤其是当输入的矩阵为一维数组时,应注意行向量与列向量在使用上的区别。

(3) 程序不完善,只能对某些数据运行正确,而对另一些数据则运行错误,或是根本无法正常运行,这有可能是算法考虑不周所致。

5.4.2 MATLAB 调试菜单

MATLAB 的 M 文件编辑器除了能编辑修改文件外,还能对程序进行调试。通过调试菜单可以查看和修改函数工作空间中的变量,从而准确地找到运行错误。通过调试菜单设置断点可以使程序运行到某一行暂停运行,可以查看和修改工作空间中的变量值,来判断断点之前的语句逻辑是否正确。还可以通过调试菜单逐行运行程序,逐行检查和判断程序是否正确。

MATLAB 调试菜单界面如图 5-1 所示。

(1) 编辑打开:用于调试打开的 M 文件。

(2) 单步调试:用于单步调试程序。

(3) 调试进入:用于单步调试进入子函数。

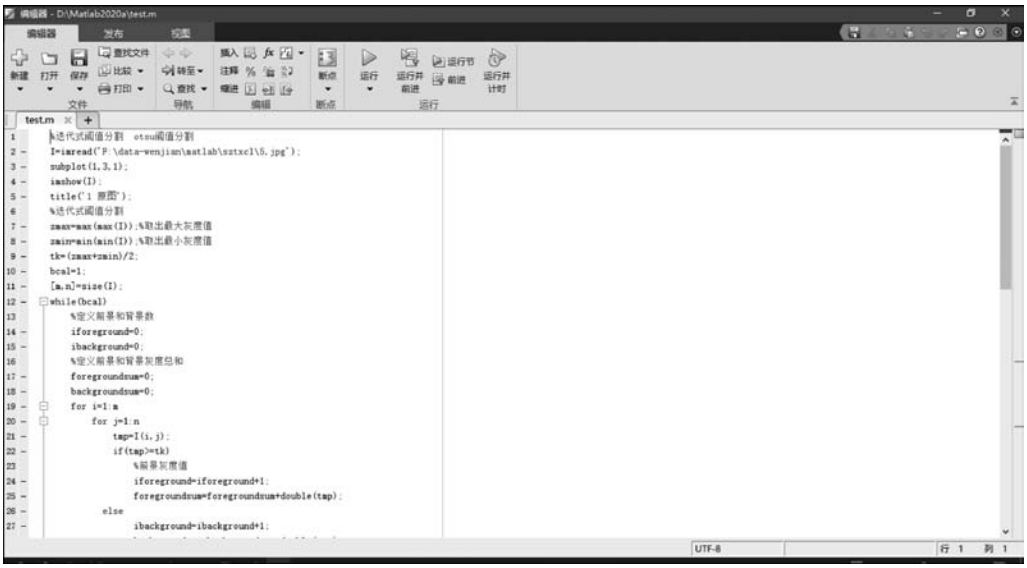

图 5-1　调试菜单界面

(4) 调试退出：用于单步调试从子函数跳出。

(5) 继续：程序执行到下一断点。

(6) 清除断点：清除所有打开文件中的断点。

(7) 错误停止：在程序出错或报警处停止往下执行。

(8) 退出：退出调试模式。

对于 MATLAB 调试，MATLAB 提供了 F9 键进行选中的程序运行，也可以按 Ctrl＋Enter 键运行程序。当然，对于工具调试也有一些快捷键，具体为：

(1) 快捷键 F10——实现单步调试。

(2) 快捷键 F11——用于单步调试进入子函数。

(3) 快捷键 Shift＋F11——用于单步调试从子函数跳出。

(4) 快捷键 F5——实现程序执行到下一断点。

5.4.3　调试命令

除了采用调试菜单调试程序外，MATLAB 还提供了一些命令(如表 5-2 所示)用于程序调试。命令的功能和调试菜单命令类似，具体使用方法请查询 MATLAB 帮助文档。

表 5-2　MATLAB 常用调试命令及功能

命　令	功　能	命　令	功　能
dbstop	用于在 M 文件中设置断点	dbup	修改当前工作空间的上、下文关系
dbstatus	显示断点信息	dbdown	
dbtype	显示 M 文件文本(包括行号)	dbclear	清除已设置的断点
dbstep	从断点处执行 M 文件	dbcont	继续执行
dbstack	显示 M 文件执行时调用的堆栈等	dbquit	退出调试状态

