WHILE-LOOP

WHILE …. END LOOP!!!

```
a = 1;
b = .25;
steps = 0;
while a < 10
    a = a + a^b;
    steps = steps + 1;
end
a
Steps

a =
    10.9475
steps =
     7
```

```
goal = 100;
x = 1;
while x ~= 100
      x^2
      x = x + 2;
end
```

Matlab will get stuck because this loop is endless

WHILE …. END LOOP!!!

Solution:

```
goal = 100;
x = 1;
steps = 1
while x ~= 100
      x^2
      x = x + 2;
      steps = steps + 1;
      if steps > 100
            break
      end
end
```

```
steps =
     1
ans =
     1
ans =
     9
ans =
    25
ans =
    49
ans =
    81
% ... output omitted
ans =
       38025
ans =
       38809
ans =
       39601
```

WHILE …. END LOOP!!!

```
clc
outputForTesting = true;
while outputForTesting
    candidate = [randperm(3) randperm(3) randperm(3)];
    if candidate(3) ~= candidate(4) & …
         candidate(6) ~= candidate(7);
        outputForTesting = false;
    end
    if outputForTesting
        badcandidate = candidate
    end
end
goodsequence = candidate
```

```
badcandidate =
     3     1     2     2     3     1     1     2     3


goodsequence =
     2     1     3     1     3     2     3     1     2
```

FOR LOOP TAKES LONG TIME TO EXECUTE

```
% Part 1: Generate numbers using RANDN
clc
close all
clear
tic
r = randn(1000,1000);
SecondsToGenerateMillionRandom_Directly = toc

% Part 2: Generate numbers using FOR, without preallocation
clear r
tic
for ii = 1:1000
    for jj = 1:1000
        r(ii,jj) = randn(1,1);
    end
    t(ii) = toc;
end
SecondsToGenerateMillionRandom_Forloops = toc
```

FOR LOOP TAKES LONG TIME TO EXECUTE

```
% Part 3: Generate numbers using FOR, with preallocation
clear r
tic
r = zeros(1000,1000);
for ii = 1:1000
    for jj = 1:1000
        r(ii,jj) = randn(1,1);
        tpa(ii) = toc;
    end
end
SecondsToGenerateMillionRandom_Preallocated_ThenForLoops = toc
```

```
SecondsToGenerateMillionRandom_Directly =
    0.0226

SecondsToGenerateMillionRandom_Forloops =
    1.4035

SecondsToGenerateMillionRandom_Preallocated_ThenForLoops =
    1.5100
```

TIPS FOR MINIMIZING USAGE OF FOR-LOOP

INSTANT IF-ING

```
a = [12 15 17 13 15 12 14];
b  =  ( a  > 13 );
```

```
b =
     0     1     1     0     1     0     1
```

```
a = [12 15 17 13 15 12 14];
b  =  ( a  > 13 );
indices_of_good_values_in_a = find(b)
the_good_values_themselves = a(indices_of_good_values_in_a)
```

```
indices_of_good_values_in_a =
     2     3     5     7
the_good_values_themselves =
    15    17    15    14
```

```
a = [12 15 17 13 15 12 14];
the_good_values_instantly  = a( a>13 )
```

```
the_good_values_instantly =
    15    17    15    14
```

INSTANT IF-ING

```
m1 = [
    16    13     3     2
     9    12     6     7
     5     8    10    11
     4     1    15    14];
m2 = [
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1];
cells_inwhich_m1_equals_m2 = (m1 == m2)
indices_of_the_equal_values = find(m1 == m2)
the_values_that_are_equal = m1(m1 == m2)

not_m1_equals_m2 = ~(m1 == m2)
m1_notequal_m2 = (m1 ~= m2)
```

