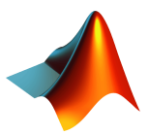
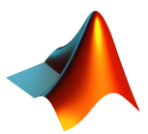


MATLAB 프로그래밍 및 실습

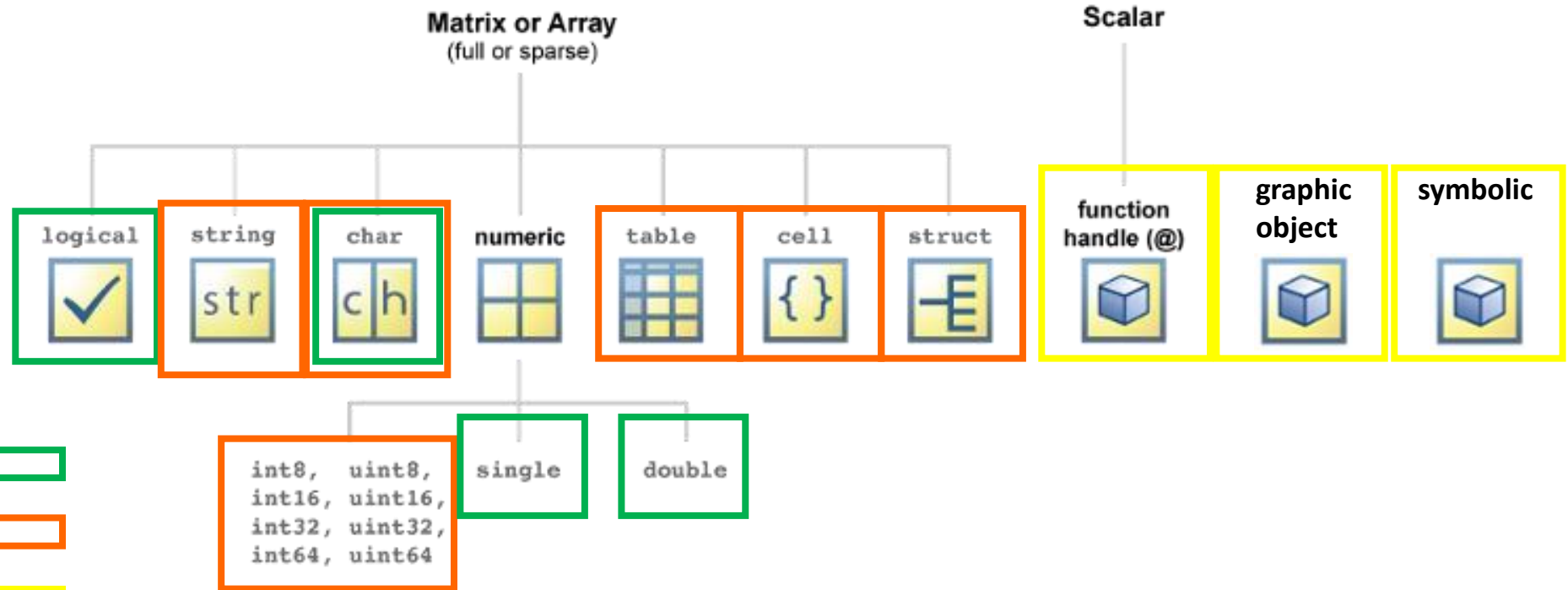
8강. 고급 자료형과 파일입출력



자료형



matlab 기본 자료형



배운거



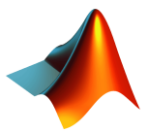
오늘 배울거



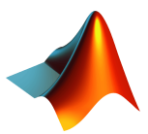
나중에 배울거



* 기본자료형 외에도 자료형은 많음



텍스트 파일 입출력



fprintf의 정체는...?

fprintf ??
↓
Write data to text file

Syntax ?? OK OK OK
↓ ↓ ↓ ↓
fprintf(fileID,formatSpec,A1,...,An)
fprintf(formatSpec,A1,...,An)

nbytes = fprintf(____)

Description ??
↓

fprintf(fileID,formatSpec,A1,...,An) applies the formatSpec to all elements of arrays A1,...,An in column order, and writes the data to a text file. fprintf uses the encoding scheme specified in the call to fopen.

Wait, what?

fprintf(formatSpec,A1,...,An) formats data and displays the results on the screen.

- fprintf의 본래 기능은...
 - 텍스트 파일에 문자열 쓰기
 - fprintf = file + **print** + **format**
 - fileID를 지정하지 않으면 명령창에 출력
- fileID란?

```
fprintf('I am %. %d years old.\n', 'Quill', 38);
```

Hmm. OK

텍스트 파일 쓰기

```
x = 0:0.1:1;
A = [x; exp(x)];

fileID = fopen('exp1.txt','w');

fprintf(fileID,'%6s %12s\n', 'x', 'exp(x)');
fprintf(fileID,'%6.2f %12.8f\n', A);

fclose(fileID);

% same code, using for loop
fid = fopen('exp2.txt','w');

fprintf(fid,'%6s %12s\n','x','exp(x)');
for i=1:size(A,2)
    fprintf(fileID,'%6.2f %12.8f\n', A(1,i), A(2,i));
end

fclose(fid);
```

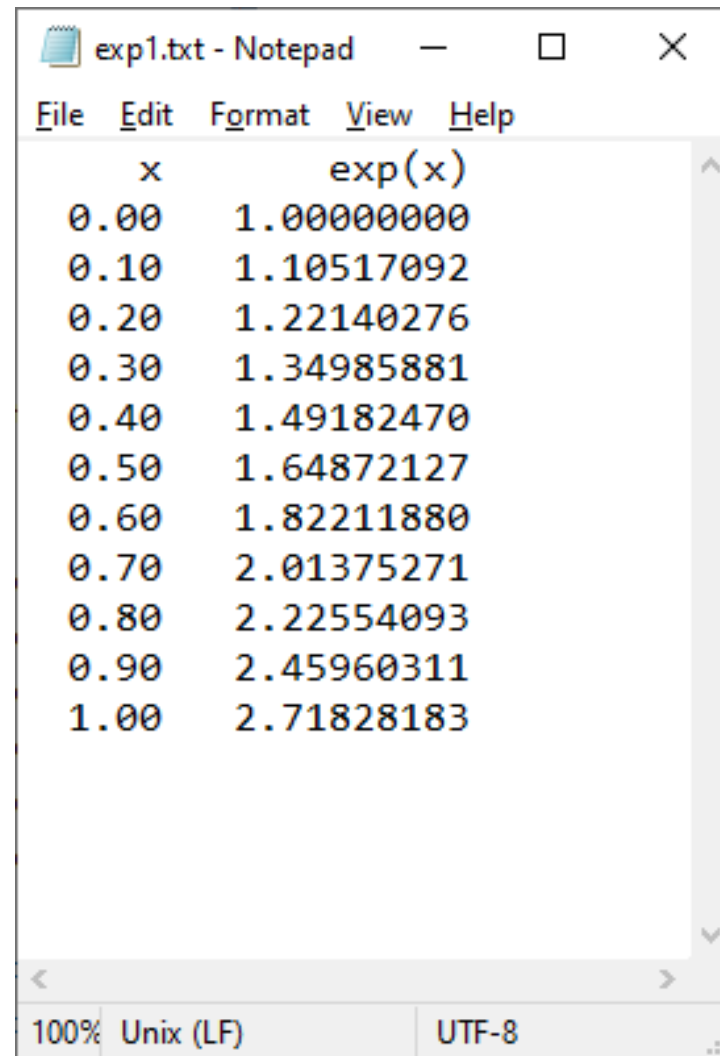
fprintf(fileID, '%6.2f %12.8f\n', A);

↑
텍스트를
쓴다.

↑
fileID에

↑
이러한
포맷으로

↑
A의 값을
(1차원 인덱싱으로)



x	exp(x)
0.00	1.00000000
0.10	1.10517092
0.20	1.22140276
0.30	1.34985881
0.40	1.49182470
0.50	1.64872127
0.60	1.82211880
0.70	2.01375271
0.80	2.22554093
0.90	2.45960311
1.00	2.71828183

format specifier

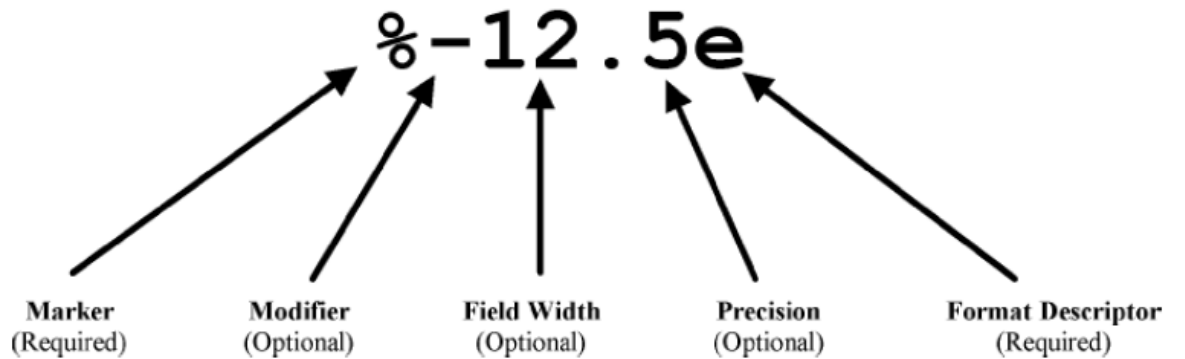


Table B-5 Format Conversion Specifiers for `fprintf`

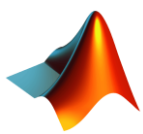
Specifier	Description
<code>%c</code>	Single character
<code>%d</code>	Decimal notation (signed)
<code>%e</code>	Exponential notation (using a lowercase <code>e</code> as in <code>3.1416e+00</code>)
<code>%E</code>	Exponential notation (using an uppercase <code>E</code> as in <code>3.1416E+00</code>)
<code>%f</code>	Fixed-point notation
<code>%g</code>	The more compact of <code>%e</code> or <code>%f</code> (insignificant zeros do not print)
<code>%G</code>	Same as <code>%g</code> , but using an uppercase <code>E</code>
<code>%o</code>	Octal notation (unsigned)
<code>%s</code>	String of characters
<code>%u</code>	Decimal notation (unsigned)
<code>%x</code>	Hexadecimal notation (using lowercase letters <code>a-f</code>)
<code>%X</code>	Hexadecimal notation (using uppercase letters <code>A-F</code>)

