

第五章

程式控制流程

謝明興 撰

| | | |
|-----|--------------------|----|
| 5-1 | 迴圈for loops | 2 |
| 5-2 | 迴圈while Loops..... | 6 |
| 5-3 | if條件指令..... | 7 |
| 5-4 | switch多重條件指令 | 9 |
| 5-5 | 條件敘述與遞迴函式..... | 11 |
| 習題 | | 12 |

<http://www.tau.edu.tw/shiehms/matlab.htm>

5-2 MATLAB 程式設計基礎

<http://wwwt.au.edu.tw/shiehms/m5.zip>

5-1 迴圈 for loops

MATLAB 提供的 for loops 迴圈指令使用語法如下：

```
for 變數 = 表示式  
    程式敘述;  
end
```

其中表示式如帶有：運算子的向量，如

```
for R = 1:N
```

```
for S = 1.0: -0.1: 0.0 遞增值為-0.1
```

```
for S = 矩陣
```

break; 指令可以跳脫 **for** 迴圈，如在 **for** 多重迴圈中，則跳出 **break** 敘述所在的 **for** 迴圈

continue; 指令可以跳到 **for** 迴圈主體內的下一個指令繼續執行

for 迴圈會降低 MATLAB 的執行效率，應盡量使用向量化運算



for01.m

```
% for01.m  
clear all  
n=5  
for r = 1:n  
    for c = 1:n  
        A(r,c) = 1/(r+c-1);  
    end  
end  
A
```

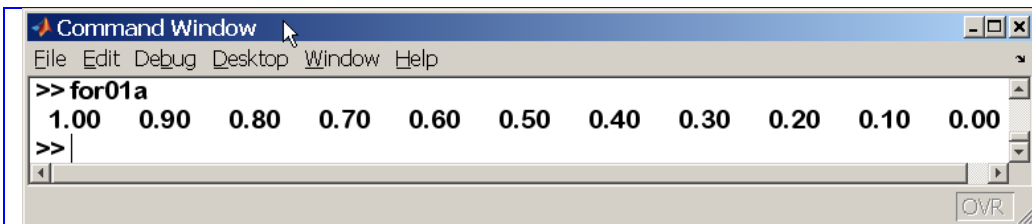
for01.m 執行結果

```
n =
    5
A =
    1.0000    0.5000    0.3333    0.2500    0.2000
    0.5000    0.3333    0.2500    0.2000    0.1667
    0.3333    0.2500    0.2000    0.1667    0.1429
    0.2500    0.2000    0.1667    0.1429    0.1250
    0.2000    0.1667    0.1429    0.1250    0.1111
```

for01a.m

```
% for01a.m
clear all
for r = 1.0:-0.1:0.0
    fprintf('%6.2f\t', r)
end
fprintf('\n');
```

for01a執行結果



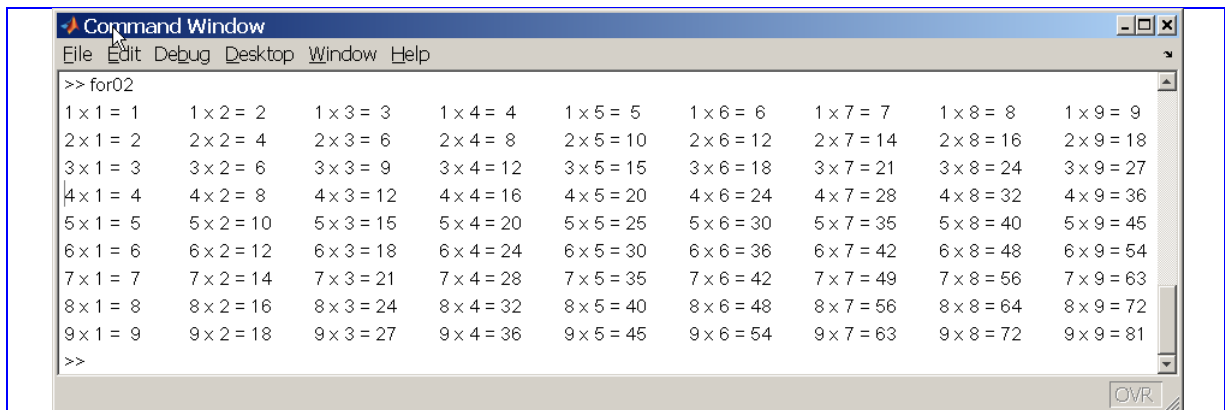
```
Command Window
File Edit Debug Desktop Window Help
>> for01a
 1.00  0.90  0.80  0.70  0.60  0.50  0.40  0.30  0.20  0.10  0.00
>>
```