INSTANT IF-ING

```
cells_inwhich_m1_equals_m2 =
     1     0     1     0
     0     0     0     0
     0     0     0     0
     1     0     1     0
indices_of_the_equal_values =
     1
     4
     9
    12
the_values_that_are_equal =
    16
     4
     3
    15
```

```
not_m1_equals_m2 =
     0     1     0     1
     1     1     1     1
     1     1     1     1
     0     1     0     1
m1_notequal_m2 =
     0     1     0     1
     1     1     1     1
     1     1     1     1
     0     1     0     1
```

USING 'FIND' COMMAND

```
h=randperm(11)+10
h==12|h==16
find(h==12|h==16)
values_sought = h(find(h==12|h==16))
```

```
h =
    18    14    20    19    21    15    13    11    17    16    12

ans =
     0     0     0     0     0     0     0     0     0     1     1

ans =
    10    11

values_sought =
    16    12
```

APPLYING CONTINGENCIES

```
rng('shuffle')
clc
sequenceOf1sAnd2s = [ones(1,6) ones(1,6)*2];
done = false;
cycles = 0;

while not(done)
    cycles = cycles + 1;
    done = true;
    sequenceOf1sAnd2s = sequenceOf1sAnd2s(randperm(12));
    for i = 4:12
        % Detect any runs of 1's or 2's
        if    sequenceOf1sAnd2s(i) == sequenceOf1sAnd2s(i-1)...
            & sequenceOf1sAnd2s(i) == sequenceOf1sAnd2s(i-2)...
            & sequenceOf1sAnd2s(i) == sequenceOf1sAnd2s(i-3)
        done = false;
        end
    end
end

sequenceOf1sAnd2s
cycles
```

EXERCISE

2. Make a fake data from a psychophysical experiment. Your code should start from rng('default'). The name of the data should be 'rtData'. rtData is 50x2 matrix, where the first column contains a mean response time of a subject and the second column has proportions the subject was correct.

Mean response time should follow a normal distribution whose mean is 300 ms and standard deviation is 25 ms. The proportion correct should be values from an uniform distribution that are bounded by 0 and 1 (values between 0 and 1)

a. Make an array matrix for showing identities of subjects (transpose([1:50])). Add this array to rtData matrix as the first column. Now rtData is 50x3 matrix.

b. Make two new empty matricies called 'chosen_ones' and 'chosen_ones_data'. Only select subjects whose mean response time is less than 400 (ms), and performance (proportions correct) is higher than 60 % (0.6). Put these selected subjects' data to the 'chosen_ones_data'. Put subject's identities into 'chosen_ones' matrix. Use for-loop for solving this task.

c. Report mean and standard deviation of reaction time of the chosen ones.

d. Do b) again using logical statement instead of for-loop. Report mean and standard deviation of reaction time of the chosen ones, and see if the results are the same with ones from 3).

EXERCISE

3. Write a program to compute the standard error of a uniform distribution(use rand) that has n values (a value you specify for each run of the program). Build in a contingency so you divide the standard deviation by the square root of n if n is greater than or equal to 30, but you divide by the square root of n-1 if n is less than 30.

N개로 이루어진, uniform distribution을 따르는 값들의 표준오차를 구하는 프로그램을 만들어 보라. 조건문을 만들어 만일 주어진 N이 30 보다 크면 만들어진 값의 표준편차를 square root of N 으로 나누어 주고, 30보다 작으면 만들어진 값들의 표준편차를 square root of N-1로 나누어 주도록 하시오.

INPUT-OUTPUT

GETTING INPUT FROM KEYBOARD

GETTING VALUES

```
favorite = -inf;
while (favorite < 2) | (favorite > 7)
    favorite = ...
    input('What is your favorite
number between 2 and 7? ')
end
disp('OK, got it!')
```

```
What is your favorite number between 2 and 7? 1
favorite =
     1
What is your favorite number between 2 and 7? 10
favorite =
    10
What is your favorite number between 2 and 7? 0
favorite =
     0
What is your favorite number between 2 and 7? 4
favorite =
     4
OK, got it!
```

GETTING REACTION TIME!

```
tic
response = input('What is five plus the square root of 64? ')
Reaction_Time = toc
```

```
What is five plus the square root of 64?

response =
    13

Reaction_Time =
    1.9696
```

