```
%====================================================================
======
% Sec 5.1 some command line for the ralationshop operation
% please check the table 5.1 (relation operator)
% & 5.2 (Logical operator)
%====================================================================
======

clc;clear;
r = rand(1,5);
  % output is a logical array with '1' (true) or '0' (false)
x=(r <= 0.5)

r1=((rand(4,4).*3)-1.5);
% r1 =
%
%    0.9442    0.3971    1.3725    1.3715
%    1.2174   -1.2074    1.3947   -0.0439
%   -1.1190   -0.6645   -1.0272    0.9008
%    1.2401    0.1406    1.4118   -1.0743
% set the upper-bound(1) & lower-bound (-1)of the matrix
% generate a mask Mp & Mn
Mp=(r1>1); % generate a logoical matrix for > 1.0
rr1=ones(size(r1));
r1=r1.*~Mp+rr1.*Mp;

% r1 =
%
%    0.9442    0.3971    1.0000    1.0000
%    1.0000   -1.2074    1.0000   -0.0439
%   -1.1190   -0.6645   -1.0272    0.9008
%    1.0000    0.1406    1.0000   -1.0743

% Exer do it for the lower-bound (-1)

r = 1:5;
x=r <= 3 % assign the result to the array variable x
```

```
% note (1) a & b must be the same dimension (2) it is different with '=' operator
a = 1:5;
b = [0 2 3 5 6];
a == b

% (z > 0) generate a logical array (element (0,1) )with the same dimension with y
clear all;
z=[ 1 2 -1; 0 1 -3 ; 1 1 -5];
z1=(z> 0) % check the data type of z1
z = z .* z1 % using the Logical array as a mask for the math operations

x = 0 : pi/20 : 3 * pi;
y = sin(x);
y = y .* (y > 0); % set negative values of sin(x) to zero
figure, plot(x, y) % check the figure 5.1

% 5.1.3 To avoid division by error
x = -4*pi : pi / 20 : 4*pi;
y = sin(x) ./ x;      % division by error at x(81)
figure, plot(x, y)

% resolve the problem by generate a mask using relation (x==0)
x = -4*pi : pi/20 : 4*pi;
x = x + (x == 0)*eps; % adjust x = 0 to x = eps
y = sin(x) ./ x;
figure, plot(x, y)


x = -3/2*pi : pi/100 : 3/2*pi;
y = tan(x);
figure, plot(x, y)


% 5.1.4 Avoid the infinity
x = -3/2*pi : pi/100 : 3/2*pi;
y = tan(x);
y = y .* (abs(y) < 1e10); % remove the big ones
figure, plot(x, y)
```

```
%% Counting the random number with (value >=0.5)
tic % start
a = 0; % number >= 0.5
b = 0; % number < 0.5
for n = 1:5000
        r = rand; % generate one number per loop
        if r >= 0.5
                a = a + 1;
        else
                b = b + 1;
        end;
end;
t = toc; % finish
disp( ['less than 0.5: ' num2str(a)] )
disp( ['time: ' num2str(t)] )

r = rand(1,5000)
sum( r < 0.5 ) % it should close to 2500

%% Exercise : (1) Rolling dice in p. 114 : plot the probability of outcome d==6 with #
of trials
% (2) Use the following score program or randomly generate score between (0,100)
%to find the "(number) of student"
% of the following ranges: (100 - 80) (80 - 70) (70 -60 ) (under 60)
% evaluate the average value of each range

% input the number of the student
clear all;close all;
N=input('       number of student:       ');
score=zeros(2,N);
% input the name and score of the student evaluate the average score
for i=1:N
        str1= input('student name:','s');
        eval(['name',int2str(i),'=str1;']);
%          if (i==1)
%                  name=str1;
%          else
%                  name=char(name,str1); % Create a character array.
```

```
%          end
score(1,i)=input('math score:      ');
score(2,i)=input('english score:      ');
avg(i)=(score(1,i)+score(2,i))/2; % avg(i) = sum(score(:,i))/2;
end

% output value
for i=1:N
        eval(['str1=name',int2str(i),';']);
  fprintf('the average score of %s is %3.2f \n',str1,avg(i));
end
save score_data N score

%% 5,2 Logical operator
% Check Table 5.2 in textbook p. 115 for the three logical operators
% these two results are different ???
~ 0 & 0

~ (0 & 0)



% never wrong by using brackets
a=5; b=3; c=-5; final =65
(b * (b == 4) * a * c) & (a ~= 0) % result only two cases: =0 (F)    or ~= 0 (T)
% final=50;
final=65;
(final >= 60) & (final < 70) % two relationship operations
(a ~= 0) | (b ~= 0) | (c ~= 0)
~((a == 0) & (b == 0) & (c == 0))

%% check the table 5.3 for the operator precedence in p. 116

2 > 1 & 0

~(~[1 2 0 -4 0])

% in-class Exerxise in textbook p.117
```

```
%% 5.3 subscripting using logical vectors
a = [-2 0 1 5 9];
b=a([5 1 3])    % inside the [ ] is the index address of the matrix a


v=[5 1 3];
a(v)


clc;clear;
a = [-2 0 1 5 9];
% x1 & x2 is a logical vector for the subscripting of matrix a, note same dim.
x1=logical([0 1 0 1 0])
% [0 1 0 1 0] is a numerical array, logical([0 1 0 1 0]) is a logical array,
%
x2=(a>=0)    % same as before, x2 is a logical array and can be used as a mask
b=a(x1) % extract some elements of the matrix a
c=a(x2) % extract some elements of the matrix a

a(logical([1 1 1 0 0]))
a(logical([0 0 0 0 0]))

a = [-2 0 1 5 9];
b=(a >= 0)
a=a+(a >= 0)
x=a(b)

a = a(a >= 0)

% Is a logical vectos or not
a = [-2 0 1 5 9];
islogical(a > 0) % (a>0) create a logical vector


islogical([0 0 1 1 1]) % a numerical array
```

```
%% 5.4 Logical function Check the table 5.4 : functions: any, all, find

ind=find(a>0)

a = [-2 0 1 5 9];
ind=find(a>0)
a(ind)=1

a = [-2 0 1 5 9];
find(a)
a = a( find(a) ) % find(a) return a subscripts of matrix a with nonzero value

x = [8 1 -4 8 6];
find(x >= max(x))

b = 0/0
c = 6/0
x=[c b 0 1 8 9]
isinf(x)
isnan(x)
x(isnan(x)) = [ ]

isempty(x)
y=[]
isempty(y)


%================================================================
======
%================================================================
======


% Income tax the old-fashioned way
inc = [5000 10000 15000 30000 50000];
for ti = inc
    if ti < 10000
        tax = 0.1 * ti;
```

```
        elseif ti < 20000
                tax = 1000 + 0.2 * (ti - 10000);
        else
                tax = 3000 + 0.5 * (ti - 20000);
        end;
      format compact;
      disp( [ti tax] )
end;

format short

% Income tax the logical way
inc = [5000 10000 15000 30000 50000];
tax = 0.1 * inc .* (inc <= 10000); % (inc <= 10000) creat an logical vector [1 1 0 0 0]
%    (inc > 10000 & inc <= 20000) creat an logical vector [0 0 1 0 0]
tax = tax + (inc > 10000 & inc <= 20000).* (0.2 * (inc-10000) + 1000);
tax = tax + (inc > 20000) .* (0.5 * (inc-20000) + 3000);
disp( [inc' tax'] );

%% Exercise 5.5 & 5.7 in textbook p. 125
% 5.5 sum( (salary >32000 ) .*employees ) ; salary levels are above
% 5.7 units = [200 500 700 1000 1500];cost = cost + 0.02 * (units <= 500) .* units;
%=================================================================
======
%=================================================================
======
```

