```
%=====================================================================
======
% 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;
%=================================================================
======
%=================================================================
======
```