FORMATTING NUMBERS

```
t = [-.5:.5:1]';

format bank
bank_format_t = t

format compact
compact_t = t

format rat
rational_format_t = t

format short
short_format_t = t
```

Getting current format
>> get(0, ‘Format’)
>> get(0, ‘FormatSpacing’)

```
format shortG
short_g_format_t = t

format long
long_format_t = t

format longG
long_g_format_t = t

format loose
loose_t = t

format   % return format to standard default
standard_format_t = t
```

FORMATTING NUMBERS

```
bank_format_t =
         -0.50
             0
          0.50
          1.00

compact_t =
         -0.50
             0
          0.50
          1.00

rational_format_t =
      -1/2
       0
       1/2
       1

short_format_t =
   -0.5000
         0
    0.5000
    1.0000
```

```
short_g_format_t =
          -0.5
             0
           0.5
             1

long_format_t =
  -0.500000000000000
                   0
   0.500000000000000
   1.000000000000000

long_g_format_t =
                    -0.5
                       0
                     0.5
                       1
```

```
loose_t =
                                -0.5
                                   0
                                 0.5
                                   1

standard_format_t =
   -0.5000
         0
    0.5000
    1.0000
```

IMPORTING DATA FROM KEYBOARD INPUT

SPRINTF

```
name = input('What is your name? ', 's');
greeting = sprintf('Hello, %s, I will try to help you.', name);
greeting
```

What is your name? Joonyeol

greeting =
Hello, Joonyeol, I will try to help you.

```
piVal = sprintf('The approximate value of %s is %f', 'pi', pi);
```

piVal =

The approximate value of pi is 3.141593

%d: integer output
%e: scientific notation
%f: floating point (or decimal) output
\n: return will be included in a string
\t: tab character will be included in a string

SPRINTF

```
twoLines = sprintf('two\nlines')
```

```
twoLines =
two
Lines
```

```
effort = sprintf('Let''s give %d%% effort to the project!!!', 100)
```

```
effort =
Let's give 100% effort to the project!!!
```

WRITING OUTPUT DATA TO MONITOR DISPLAY OR FILE

FPRINTF

```
fprintf('%s\n','Matlab can be fun.');


Pi_matrix = linspace(pi,2*pi,10);
fprintf('%6.0f', Pi_matrix);
fprintf('\n');
fprintf('%6.2f', Pi_matrix);
fprintf('\n');
```

```
     3     3     4     4     5     5     5     6     6     6
  3.14  3.49  3.84  4.19  4.54  4.89  5.24  5.59  5.93  6.28
```

FPRINTF

```
fprintf('%4d', [1:10]);
fprintf('\n\n');
fprintf('%5d', [1:10]);
fprintf('\n\n');
fprintf('%6d', [1:10]);
fprintf('\n\n');
```

1   2   3   4   5   6   7   8   9 10

1   2   3   4   5   6   7   8   9 10

1   2   3   4   5   6   7   8   9 10

```
a = [3.1:5.1];
b = [3:5];
c = a*2;
d = b+2;
fprintf('%6.2f', a); fprintf('%4d', b); fprintf('\n');
fprintf('%6.2f', c); fprintf('%4d', d); fprintf('\n');
```

OPENING FILES

number returned by fopen which you will use to refer to this file

```
fid = fopen(filename, permission)
```

string with name of file or full path to file if it's not in the current directory

string containing code that determines what Matlab is allowed to do with this file

OPENING FILES

- Permission codes
  - ‘r’ open file for reading
  - ‘w’ open file for writing (will create or overwrite)
  - ‘a’ append data to file (will create if doesn’t already exist)

PERMISSION

'r'	읽기 위한 파일을 엽니다.
'w'	쓰기 위한 파일을 열거나 새로 만듭니다. 기존 내용(있는 경우)을 무시합니다.
'a'	쓰기 위한 파일을 열거나 새로 만듭니다. 데이터를 파일 끝에 추가(append)합니다.
'r+'	읽고 쓰기 위한 파일을 엽니다.
'w+'	읽고 쓰기 위한 파일을 열거나 새로 만듭니다. 기존 내용(있는 경우)을 무시합니다.
'a+'	읽고 쓰기 위한 파일을 열거나 새로 만듭니다. 데이터를 파일 끝에 추가(append)합니다.