Table B-6 Format Flags

Flag	Description
Minus sign (<code>-</code>)	Left-justifies the converted argument in its field (<i>Example:</i> <code>%-5.2d</code>). If this flag is not present, the argument is right-justified.
<code>+</code>	Always print a <code>+</code> or <code>-</code> sign (<i>Example:</i> <code>%+5.2d</code>).
<code>0</code>	Pad argument with leading zeros instead of blanks (<i>Example:</i> <code>%05.2d</code>).

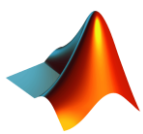
fprintf 예시

Function	Result	Comment
<code>fprintf('%d\n', 123)</code>	---- ---- 123	Display the number using as many characters as required. For the number 123, three characters are required.
<code>fprintf('%6d\n', 123)</code>	---- ---- 123	Display the number in a 6-character-wide field. By default the number is <i>right justified</i> in the field.
<code>fprintf('%6.4d\n', 123)</code>	---- ---- 0123	Display the number in a 6-character-wide field using a minimum of 4 characters. By default the number is <i>right justified</i> in the field.
<code>fprintf('%-6.4d\n', 123)</code>	---- ---- 0123	Display the number in a 6-character-wide field using a minimum of 4 characters. The number is <i>left justified</i> in the field.
<code>fprintf('%16.4d\n', 123)</code>	---- ---- +0123	Display the number in a 6-character-wide field using a minimum of 4 characters plus a sign character. By default the number is <i>right justified</i> in the field.



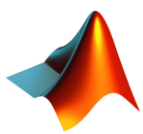
fprintf 예시

Function	Result	Comment
<code>fprintf('%f\n', 123.4)</code>	---- ---- 123.400000	Display the number using as many characters as required. The default case for <code>%f</code> is to display 6 digits after the decimal place.
<code>fprintf('%8.2f\n', 123.4)</code>	---- ---- 123.40	Display the number in an 8-character-wide field, with two places after the decimal point. The number is <i>right justified</i> in the field.
<code>fprintf('%4.2f\n', 123.4)</code>	---- ---- 123.40	Display the number in a 6-character-wide field. The width specification was ignored because it was too small to display the number.
<code>fprintf('%10.2e\n', 123.4)</code>	---- ---- 1.23e+002	Display the number in exponential format in a 10-character-wide field using 2 decimal places. By default the number is <i>right justified</i> in the field.
<code>fprintf('%10.2E\n', 123.4)</code>	---- ---- 1.23E+002	The same but with a capital E for the exponent.

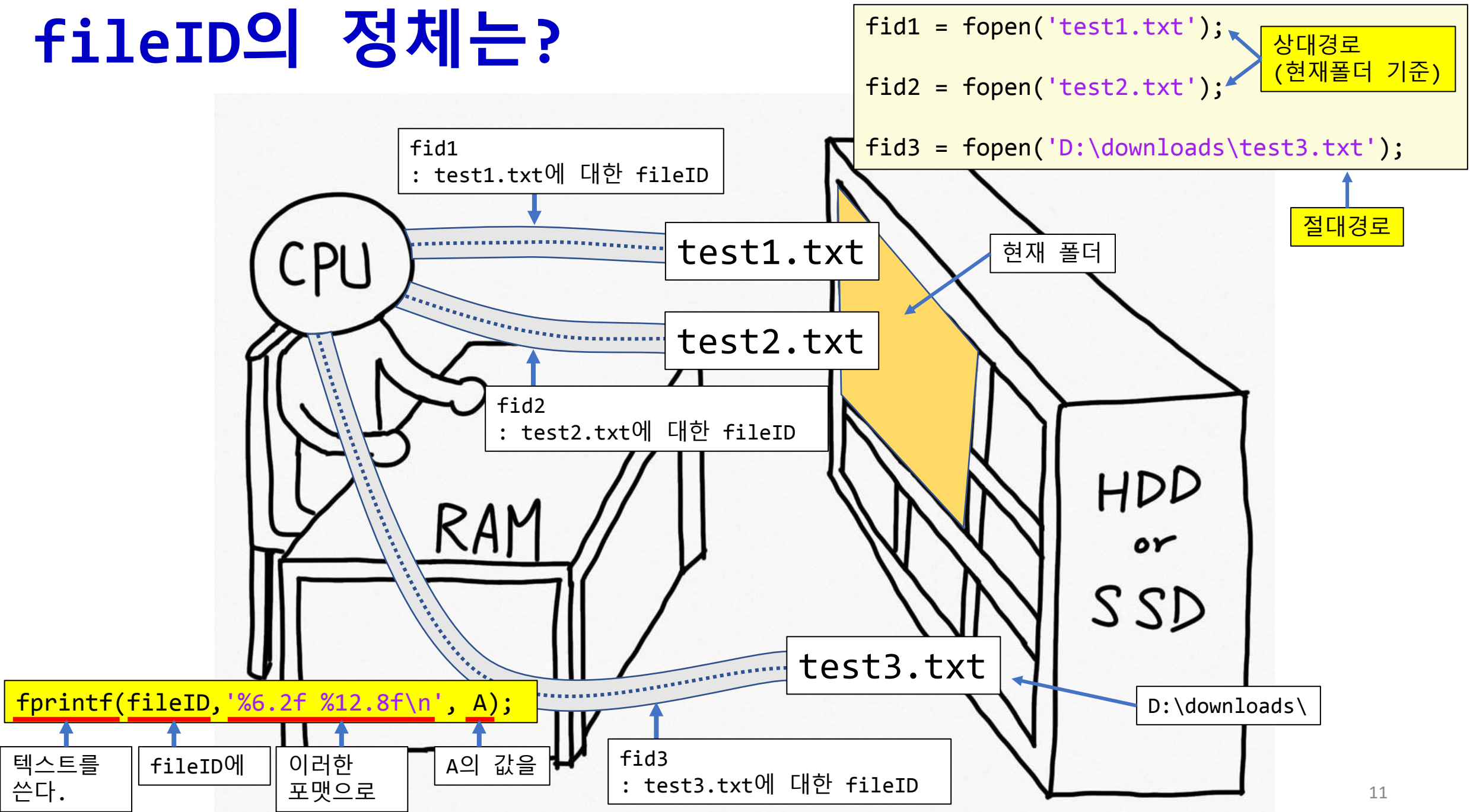


fprintf 예시

Function	Result	Comment
<code>fprintf('%c\n', 's')</code>	---- ---- s	Displays a single character.
<code>fprintf('%s\n', 'string')</code>	---- ---- string	Display the character string.
<code>fprintf('%8s\n', 'string')</code>	---- ---- string	Display the character string in an 8-character-wide field. By default the string is <i>right justified</i> in the field.
<code>fprintf('%-8s\n', 'string')</code>	---- ---- string	Display the character string in an 8-character-wide field. The string is <i>left justified</i> in the field.



fileID의 정체는?



파일 열고 닫기 (fileID 생성, 소멸)

```
x = 0:0.1:1;
A = [x; exp(x)];

fileID = fopen('exp1.txt','w');
fprintf(fileID,'%6s %12s\n', 'x', 'exp(x)');
fprintf(fileID,'%6.2f %12.8f\n', A);

fclose(fileID);

% same code, using for loop
fid = fopen('exp2.txt','w');
fprintf(fid,'%6s %12s\n','x','exp(x)');
for i=1:size(A,2)
    fprintf(fileID,'%6.2f %12.8f\n', A(1,i), A(2,i));
end

fclose(fid);
```

fileID = fopen('exp.txt','w');

fileID를
생성한다.

파일을
열고

이 파일에
대해

쓰기
권한으로

fclose(fileID);

파일을
닫는다.

이 fileID를
갖는

fclose('all');

파일을
닫는다.

열려있는 모든
파일에 대해

※ 파일을 닫아야 하는 이유?