5-4 MATLAB 程式設計基礎

for02.m

```
% for02.m
clear all
n=9;
for r = 1:n
    for c = 1:n
        fprintf('%d x %d = %2d \t', r, c, r*c)
    end
    fprintf('\n');
end
```

for02執行結果



The screenshot shows the MATLAB Command Window with the following output:

```
>> for02
1 x 1 = 1    1 x 2 = 2    1 x 3 = 3    1 x 4 = 4    1 x 5 = 5    1 x 6 = 6    1 x 7 = 7    1 x 8 = 8    1 x 9 = 9
2 x 1 = 2    2 x 2 = 4    2 x 3 = 6    2 x 4 = 8    2 x 5 = 10   2 x 6 = 12   2 x 7 = 14   2 x 8 = 16   2 x 9 = 18
3 x 1 = 3    3 x 2 = 6    3 x 3 = 9    3 x 4 = 12   3 x 5 = 15   3 x 6 = 18   3 x 7 = 21   3 x 8 = 24   3 x 9 = 27
4 x 1 = 4    4 x 2 = 8    4 x 3 = 12   4 x 4 = 16   4 x 5 = 20   4 x 6 = 24   4 x 7 = 28   4 x 8 = 32   4 x 9 = 36
5 x 1 = 5    5 x 2 = 10   5 x 3 = 15   5 x 4 = 20   5 x 5 = 25   5 x 6 = 30   5 x 7 = 35   5 x 8 = 40   5 x 9 = 45
6 x 1 = 6    6 x 2 = 12   6 x 3 = 18   6 x 4 = 24   6 x 5 = 30   6 x 6 = 36   6 x 7 = 42   6 x 8 = 48   6 x 9 = 54
7 x 1 = 7    7 x 2 = 14   7 x 3 = 21   7 x 4 = 28   7 x 5 = 35   7 x 6 = 42   7 x 7 = 49   7 x 8 = 56   7 x 9 = 63
8 x 1 = 8    8 x 2 = 16   8 x 3 = 24   8 x 4 = 32   8 x 5 = 40   8 x 6 = 48   8 x 7 = 56   8 x 8 = 64   8 x 9 = 72
9 x 1 = 9    9 x 2 = 18   9 x 3 = 27   9 x 4 = 36   9 x 5 = 45   9 x 6 = 54   9 x 7 = 63   9 x 8 = 72   9 x 9 = 81
>>
```

for03.m 向量

```
% for03.m
clear all
S=[1 3 5 7 9];
for r = S
    fprintf(' %3d \t', r*r)
end
fprintf('\n');
```



for03執行結果

```
>> for03
    1      9     25     49     81
```



for04.m 矩陣

```
% for04.m
clear all
S=[1 2; 11 12; 21 22];
for r = S
    fprintf(' %3d \t', r)
end
fprintf('\n');
```



for04執行結果

```
>> for04
    1     11     21     2     12     22
```



for04a.m矩陣

```
% for04a.m
clear all
S=[1 2; 11 12; 21 22];
for r = S
    fprintf(' %3d \t', r.*r)
end
fprintf('\n');
```



for04a執行結果

```
>> for04a
```

5-6 MATLAB 程式設計基礎

5-2 迴圈 while Loops

MATLAB 提供的 **while** 迴圈指令使用語法如下：

while 條件式

 程式敘述;

end

break; 指令可以跳脫 **for** 迴圈，如在 **for** 多重迴圈中，則跳出 **break** 敘述所在的 **for** 迴圈

while 迴圈會降低 **MATLAB** 的執行效率，應盡量使用向量化運算



while01prod.m顯示**k!**最小於且最接近**10000**的**k**值

```
% while01prod.m
clear all
k=1;
res=10000;
while prod(1:k) < res
    k=k+1;
end
k=k-1;
fprintf('The nearest of %d of prod(%d) is %d! = %4g \t', res, k, k,
prod(1:k))
fprintf('\n');
```



while01prod執行結果

```
>> while01prod
The nearest of 10000 of prod(7) is 7! = 5040
```

5-3 if 條件指令

MATLAB 提供的 **if** 條件指令使用語法如下：

```
if 條件式  
    程式敘述;
```

```
else  
    程式敘述;
```

```
end
```