텍스트 모드에서 파일을 열려면 문자 't'를 permission 인수에 연결하십시오(예: 'rt' 또는 'wt+').

WORKING WITH FILES

- Introducing fopen() and fclose()
- General plan for working with files:

fopen()

<read from file or write to file>

fclose()

WRITING DATA TO NAMED FILES

```
fid = fopen('mydata.txt','wt');
rr = [1.1:5.1];

fprintf(['Data echoed to Command window as it is written'...
     ' to mydata.txt\n'])
fprintf('%6.1f',rr);        % to Command window
fprintf(fid,'%6.1f',rr);    % to file associated with fid

fprintf(fid,'\n');
fprintf('\n');

fprintf('%6.1f',rr+2);
fprintf(fid,'%6.1f',rr+2);
fprintf('\n\n')
fclose(fid);

fprintf('Data as read from mydata.txt:\n')
type mydata.txt
```

WRITING DATA TO NAMED FILES

```
Data echoed to Command window as it is written to mydata.txt
   1.1   2.1   3.1   4.1   5.1
   3.1   4.1   5.1   6.1   7.1

Data as read from mydata.txt:

   1.1   2.1   3.1   4.1   5.1
   3.1   4.1   5.1   6.1   7.1
```

WRITING DATA TO NAMED FILES

```
a= [1 2 3 4 5];
acube = a.^3;
myoutfile = fopen('CubesList.txt','wt');
for i = 1:5
     fprintf(myoutfile,'The cube of %d is %3d\n',a(i),acube(i));
end
fclose(myoutfile);
type('CubesList.txt');
```

```
The cube of 1 is   1
The cube of 2 is   8
The cube of 3 is  27
The cube of 4 is  64
The cube of 5 is 125
```

DIRECTORY COMMANDS, READING FILES

```
ls

dir *.m
dir *.mat
dir *.txt

pwd
cd
```

```
Reading Excel file

M = xlsread('data.xls');

xlswrite('My_Excel_File', M)
```

```
data_from_file = load('mydata.txt')

data_from_file =
    1.1000    2.1000    3.1000    4.1000    5.1000
    3.1000    4.1000    5.1000    6.1000    7.1000
```

```
myinfile = fopen('cubeslist.txt');
nlines = 0;
while true
    thisline = fgetl(myinfile);
    nlines = nlines + 1;
    fprintf('Line %d:  %s\n',nlines,thisline);
    if feof(myinfile)
        disp('all done!')
        break
    end
end
```

MAKING SUBJECT LIST

```
timeofday = clock;
FirstOutput = sprintf('Exp5_%02d_%02d_%02d_T%02d%02d%02d.txt',round(timeofday(1:6)))
inits = input('Subject initials: ','s');
SecondOutput = strcat('Exp5_',inits,'_', sprintf('%02d_%02d_%02d',timeofday(1:3)),'.txt')
```

FirstOutput =

Exp5_2016_01_09_T214505.txt

Subject initials: JL

SecondOutput =

Exp5_JL_2016_01_09.txt

USING NATIVE MATLAB FILE FORMAT

*.mat file

save

Save workspace variables to file.

save(FILENAME) stores all variables from the current workspace in a MATLAB formatted binary file (MAT-file) called FILENAME.

save(FILENAME,VARIABLES) stores only the specified variables.

Ex) save('myData.mat', 'summarydata');

load

Load data from MAT-file into workspace.

S = load(FILENAME) loads the variables from a MAT-file into a structure array, or data from an ASCII file into a double-precision array.

S = load(FILENAME, VARIABLES) loads only the specified variables from a MAT-file.