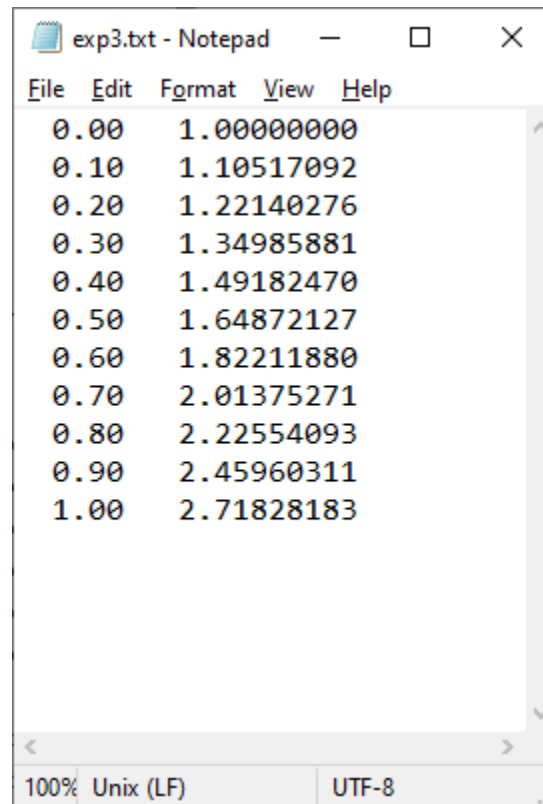
- 닫지 않으면 파일명 변경, 파일 삭제 불가
- fclose의 반환값은 0(성공) 또는 -1(실패)

텍스트 파일 읽기 - fscanf

```
x = 0:0.1:1;
A = [x; exp(x)];

fileID = fopen('exp3.txt','w');
fprintf(fileID,'%6.2f %12.8f\n', A);
fclose(fileID);

% read text file as column vector
fid = fopen('exp3.txt','r');
v1 = fscanf(fid,'%f');
fclose(fid);
```



```
exp3.txt - Notepad
File Edit Format View Help
0.00 1.00000000
0.10 1.10517092
0.20 1.22140276
0.30 1.34985881
0.40 1.49182470
0.50 1.64872127
0.60 1.82211880
0.70 2.01375271
0.80 2.2254093
0.90 2.45960311
1.00 2.71828183
100% Unix (LF) UTF-8
```

```
>> disp(v1)
0
1.0000
0.1000
1.1052
0.2000
1.2214
0.3000
1.3499
0.4000
1.4918
0.5000
1.6487
0.6000
1.8221
0.7000
2.0138
0.8000
2.2255
0.9000
2.4596
1.0000
2.7183
```

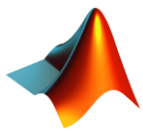
v1 = fscanf(fid, '%f');

v1에 저장한다.
(column
vector로)

파일을
읽어서

fid로부터

실수
포맷으로

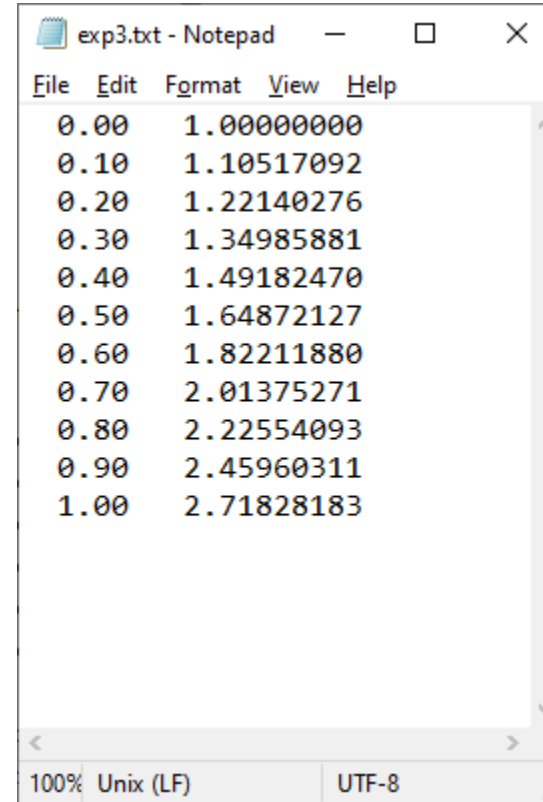


텍스트 파일 읽기 - fscanf

```
x = 0:0.1:1;
A = [x; exp(x)];

fileID = fopen('exp3.txt','w');
fprintf(fileID,'%6.2f %12.8f\n', A);
fclose(fileID);

% read text file with pre-defined array size
fid = fopen('exp3.txt','r');
v2 = fscanf(fid, '%f', [2,11]);
fclose(fid);
```



```
exp3.txt - Notepad
File Edit Format View Help
0.00 1.00000000
0.10 1.10517092
0.20 1.22140276
0.30 1.34985881
0.40 1.49182470
0.50 1.64872127
0.60 1.82211880
0.70 2.01375271
0.80 2.2254093
0.90 2.45960311
1.00 2.71828183
100% Unix (LF) UTF-8
```

```
>> disp(v2)
      0      1.0000
    0.1000    1.1052
    0.2000    1.2214
    0.3000    1.3499
    0.4000    1.4918
    0.5000    1.6487
    0.6000    1.8221
    0.7000    2.0138
    0.8000    2.2255
    0.9000    2.4596
    1.0000    2.7183
```

v2 = fscanf(fid, '%f', [2,11]);

v2에 저장한다.

파일을
읽어서

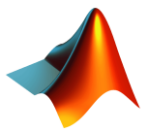
fid로부터

실수
포맷으로

2x11 행렬로 만들어서
(column-first order)

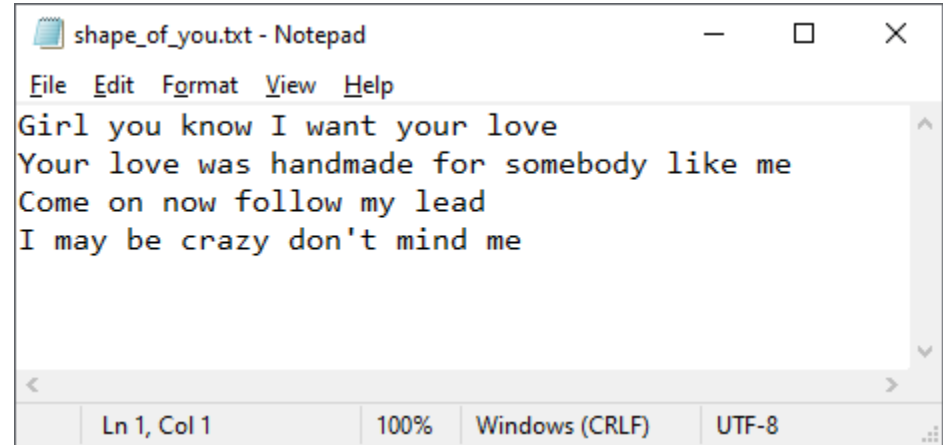
※ transpose가 왜 필요할까?

크기 미지정 -> 끝까지 column vector로
[2, inf] -> 끝까지 2-row matrix로



텍스트 파일 읽기 - fgetl (1=line)

```
fid = fopen('shape_of_you.txt','r');  
while true  
    s = fgetl(fid);  
    if s==-1 % if EOF  
        break  
    end  
    disp(s)  
end  
fclose(fid);
```



shape_of_you.txt - Notepad

File Edit Format View Help

Girl you know I want your love
Your love was handmade for somebody like me
Come on now follow my lead
I may be crazy don't mind me

Ln 1, Col 1 100% Windows (CRLF) UTF-8

Command Window

```
Girl you know I want your love  
Your love was handmade for somebody like me  
Come on now follow my lead  
I may be crazy don't mind me  
fx >>
```

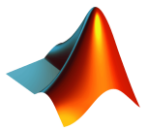
s = fgetl(fid);

s에
저장한다.

한 줄을
읽어서

fid로부터

한 줄 = 줄넘김이 나올 때까지
파일 끝(EOF)에 도달하면 -1을 반환



fileID의 권한 설정

```
% 'r' permission (reading)
fid = fopen('exp3.txt','r'); % 'r' is default
fscanf(fid, '%f', [1,2]) % reading - OK
fprintf(fid, '%6.2f', pi); % can't write
fclose(fid);
```

파일이 존재하지 않으면 에러 발생, fid=-1

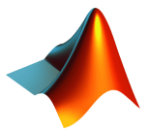
```
% 'w' permission (writing)
fid = fopen('exp3.txt','w');
fprintf(fid, '%6.2f', pi); % delete existing file content, and overwrite
fscanf(fid, '%f') % can't read
fclose(fid);
```

파일이 존재하지 않으면 새로 만듦

```
% 'a' permission (appending)
fid = fopen('exp3.txt','a');
fprintf(fid, '%6.2f\n', pi, exp(1)); % resume writing at the end of file
fscanf(fid, '%f') % can't read
fclose(fid);
```

파일이 존재하지 않으면 새로 만듦

'r'	Open file for reading.
'w'	Open or create new file for writing. Discard existing contents, if any.
'a'	Open or create new file for writing. Append data to the end of the file.