5.5 程序设计应用

5.5.1 等级与闰年

【例 5-16】 编写一个程序,输入一个数值分数,输出等级分数。要求:grade>95 为 A;95≥grade>85 为 B;85≥grade>75 为 C;75≥grade≥60 为 D;60>grade≥0 为 F。

方法 1:程序如下:

```
grade = input('请输入分数');
if grade > 95.0
  disp('The grade is A.');
elseif grade > 85.0
  disp('The grade is B.');
elseif grade > 75.0
  disp('The grade is C.');
elseif grade > = 60.0
  disp('The grade is D.');
else
  disp('The grade is F.');
end
```

输出结果为:

```
请输入分数 82
The grade is C.
请输入分数 60
The grade is D.
请输入分数 54
The grade is F.
```

方法 2:程序如下:

```
grade = input('请输入分数');
if grade > 95.0
  disp('The grade is A.');
else
    if grade > 85.0
  disp('The grade is B.');
    else
        if grade > 75.0
  disp('The grade is C.');
        else
            if grade > = 60.0
  disp('The grade is D.');
else
disp('The grade is F.');
end
end
```

```
end
end
```

输出结果为：

```
请输入分数 82
The grade is C.
请输入分数 60
The grade is D.
请输入分数 50
The grade is F.
```

说明：对于有许多选项的选择结构来说，最好在一个 if 结构中使用多个 elseif 语句，尽量不用 if 的嵌套结构。

【例 5-17】 编写一个程序，计算闰年。在公历中，闰年是这样规定的：

（1）能被 400 整除的年为闰年；

（2）能被 100 整除但不能被 400 整除的年不为闰年；

（3）能被 4 整除但不能被 100 整除的年为闰年；

（4）其余的年份不为闰年。

程序如下：

```
disp('This program calculates the day of year given the ');
disp('current date.');
month = input('Enter current month (1 - 12):');
day = input('Enter current day(1 - 31):');
year = input('Enter current year(yyyy): ');
% Check for leap year, and add extra day if necessary
if mod(year,400) == 0
leap_day = 1; % Years divisible by 400 are leap years
elseif mod(year,100) == 0
leap_day = 0; % Other centuries are not leap years
elseif mod(year,4) == 0
leap_day = 1; % Otherwise every 4th year is a leap year
else
leap_day = 0; % Other years are not leap years
end
% Calculate day of year by adding current day to the
% days in previous months.
day_of_year = day;
for ii = 1:month - 1
% Add days in months from January to last month
switch(ii)
case{1,3,5,7,8,10,12},
day_of_year = day_of_year + 31;
case{4,6,9,11},
day_of_year = day_of_year + 30;
case 2,
day_of_year = day_of_year + 28 + leap_day;
```

```
end
end
 % Tell user
fprintf('The date  % 2d/ % 2d/ % 4d is day of year  % d.\n', ...
month, day, year, day_of_year);
```

输出结果为：

```
This program calculates the day of year given the current date.
Enter current month (1 - 12):1
Enter current day(1 - 31):24
Enter current year(yyyy): 1999
The date  1/24/1999 is day of year 24.
This program calculates the day of year given the current date.
Enter current month (1 - 12):1
Enter current day(1 - 31):1
Enter current year(yyyy): 2000
The date  1/ 1/2000 is day of year 1.
This program calculates the day of year given the current date.
Enter current month (1 - 12):12
Enter current day(1 - 31):31
Enter current year(yyyy): 2000
The date 12/31/2000 is day of year 366.
This program calculates the day of year given the current date.
Enter current month (1 - 12):5
Enter current day(1 - 31):16
Enter current year(yyyy): 2001
The date  5/16/2001 is day of year 136.
```

说明：用到 mod(求余)函数来确定一个数是否能被另一个数整除。如果函数的返回值为 0，则说一个数能被另一个数整除；否则不然。

5.5.2 程序运行时间测量

在 MATLAB 中测试时间要用到函数 tic 和 toc。tic 函数复位内建计时器，toc 函数从最后一次调用 tic 开始以秒计时。因为在许多计算机中，时间钟是相当粗略的，所以有必要多运行几次以获得相应的平均数。