或

```
if 條件式 1  
    程式敘述;
```

```
elseif 條件式 2  
    程式敘述;
```

```
elseif 條件式 3  
    程式敘述;
```

```
elseif  
    程式敘述;
```

```
end
```

break; 指令可以跳脫 **for** 迴圈，如在 **for** 多重迴圈中，則跳出 **break** 敘述所在的 **for** 迴圈

while 迴圈會降低 **MATLAB** 的執行效率，應盡量使用向量化運算

5-8 MATLAB 程式設計基礎



if01prod.m顯示k!最小於且最接近10000的值

```
% if01prod.m
clear all
k=1:100;
for r = k
    if prod(1:r)>10000
        break;
    end
end
r=r-1;
fprintf('The nearest of 10000: %d! = %d \t', r, prod(1:r))
fprintf('\n');
```



if01prod執行結果

```
>> if01prod
The nearest of 10000: 7! = 5040
```


5-4 switch 多重條件指令

MATLAB 提供的 **switch** 條件指令使用語法如下：

```
switch 條件式
    case {條件值 1a, ...}
        程式敘述;
    case {條件值 2a, ...}
        程式敘述;
    case {條件值 3a, ...}
        程式敘述;
    otherwise
        程式敘述;
end
```

MATLAB 的 **switch** 條件指令不需像 C 語言需使用 **break;** 指令跳脫 **switch** 條件敘述

5-10 MATLAB 程式設計基礎

-  **switch01.m** 數入月份，輸入 2 ~ 4 顯示 **spring**, 5~7 顯示 **summer**, 輸入 8 ~ 10 顯示 **autumn**, 輸入 11, 12, 1 顯示 **winter**,

```
% switch01.m
clear all
k=input('Input month : ');
switch k
    case {2, 3, 4}
        fprintf('The season is spring !');
    case {5, 6, 7}
        fprintf('The season is summer !');
    case {8, 9, 10}
        fprintf('The season is autumn !');
    case {11, 12, 1}
        fprintf('The season is winter !');
    otherwise
        fprintf('Input 1~12');
end
fprintf('\n');
```

-  **switch01** 執行結果

```
>> switch01
Input month: 6
The season is summer !
>> switch01
Input month : 12
The season is winter !
```

5-5 條件敘述與遞迴函式

MATLAB 函式可以呼叫其他的函式甚至自己本身，重複執行本身函式稱為遞迴函式。函式寫成遞迴的形式需使用遞迴形式定義。定義遞迴形式需先定義遞迴結束情況。例如 k 階層 $k!=1 \times 2 \times 2 \times \dots \times k$ ，定義成遞迴形式則為

$$1! = 1$$

$$k! = (k-1)! \times k$$



factrec.m，利用遞迴方式數入 n 後顯示 $n!$ 的值

```
%factrec.m
function a=factrec()
p=input('Input factorial number: ');
a=fac(p);
end

function fn = fac(n)
if (n > 1)
    fn = n * fac(n-1);
else
    fn = 1;
end
end
```

$$\text{fib}(1)=1$$

$$\text{fib}(2)=1$$

$$\text{fib}(n)=\text{fib}(n-1) + \text{fib}(n-2)$$

| | | | | |
|-------------------------------------|-------------------------------------|-------------------------------------|--|--|
| 呼叫 fac(3) 3xfac(2) | | | | |
| | 呼叫 fac(2) 2xfac(1) | | | |
| | | 呼叫 fac(1) fac(1)=1 | | |

5-12 MATLAB 程式設計基礎

| | | | | |
|------|------|--|--|--|
| | 傳回 2 | | | |
| 傳回 6 | | | | |
| | | | | |
| | | | | |



執行結果

```
>> factrec
Input factorial number: 10
ans =
    3628800
```

再如 fibonacci 數列 1 1 2 3 5 8 13 21 34 55 89 ...，以遞迴形式定義為

fib(1)=1

fib(2)=1

fib(n)=fib(n-1) + fib(n-2)

習題



利用**if**多重條件敘述，當使用者輸入的分數值**>=80**時顯示等第**A**，介於**70~80**間顯示等第**B**，介於**60~70**間顯示等第**C**，低於**60**間顯示等第**D**。
if02grade.m。



利用**for**迴圈計算**1 - 1/2 + 1/3 - 1/4 + 1/5 - 1/6 + 1/7 - 1/8 + 1/9 - 1/10 + ...+... - 1/100**的值。**forsum1.m**。



輸入**n**的值，利用**for**迴圈計算**n!**的值(不使用**prod()**)。**forfact.m**。



輸入**n**的值，利用**while**迴圈計算**n!**的值(不使用**prod()**)。**whilefact.m**。



使用**cputime**或**tic, toc**指令，利用遞迴方式顯示**20!**的值。並傳回**cpu**運算

的時間。



利用遞迴方式數入 n 後顯示`fibonacci(n)`的值。注意重設遞迴數目可以使用指令`set(0,'RecursionLimit',N)`。 N 為函式遞迴的上限。