절대경로 vs 상대경로

```
fid = fopen('exp1.txt'); % 상대경로 (현재폴더 기준)
```

```
fid = fopen('D:\examples\exp1.txt'); % 절대경로
```

```
fid = fopen('.\exp1.txt'); % 현재폴더에 있는 exp1.txt 파일 열기
```

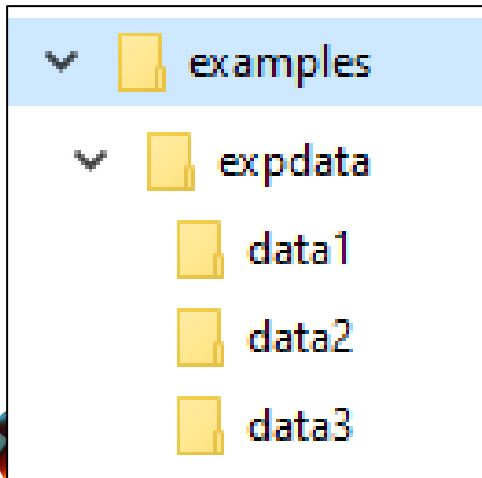
```
fid = fopen('..\data\exp1.txt'); % 현재폴더\data 폴더에 있는 exp1.txt 파일 열기
```

```
fid = fopen('../../data\exp1.txt'); % 현재폴더의 상위폴더 기준 \data\exp1.txt 파일 열기
```

예) 아래와 같이 생성된 실험 데이터를 다루고 싶다면?

: 절대경로 사용 시 상위폴더명을 매번 입력해주어야 함

: 상대경로로 좀 더 손쉽게 파일명 작성 가능



참고1) dir 입력시 나오는 .과 ..
: 각각 현재폴더와 상위폴더를 뜻함
: imread에도 사용 가능

참고2) fileparts

fprintf vs sprintf

```
clipboard('copy', sprintf('\nclear\nclose all\nclc\n\n'));
```

```
x = 0:0.1:1;  
A = [x; exp(x)];  
  
% write A into a char array S, not into a file  
s = sprintf('%6.2f %12.8f\n', A);  
fprintf(s)  
  
name = 'Quill';  
age = 38;  
s = sprintf('I am %s.\n%d years old.\n', name, age);  
fprintf(s)
```

s의 길이를 확인해보자.

Command Window

```
0.00    1.00000000  
0.10    1.10517092  
0.20    1.22140276  
0.30    1.34985881  
0.40    1.49182470  
0.50    1.64872127  
0.60    1.82211880  
0.70    2.01375271  
0.80    2.22554093  
0.90    2.45960311  
1.00    2.71828183
```

I am Quill.

38 years old.

fx >>

s = sprintf('I am %s.\n%d years old.\n', name, age);

s에 저장한다.

문자열을 만들어서

이런 포맷으로

이 변수들을 이용하여

- * \n, \t 등이 포함된 문자열을 미리 생성해둘 수 있음
- * 같은 문자열이 반복되어 사용될 때
해당 문자열을 코드 맨 앞에 만들어두고
코드 뒷부분에서 사용 가능

2강 때 했던 예제를 파일에 써보자.

```
fid = fopen('lec2_example.txt','w');

fprintf(fid,'I am Groot.');
```

`fprintf(fid,'I am Groot.');`

```
fprintf(fid,'\tI am Quill.');
```

`fprintf(fid,'I need a new line.\n');`

```
fprintf(fid,'\t\tThere you go!\n');
```

`name = 'Quill'; age = 38;`

```
fprintf(fid,'I am %s. %d years old.\n', name, age);
```

`weight_old = 136.078;`

`weight_now = 104.326;`

```
fprintf(fid,'I weighed %f kg once.\n',weight_old);
```

`fprintf(fid,'But I am %g kg now.\n',weight_now);`

```
fprintf(fid,'So I lost %g%% of my weight.\n', ...
        100*(1-weight_now/weight_old));
```

`fprintf(fid,'pi is about %5.2f.\n',pi);`

`fprintf(fid,'pi is about %10.2f.\n',pi);`

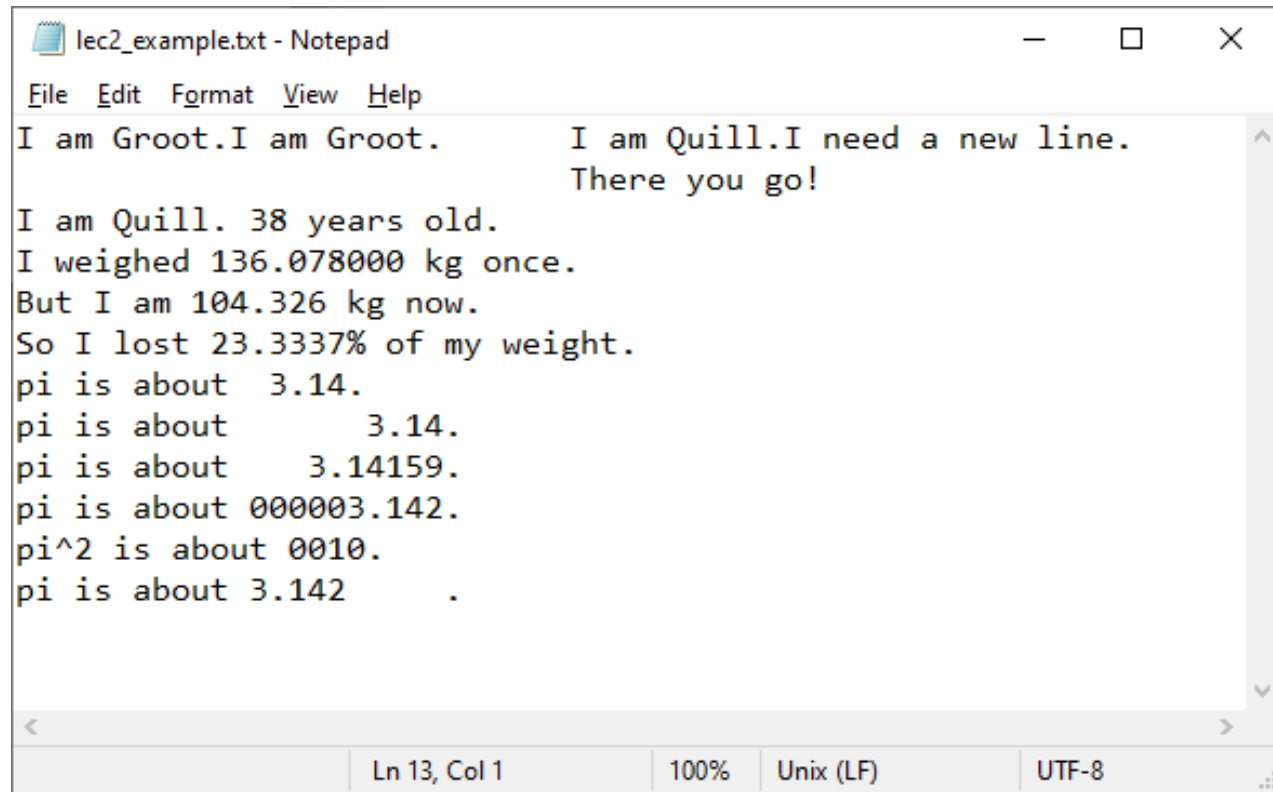
`fprintf(fid,'pi is about %10.5f.\n',pi);`

`fprintf(fid,'pi is about %010.3f.\n',pi);`

```
fprintf(fid,'pi^2 is about %04d.\n', round(pi^2));
```

`fprintf(fid,'pi is about % -10.3f.\n',pi);`

```
fclose(fid);
```

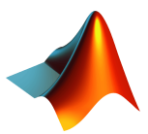


```
lec2_example.txt - Notepad
File Edit Format View Help
I am Groot.I am Groot.      I am Quill.I need a new line.
                             There you go!

I am Quill. 38 years old.
I weighed 136.078000 kg once.
But I am 104.326 kg now.
So I lost 23.3337% of my weight.
pi is about  3.14.
pi is about      3.14.
pi is about    3.14159.
pi is about 000003.142.
pi^2 is about 0010.
pi is about 3.142      .

Ln 13, Col 1    100%    Unix (LF)    UTF-8
```

이미지 다루기 + 정수 자료형



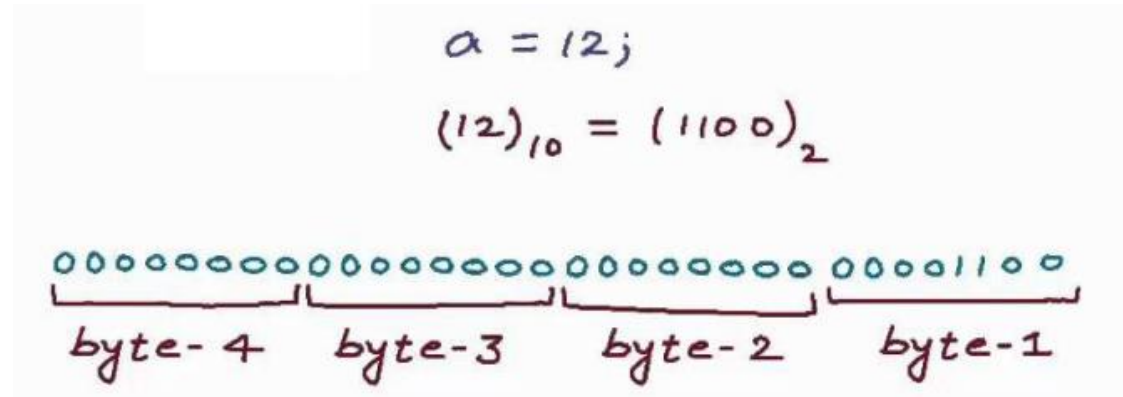
이미지도 행렬이다.