【例 5-18】 比较循环和向量算法执行所用的时间，要求：

(1) 用 for 循环计算 1～10 000 的每一个整数的平方，事先不初始化平方数组。

(2) 用 for 循环计算 1～10 000 的每一个整数的平方，事先初始化平方数组。

(3) 用向量算法计算 1～10 000 的每一个整数的平方。

程序如下：

```
% 1. Using a for loop with an uninitialized output array.
% 2. Using a for loop with an preallocated output array.
% 3. Using vectors.
% "square". This calculation is done only once
% because it is so slow.
```

```
maxcount = 1;                          % One repetition
tic;                                   % Start timer
for jj = 1:maxcount
clear square                           % Clear output array
for ii = 1:10000
square(ii) = ii^2;                     % Calculate square
end
end
average1 = (toc)/maxcount;             % Calculate average time
 % Perform calculation with a preallocated array
 % "square". This calculation is averaged over 10
 % loops.
maxcount = 10;                         % One repetition
tic;                                   % Start timer
for jj = 1:maxcount
clear square                           % Clear output array
square = zeros(1,10000);               % Preinitialize array
for ii = 1:10000
square(ii) = ii^2;                     % Calculate square
end
end
average2 = (toc)/maxcount;             % Calculate average time
 % Perform calculation with vectors. This calculation
 % averaged over 100 executions.
maxcount = 100;                        % One repetition
tic;                                   % Start timer
for jj = 1:maxcount
clear square                           % Clear output array
ii = 1:10000;                          % Set up vector
square = ii.^2;                        % Calculate square
end
average3 = (toc)/maxcount;             % Calculate average time
 % Display results
fprintf('Loop / uninitialized array =  %8.4f\n', average1);
fprintf('Loop / initialized array =  %8.4f\n', average2);
fprintf('Vectorized =  %8.4f\n', average3);
```

输出结果为：

```
Loop / uninitialized array = 0.0942
Loop / initialized array = 0.0005
Vectorized = 0.0001
Loop / uninitialized array = 0.0742
Loop / initialized array = 0.0006
Vectorized = 0.0001
```

【例 5-19】 比较循环结构、选择结构与应用逻辑数组运算的快慢，要求：

(1) 创建一个含 10 000 个元素的数组，其值依次为 1～10 000 的整数。用 for 循环和 if

结构计算大于 5000 的元素的平方根。

(2) 创建一个含 10 000 个元素的数组,其值依次为 1～10 000 的整数。用逻辑数组计算大于 5000 的元素的平方根。

程序如下:

```
% 1.Using a for loop and if construct.
% 2.Using a logical array.
% Perform calculation using loops and branches
maxcount = 1;                  % One repetition
tic;                           % Start timer
for jj = 1:maxcount
a = 1:10000;                   % Declare array a
for ii = 1:10000
if a(ii)> 5000
a(ii) = sqrt(a(ii));
end
end
end
average1 = (toc)/maxcount;     % Calculate average time
% Perform calculation using logical arrays.
maxcount = 10;                 % One repetition
tic;                           % Start timer
for jj = 1:maxcount
a = 1:10000;                   % Declare array a
b = a > 5000;                  % Create mask
a(b) = sqrt(a(b));             % Take square root
end
average2 = (toc)/maxcount;     % Calculate average time
% Display result
fprintf('Loop /if approach =  %8.4f\n',average1);
fprintf('Logical array approach =  %8.4f\n',average2);
```

输出结果为:

```
Loop /if approach =  0.0293
Logical array approach = 0.0006
Loop /if approach = 0.0022
Logical array approach = 0.0008
```

由此例可以看到,用逻辑数组方法速度是另一种方法的 20 倍以上。

编程总结:在有选择结构和循环结构的编程中,要遵循以下编程指导思想。如果长期坚持这些原则,那么代码将会有很少的错误,有了错误也易于修改,而且在以后修改程序时,也使别人易于理解。

(1) 对于 for 循环体总是要缩进两个或更多空格,以增强程序的可读性。

(2) 在循环体中绝不修改循环变量的值。

(3) 在循环执行开始之前,总是要预先分配一个数组,这样能大大增加循环运行的

速度。

(4) 如果有可能,可用逻辑函数选择数组中的元素。用逻辑数组进行运算比循环快得多。

5.5.3 M文件与函数的区别

【例 5-20】 经典的鸡兔同笼问题:鸡兔同笼,头36,脚100。求鸡兔各几只。

程序如下:

```
i = 1;
while i > 0
    if rem(100 - i * 2,4) == 0&(i + (100 - i * 2)/4) == 36
        break;
    end
    i = i + 1;
    n1 = i;
    n2 = (100 - 2 * i)/4;
    break
end
    fprintf('The number of chicken is %d.\n',n1);
    fprintf('The number of rabbit is %d.\n',n2);
```