Table 5.1: Relationship operators

關係運算元符號	說　明
>	大於
>=	大於等於
<	小於
<=	小於等於
==	等於
~=	不等於

Table 5.2 Logical operators

<table>
<tr><th>邏輯運算元符號</th><th>說　明</th></tr>
<tr><td>&</td><td>AND (且)</td></tr>
<tr><td>|</td><td>OR (或)</td></tr>
<tr><td>~</td><td>NOT (非)</td></tr>
</table>

Table 5.4 Logical functions

邏輯判斷函數	用法說明
any(x)	若向量 x 中有任何一個元素不等於 0，則回傳 1。
any(A)	矩陣 A 中任一行向量不等於 0 向量，則回傳 1 給該行向量。
all(x)	若向量 x 中所有元素均不為 0，則回傳 1。
all(A)	矩陣 A 中任一行向量中均不為 0 向量，則回傳 1 給該行向量。
find(邏輯條件)	找出符合指定邏輯條件元素的位置指標。

Table 5.3 in text p.116

Table 5.3 Operator Precedence (See **Help** on `operator precedence`)

Precedence	Operators
1.	( )
2.	^ .^ ' .' (pure transpose)
3.	+ (unary plus) - (unary minus) ~ (NOT)
4.	* / \ .* ./ .\
5.	+ (addition) - (subtraction)
6.	:
7.	> < >= <= == ~=
8.	& (AND)
9.	| (OR)

Table 6.1 (I) : Functions for matrix operation

指　令	說　明
flipud(A)	矩陣上下顛倒。
fliplr(A)	矩陣左右顛倒。
rot90(A)	旋轉 90 度 (逆時針)。
rot90(A,k)	旋轉 90×k 度 (逆時針)，k 為整數。
reshape(A, m, n)	重定矩陣 A 為 m 列 n 行矩陣。注意：m 與 n 為整數引數，其必須滿足 m×n 為原 A 矩陣元素總數之關係。
diag(A)	取對角線 (diagonal) 元素所形成之向量。
triu(A)	取出矩陣 A 之右上部，其餘元素設定為 0，形成一個上三角矩陣。