- 실습1.
 - datatip으로 여기저기 찍어본다.
(alt + 클릭 -> datatip 여러 개)
 - 밝은 곳과 어두운 곳의 index를 비교해본다.
 - workspace 상의 행렬 크기와 이미지 크기를 비교해본다.
 - workspace 변수를 직접 열어보고, datatip 값과 비교해본다.
- 실습2.
 - imcrop을 통해 원하는 영역만 crop해본다.
 - 평균 픽셀값을 비교해본다.
- 실습3.
 - 처음보는 자료형 uint8를 발견하고 의아해한다.

uint8...? double...? 우선 자료형 복습

- 정수 12를 저장하는 방법

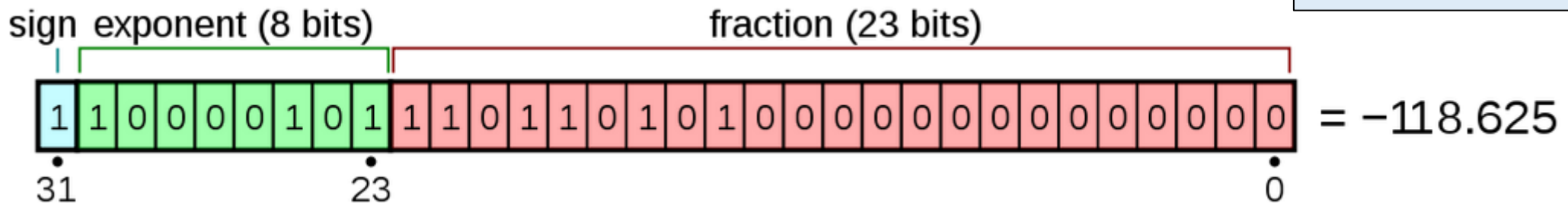


- 실수 -118.625를 저장하는 방법

$$-118.625 \text{ (10진수)} = -1.110110101 \times 2^{110} \text{ (2진수)}$$

	exponent	fraction
single	8 bits	23 bits
double	11 bits	52 bits

* 기본자료형은 double



uint8...?

- 이미지 픽셀값 (픽셀 밝기)

- white = 255 / black = 0
- 그 사이는 0-255 사이의 정수이면 밝기를 표현하기에 충분하다! (총 256단계)
- 컴퓨터가 색을 어떻게 만드는지를 기억해보자. (0 = 0x00, 255=0xFF)

- 정수 자료형은 아래와 같다.

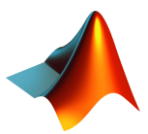
* 실수 자료형

	최소값	최대값	참고
single	-3.4028e+38	3.4028e+38	realmax('single') realmax('double')
double	-1.7977e+308	1.7977e+308	

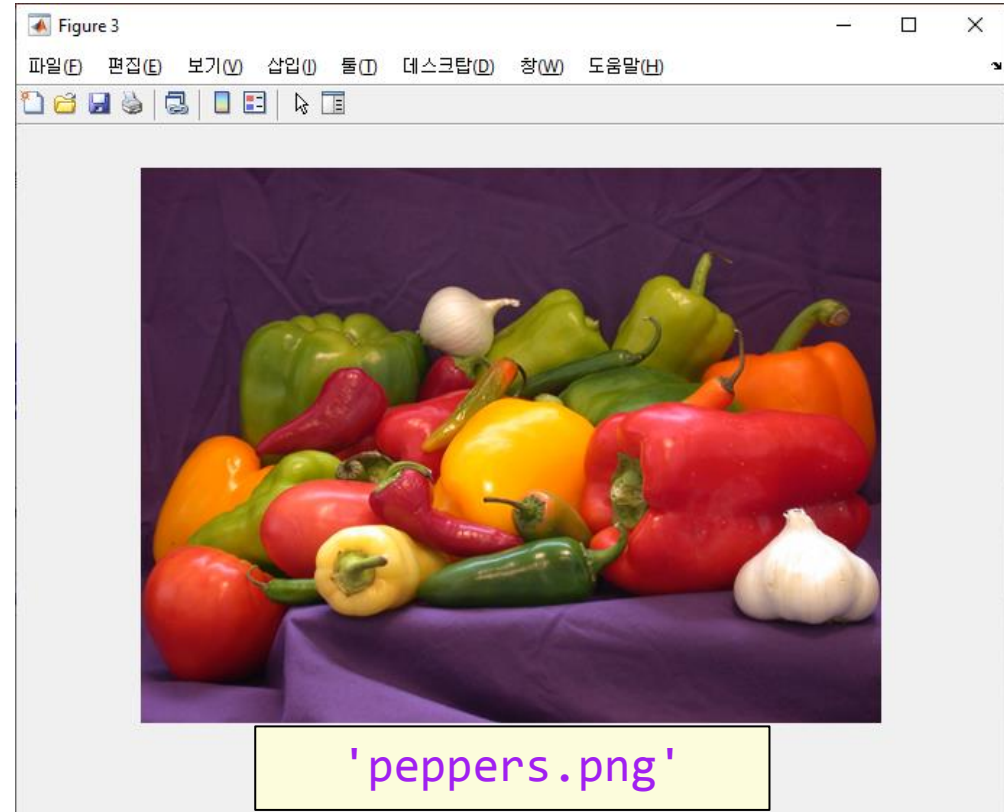
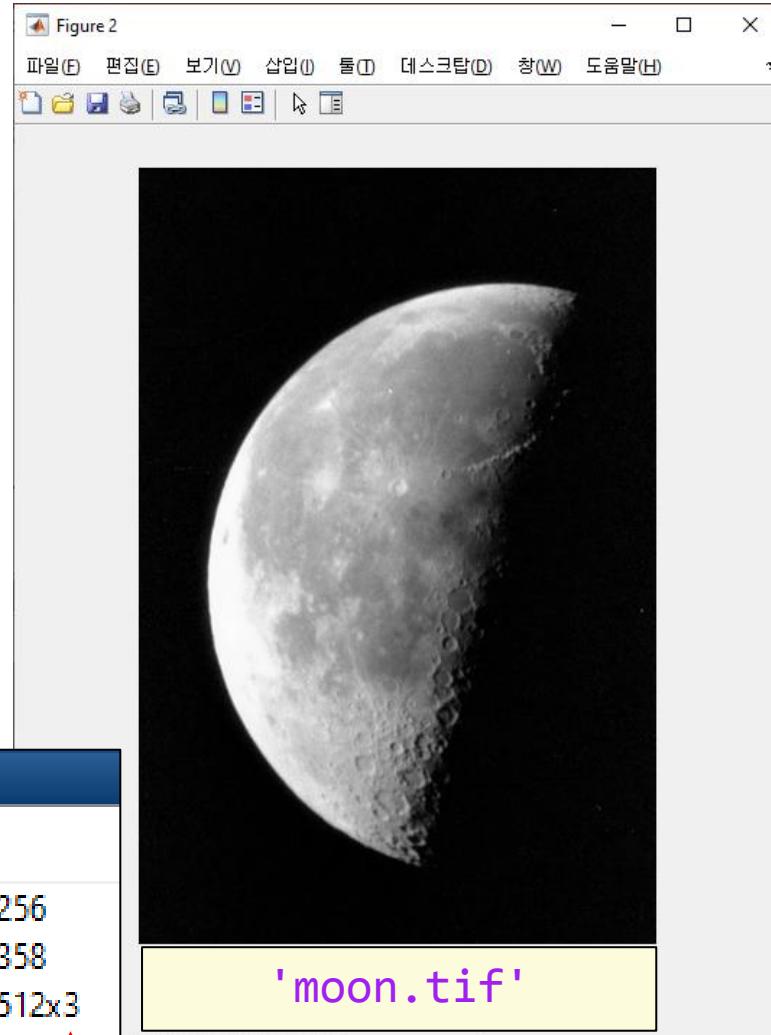
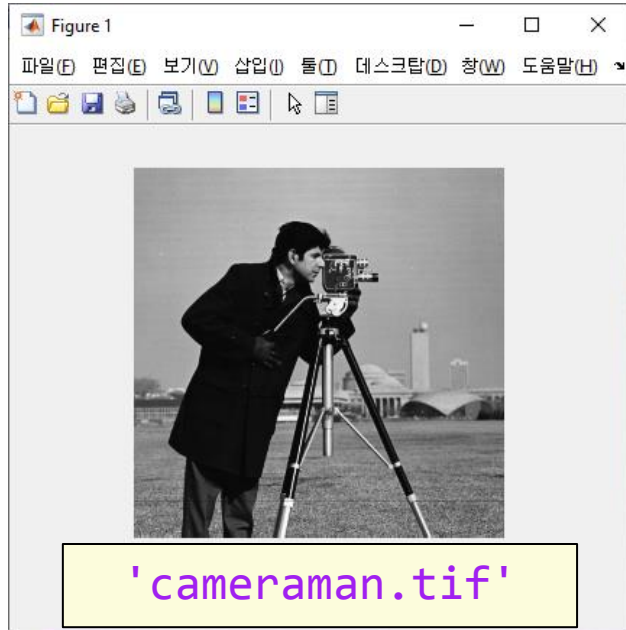
double = 8바이트
uint8 = 1바이트