将该M文件以chicken为文件名保存,并在命令窗口中输入chicken,运行该程序得到如下结果:

```
chicken
The number of chicken is 2.
The number of rabbit is 24.
```

【例 5-21】 编制一个函数,要求任意输入两个数值后,用两个子函数分别求出它们的和与它们的绝对值的和,再将这两个和相乘。

编写如下函数:

```
function ch = zihanshu(x,y)                          % 主函数
ch = zihanshu1(x,y) * zihanshu2(x,y);
function ch = zihanshu1(x,y)                         % 子函数 1
ch = abs(x) + abs(y);
function ch = zihanshu2(x,y)                         % 子函数 2
ch = x + y;
```

调用结果为:

```
x = 1;y = -9;
zihanshu(x,y)
ans =
    -80
```

实训项目五

本实训项目的目的如下：

- 掌握建立和执行 M 文件的方法。
- 掌握利用 if 语句实现选择结构的方法。
- 掌握利用 switch 语句实现多分支选择结构的方法。
- 熟悉 try 语句的使用。
- 掌握利用 for 语句实现循环结构的方法。
- 掌握利用 while 语句实现循环结构的方法。
- 熟悉利用向量运算来代替循环操作的方法。
- 掌握定义和调用 MATLAB 函数的方法。

5-1　计算以下分段函数值。

$$\begin{cases} x, & x<0 \\ 2x^2+1, & 0\leqslant x<1 \\ 3x^3+2x^2+1, & x\geqslant 1 \end{cases}$$

5-2　编写一个程序，求下列函数的值。

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

5-3　我国新税法规定：个税按 5000 元/月的起征标准算，全月应纳税税率如表 5-3 所示。

表 5-3　个税税率

全月收入中应缴纳所得税部分	税率/%
不超过 3000 元的	3
3000～12 000 元的部分	10
12 000～25 000 元的部分	20
25 000～35 000 元的部分	25
35 000～55 000 元的部分	30
55 000～80 000 元的部分	35
超过 80 000 元的部分	45

试编程加以计算。

5-4　计算下列各式的值。

(1) 1+2+3+…+99+100

(2) 1!+2!+3!+…+99!+100!

要求：

① 分别用 for 循环、while 循环和向量显示进行；

②比较三者程序运行的时间。

5-5　根据 $y=1+\frac{1}{3}+\frac{1}{5}+\cdots+\frac{1}{2n-1}$，求 $y<3$ 时的最大 n 值和 y 值。

5-6　计算 $\arcsin x$。

$$\arcsin x \approx x+\frac{x^2}{4\times 3}+\frac{1\times 3\times x^5}{16\times 4\times 5}+\cdots+\frac{(2n)!}{2^{2n}(n!)^2}\frac{x^{2n+1}}{(2n+1)},\quad \text{其中}\ |x|<1。$$

x 为输入参数，当 x 不满足条件时就不计算，并显示提示；当 x^{2n+1} 前的系数小于 0.000 01 时，则循环结束(其中求 $n!$ 可以使用函数 factorial，如果不使用该函数，如何实现该程序)。

5-7　编写 M 函数文件，通过流程控制语句，建立如下的矩阵。

$$\boldsymbol{y}=\begin{bmatrix} 0 & 1 & 2 & 3 & \cdots & n \\ 0 & 0 & 1 & 2 & \cdots & n-1 \\ 0 & 0 & 0 & 1 & \cdots & n-2 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

5-8　已知

$$\begin{cases} f_1=1, & n=1 \\ f_2=0, & n=2 \\ f_3=1, & n=3 \\ f_n=f_{n-1}-2f_{n-2}+f_{n-3}, & n>3 \end{cases}$$

求 $f_1\sim f_{100}$ 中：

(1) 最大值、最小值、各数之和。

(2) 正数、零、负数的个数。

5-9　已知

$$y=\frac{f(40)}{f(30)+f(20)}$$

(1) 当 $f(n)=n+10\ln(n^2+5)$ 时，求 y 的值。

(2) 当 $f(n)=1\times 2+2\times 3+3\times 4+\cdots+n\times(n+1)$ 时，求 y 的值。