Table 6.1 (II) : Generate the special matrix

指　令	說　明
eye(n, m)	n×m 單位矩陣 (identify matrix)。
eye(n)	n×n 單位矩陣。
ones(n, m)	n×m 常數矩陣，元素全部為 1。
ones(n)	n×n 常數矩陣，元素全部為 1。
zeros(n, m)	n×m 常數矩陣，元素全部為 0。
zeros(n)	n×n 常數矩陣 ，元素全部為 0。
rand(n, m)	亂數所形成之 n×m 的矩陣。亂數在 0 與 1 間均勻分布 (uniform distribution)。
randn(n)	亂數所形成之 n×n 的矩陣。亂數成常態分配 (normal distribution)。
pascal(n)	產生一個 n×n 的 Pascal 矩陣。
magic(n)	形成一個 n×n (n > 2) 的魔術矩陣 (magic matrix)。
compan(p)	建構多項式 p 的伴隨矩陣 (companion matrix)。
cat(n, A, B, C, ⋯)	輸入 n 維矩陣。即，將 n-1 維矩陣 A, B, C 等，疊成為 n 維矩陣。

- The logical functions `any` and `all` return scalars when taking vector arguments, and are consequently useful in `if` statements.
- Logical vectors may often be used instead of the more conventional `elseif` ladder. This provides faster more elegant code, but requires more ingenuity and the code may be less clear to read later on.

EXERCISES

5.1 Determine the values of the following expressions yourself before checking your answers using MATLAB. You may need to consult Table 5.3.
   (a) `1 & -1`
   (b) `13 & ~(-6)`
   (c) `0 < -2|0`
   (d) `~[1 0 2] * 3`
   (e) `0 <= 0.2 <= 0.4`
   (f) `5 > 4 > 3`
   (g) `2 > 3 & 1`

5.2 Given that `a = [1 0 2]` and `b = [0 2 2]` determine the values of the following expressions. Check your answers with MATLAB.
   (a) `a ~= b`
   (b) `a < b`
   (c) `a < b < a`
   (d) `a < b < b`
   (e) `a | (~a)`
   (f) `b & (~b)`
   (g) `a(~(~b))`
   (h) `a = b == a` (determine final value of `a`)

5.3 Write some MATLAB statements on the command line which use logical vectors to count how many elements of a vector `x` are negative, zero or positive. Check that they work, e.g., with the vector

```
[-4  0  5  -3  0  3  7  -1  6]
```

5.4 The Receiver of Revenue (Internal Revenue Service) decides to change the tax table used in Section 5.5 slightly by introducing an extra tax bracket and changing the tax-rate in the third bracket, as shown in the table on the next page.
Amend the logical vector script to handle this table, and test it on the following list of incomes (dollars): 5000, 10 000, 15 000, 22 000, 30 000, 38 000 and 50 000.

5.5 A certain company offers seven annual salary levels (dollars): 12 000, 15 000, 18 000, 24 000, 35 000, 50 000 and 70 000. The number of employees paid at each level are, respectively: 3000, 2500, 1500, 1000, 400, 100 and 25. Write some statements at the command line to find the following:

Taxable income	Tax payable
$10 000 or less	10% of taxable income
Between $10 000 and $20 000	$1000 + 20% of amount by which taxable income exceeds $10 000
Between $20 000 and $40 000	$3000 + 30% of amount by which taxable income exceeds $20 000
More than $40 000	$9000 + 50 per cent of amount by which taxable income exceeds $40 000

(a) The average salary *level*. Use `mean`. (Answer: 32 000)
(b) The number of employees above and below this average salary level. Use logical vectors to find which salary levels are above and below the average level. Multiply these logical vectors element by element with the employee vector, and `sum` the result.
(Answer: 525 above, 8000 below)
(c) The *average salary earned* by an individual in the company (i.e., the total annual salary bill divided by the total number of employees). (Answer: 17 038.12).

5.6 Write some statements on the command line to remove the largest element(s) from a vector. Try it out on `x = [1 2 5 0 5]`. The idea is to end up with `[1 2 0]` in `x`. Use `find` and the empty vector `[ ]`.

5.7 The electricity accounts of residents in a very small rural community are calculated as follows:

- if 500 units or less are used the cost is 2 cents per unit;
- if more than 500, but not more than 1000 units are used, the cost is $10 for the first 500 units, and then 5 cents for every unit in excess of 500;
- if more than 1000 units are used, the cost is $35 for the first 1000 units plus 10 cents for every unit in excess of 1000;
- in addition, a basic service fee of $5 is charged, no matter how much electricity is used.

The five residents use the following amounts (units) of electricity in a certain month: 200, 500, 700, 1000 and 1500. Write a program which uses logical vectors to calculate how much they must pay. Display the results in two columns: one for the electricity used in each case, and one for amount owed. (Answers: $9, $15, $25, $40, $90)

APPENDIX 5.A SUPPLEMENTARY MATERIAL

Supplementary material related to this chapter can be found online at http://dx.doi.org/10.1016/B978-0-08-100877-5.00006-2.