대부분 이미지
파일의 자료형

자료형	최소값	최대값	자료형 이름은 함수명이기도 하다.
int8	-2^7	2^7-1	<pre>>> uint8(12.45) ans = uint8 12 >> uint8(1000) ans = uint8 255 >> uint16(-100) ans = uint16 0</pre>
int16	-2^{15}	$2^{15}-1$	
int32	-2^{31}	$2^{31}-1$	
int64	-2^{63}	$2^{63}-1$	
uint8	0	2^8-1	
uint16	0	$2^{16}-1$	
uint32	0	$2^{32}-1$	
uint64	0	$2^{64}-1$	

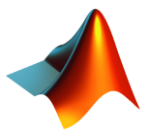


컬러 이미지 vs 흑백 이미지

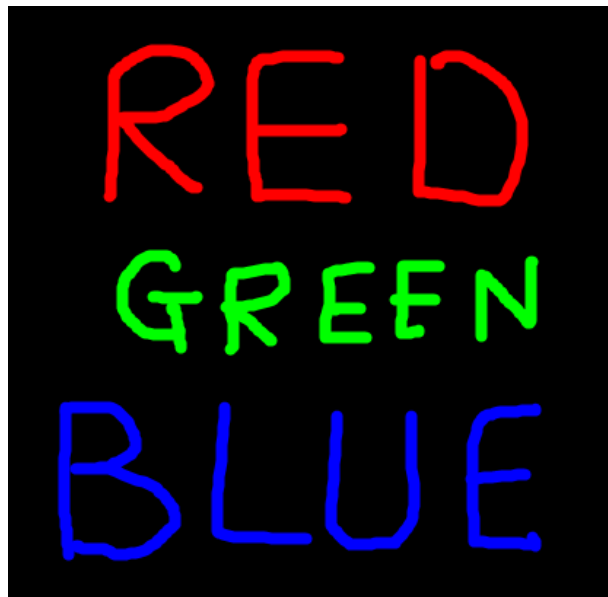


작업 공간		
이름	값	크기
camman	256x256 uint8	256x256
moon	537x358 uint8	537x358
peppers	384x512x3 uint8	384x512x3

?? OK ??



컴퓨터가 색깔을 어떻게 만든다고?



```
img = imread('RGB_test.png');
```

색

RED

GREEN

BLUE

BLACK

HITE

색 조합

RED only

GREEN only

BLUE only

NO COLOR

ALL COLORS

uint8 (0-255)

8비트 조합

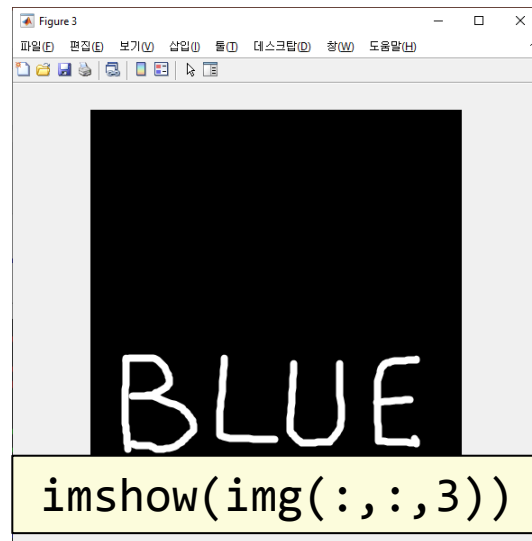
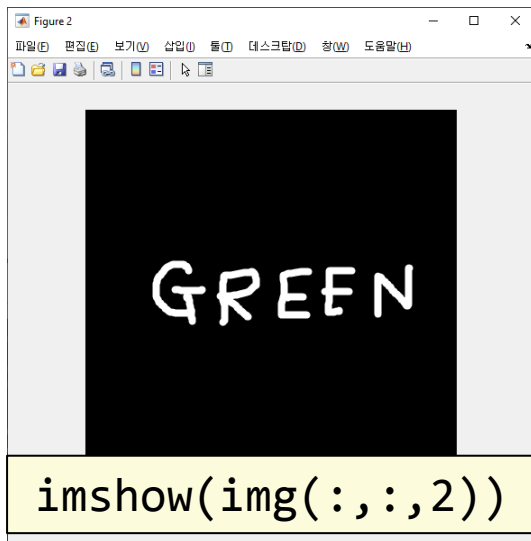
255, 0, 0

0, 255, 0

0, 0, 255

0, 0, 0

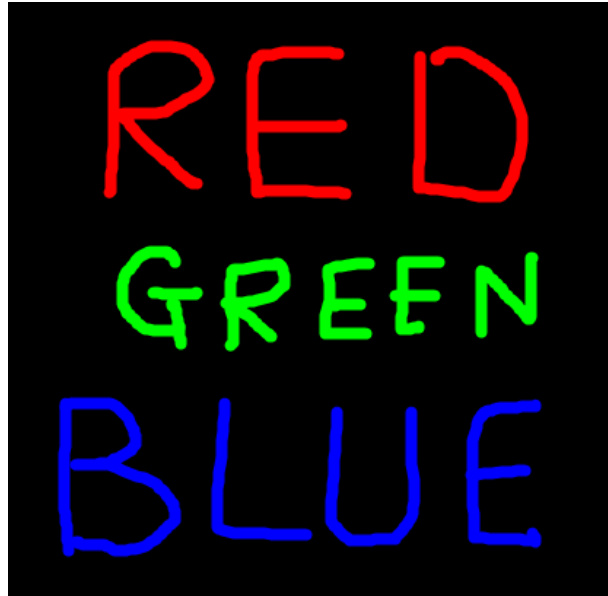
255, 255, 255



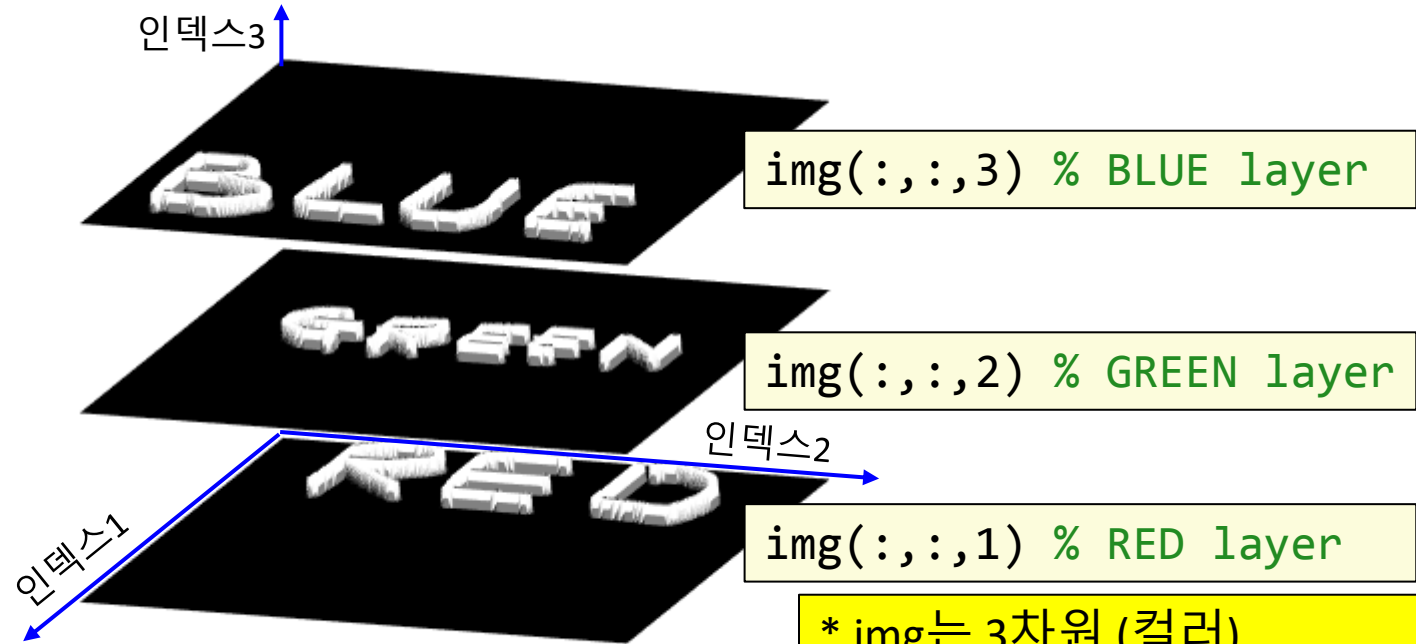
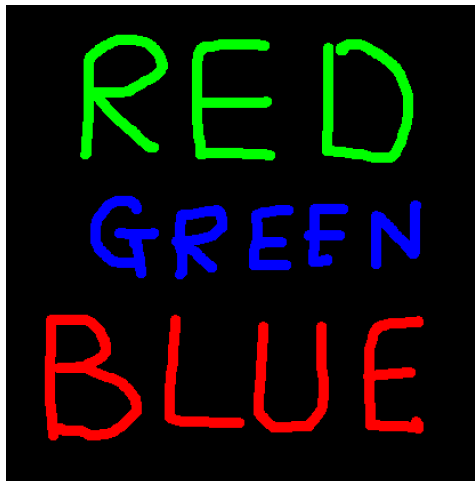
• 실습

- 컬러 이미지에 datatip을 찍어보자. (peppers.png도 해보자.)
- 각 색깔 이미지에 datatip을 찍어보자.
- 왜 색깔 이미지는 흑백일까?