Ex) load('myData.mat', 'summarydata');

EXERCISE

```
Marginal sums for N = 6

35   1   6  26  19  24  |   111
 3  32   7  21  23  25  |   111
31   9   2  22  27  20  |   111
 8  28  33  17  10  15  |   111
30   5  34  12  14  16  |   111
 4  36  29  13  18  11  |   111
--  --  --  --  --  --
111 111 111 111 111 111
```

EXERCISE

```
N = input('Please specify the dimension of the matrix. N =  ');
if N < 3 | N > 9
    disp('N should be any integer value between 2 and 10');
end

select = input('Do you want to make a magic matrix or uniform random matrix?\nIf
you want a magic matrix, press 1, otherwise, press 2 : ');

switch select
case 1
    M = magic(N);
case 2
    M = randi(20, N, N);
otherwise
    disp('You should press 1 or 2');
end

sumColumn = sum(M, 1);
sumRow = sum(M,2);
```

EXERCISE

```
fid = fopen('matFile.txt', 'wt');
fprintf(fid, 'Marginal sums for N = %d\n\n', N);
for i = 1:N
    for j = 1:N
        fprintf(fid, '%2d\t', M(i,j));
    end
    fprintf(fid, '|\t%2d\n', sumRow(i));
end
for i = 1:N
    fprintf(fid, '--\t');
end
fprintf(fid, '\n');
for i = 1:N
    fprintf(fid, '%2d\t', sumColumn(i));
end
fprintf(fid, '\n');
fclose(fid);
```

EXERCISE

4. You are studying how quickly subjects can learn and you have found that in a particular experimental paradigm, the accuracy of a subject is typically a logarithmic function of the number of trials they have experienced. You have created an equation which characterizes their behavior very neatly and you now want to use your MATLAB expertise to predict how quickly subjects can learn the task. In your experiment, subjects have to choose between 4 different stimuli that are equiprobable, which means that their chance of guessing correctly is ¼, and learning the task allows them to improve their performance above this level. The equation you have discovered is:

p_correct = base_rate + learning_rate*log(trial),

where trial can take on integer values, learning_rate can be any real number between 0 and 1, base_rate equals ¼, and p_correct cannot exceed 1. Write a program that allows you to specify the learning_rate and criterion parameters and then computes for you how many trials it will take for subjects to reach that criterion using this equation.

EXERCISE

5. The following code will generate a set of student data in which column 1 is the student number, column 2 is an integer representing the class year ('15, '16, '17, or '18), and column 3 the student's grade point average (which ranges from 2.0 to 4.0). Since the random number generator is initialized at the beginning, you will get the same sequence as we did, and your checkvalues should agree with the values reported in the comment lines.

```
clear
rng('default')
data(:,1) = randperm(300);
data(:,2) = randi(4,300,1) + 14;
data(:,3) = randi(20,300,1)/10 + 2;
format short
checkvalues = mean(data)
```

checkvalues should be:

150.5000 16.4533 2.9837

Without printing out the data matrix, answer the following: What students (by student number) had a 4.0 average? Who are the seniors (class of '15) who will graduate with honors (GPA >= 3.5)? How many first-year students (class of '18) are likely to elect to be Psychology majors, as predicted by their GPA being greater than 3.0? What is the GPA of student #1 (the student with that student number, not necessarily the first student in the matrix). What is the standard deviation of the GPAs of second-year students (class of '17)?

EXERCISE

6. From the exercise 5, printing out the data matrix, writing the following to a file named ‘studentGPA.txt’: What students (by student number) had a 4.0 average? Who are the seniors (class of '15) who will graduate with honors (GPA >= 3.5)? How many first-year students (class of '18) are likely to elect to be Psychology majors, as predicted by their GPA being greater than 3.0?

Output should be something similar with,

Students with 4.0 average GPA

ID	Yr	GPA
40	18	4.0
34	17	4.0
160	17	4.0
232	18	4.0
259	17	4.0
202	18	4.0
56	16	4.0
298	17	4.0
73	15	4.0