컬러 이미지 크기가 $M \times N \times 3$ 인 이유



```
img = imread('RGB_test.png');
```



```
img(:,:,3) % BLUE layer
```

```
img(:,:,2) % GREEN layer
```

```
img(:,:,1) % RED layer
```

- * img는 3차원 (컬러)
- * 각 layer는 2차원 (흑백)
- * 한 픽셀에 값 3개가 필요 (RGB)

• 실습

- cameraman.tif와 peppers.png 파일의 파일정보를 보자. (bit depth 확인)
- 왼쪽같은 이미지는 어떻게 만들까?
- 그럼 imshow는 마법의 함수인가?
 - 2차원 행렬을 넣으면 흑백 이미지 출력?
 - 3차원 행렬을 넣으면 컬러 이미지 출력?

imshow 도움말을 보자.

▼ I — Input grayscale image matrix

Input grayscale image, specified as a matrix. A grayscale image can be any numeric data type.

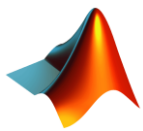
Data Types: single | double | int8 | int16 | int32 | int64 | uint8 | uint16 | uint32 | uint64 | logical

▼ RGB — Input truecolor image m -by- n -by-3 array

Input truecolor image, specified as an m -by- n -by-3 array.

If you specify a truecolor image of data type single or double, then values should be in the range [0, 1]. If pixel values are outside this range, then you can use the [rescale](#) function to scale pixel values to the range [0, 1]. The '[DisplayRange](#)' argument has no effect when the input image is truecolor.

Data Types: single | double | uint8 | uint16



`imshow(I, [low high])` displays the grayscale image `I`, specifying the display range as a two-element vector, `[low high]`. For more information, see the [DisplayRange](#) parameter.

low = 까만색, high = 흰색

`imshow(I, [])` displays the grayscale image `I`, scaling the display based on the range of pixel values in `I`. `imshow` uses `[min(I(:)) max(I(:))]` as the display range. `imshow` displays the minimum value in `I` as black and the maximum value as white. For more information, see the [DisplayRange](#) parameter.

✓ **'DisplayRange' — Grayscale image display range**
two-element vector | []

Display range of a grayscale image, specified as a two-element vector of the form `[low high]`. The `imshow` function displays the value `low` (and any value less than `low`) as black, and it displays the value `high` (and any value greater than `high`) as white. Values between `low` and `high` are displayed as intermediate shades of gray, using the default number of gray levels.

If you specify an empty matrix (`[]`), then `imshow` uses a display range of `[min(I(:)) max(I(:))]`. In other words, the minimum value in `I` is black, and the maximum value is white.

If you do not specify a display range, then `imshow` selects a default display range based on the image data type.

- If `I` is an integer type, then **'DisplayRange' defaults to the minimum and maximum representable values for that integer class.** For example, the default display range for `uint16` arrays is `[0, 65535]`.
- If `I` is data type `single` or `double`, then the default display range is `[0, 1]`.

i Note

Including the parameter name is optional, except when the image is specified by a file name. The syntax `imshow(I, [low high])` is equivalent to `imshow(I, 'DisplayRange', [low high])`. If you call `imshow` with a file name, then you must specify the **'DisplayRange'** parameter.

imshow의 display range 조절

low = 까만색, high = 흰색



```
img = imread('cameraman.tif');
```

```
figure,
```

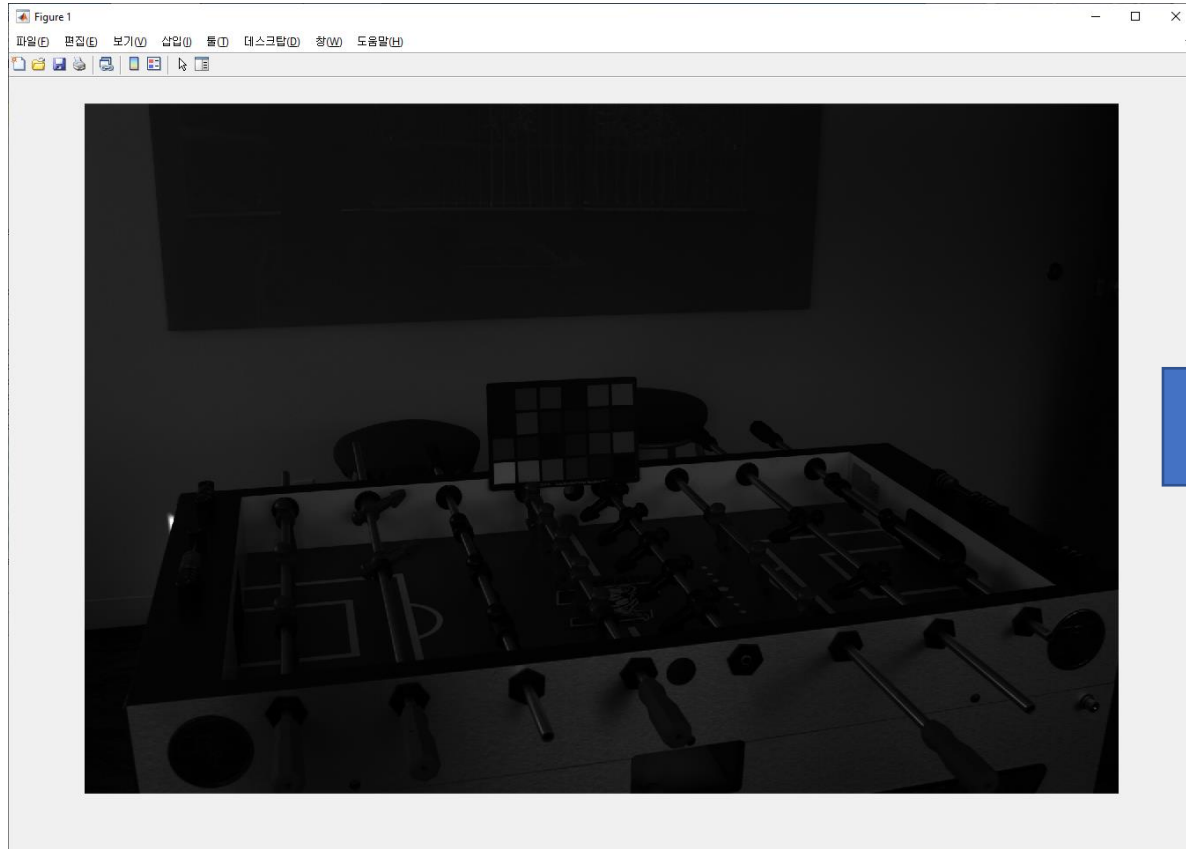
```
subplot(1,3,1), imshow(img) % img 자료형의 min, max 값으로 표시범위 지정 (uint8: [0, 255])
```

```
subplot(1,3,2), imshow(img,[]) % img의 min, max 값으로 표시범위 지정
```

```
subplot(1,3,3), imshow(img,[0, 512]) % 표시범위를 [0, 512]로 지정
```

uint8이 아닌 자료형을 쓸 일이...? 있다!

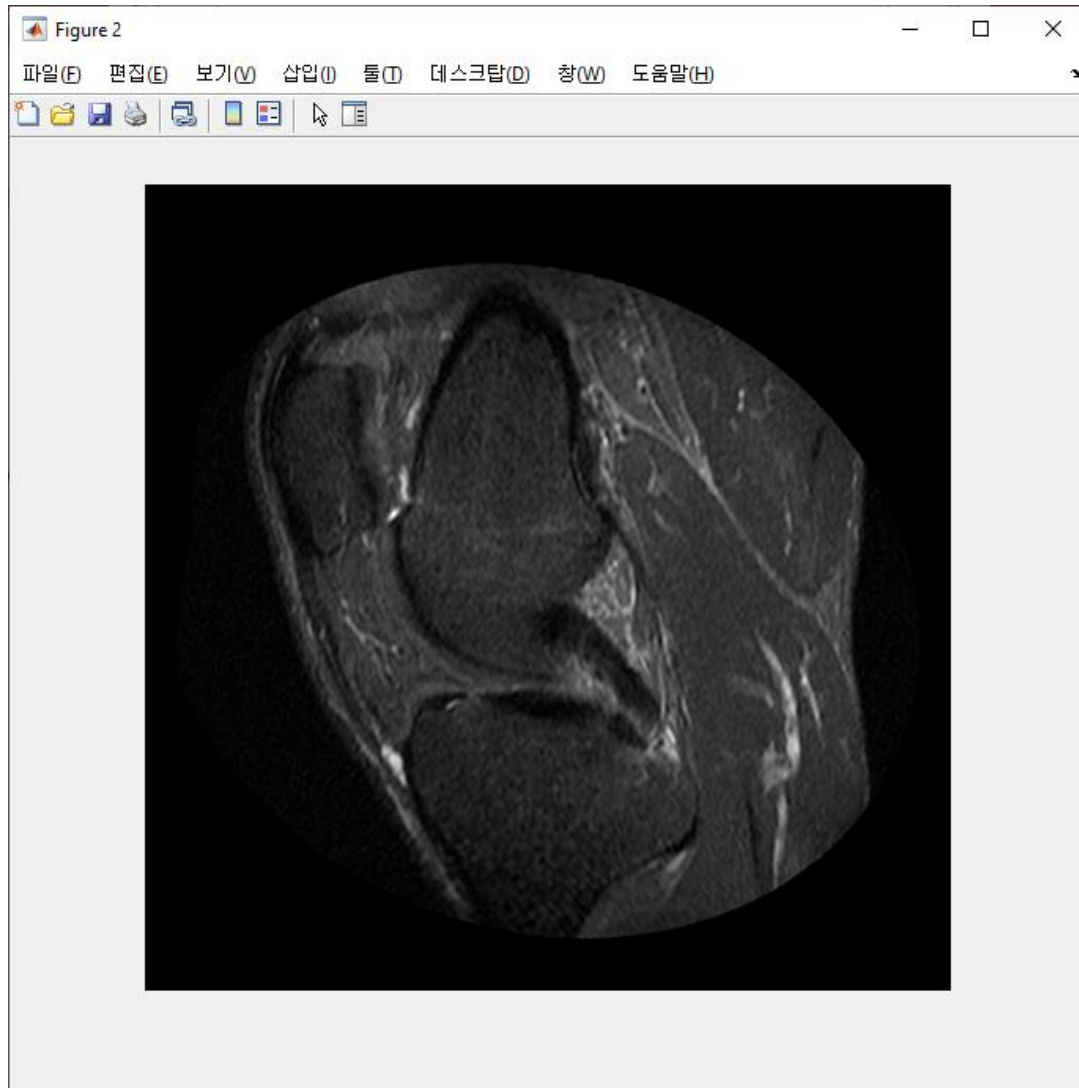
- 16 bit 이미지 (0-65535)



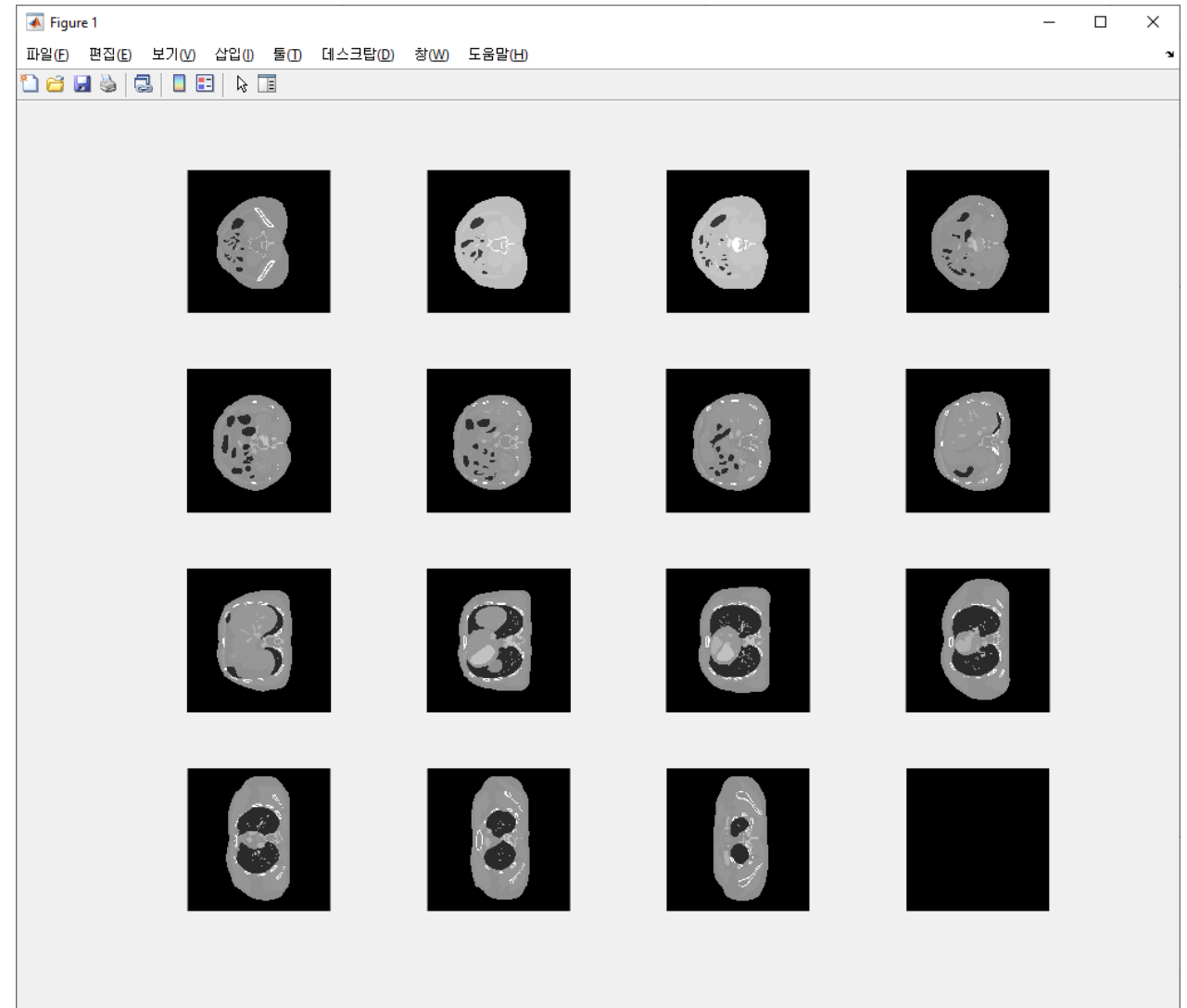
```
img = imread('foosballraw.tiff');  
figure, imshow(img)
```

- 실습: 대조도 조절
 - 이미지 전체
 - 거울 속 이미지
 - 컬러맵 수동 조절

uint8이 아닌 자료형을 쓸 일이...? 있다!



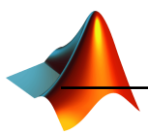
'knee2.dcm' (16 bit int)



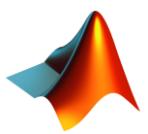
'img128.mat' (single)

이미지를 다루는 함수들

함수명	동작	예시	참고
imread	이미지 파일을 읽어옴	<code>img = imread('peppers.png');</code>	
imshow	이미지를 figure 창에 그림	<code>img = imread('peppers.png');</code> <code>imshow(img)</code>	imshow의 입력으로 파일명 가능
imwrite	이미지 파일 저장	<code>imwrite(img, 'myimage.png');</code>	자료형, 확장자에 따라 표시범위가 달라짐
imresize	이미지 크기 조절	<code>img = imresize(img, 0.5);</code> <code>img = imresize(img, [256 256]);</code>	
rgb2gray	컬러 이미지를 흑백 이미지로 변경	<code>img = rgb2gray(rgb);</code>	$\text{gray value} = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$
imfinfo	이미지 파일 정보 출력	<code>imfinfo('myimage.png')</code>	

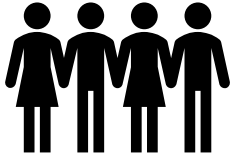


셀 배열 cell array



numeric 자료형의 한계

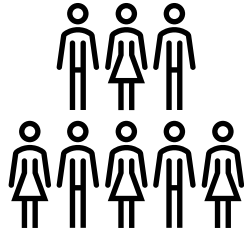
1반 (4명)



scores1 =

1반 과제점수	
강소라	76
나윤권	88
박소담	87
조정치	71

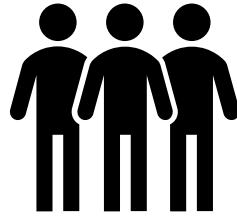
2반 (8명)



scores2 =

2반 과제점수	
고아라	90
라미란	87
마동석	65
서유리	78
신해철	99
아이유	80
유아인	86
장윤주	78

3반 (3명)



scores3 =

3반 과제점수	
김수현	71
장기하	82
황정민	94

scores = [scores1 scores2 scores3];

교과목 전체 점수

76	90	71
88	87	82
87	65	94
71	78	
	99	
	80	
	86	
	78	

배열의
차원이
일치하지
않습니다

scores1 = ...

scores2 = ...

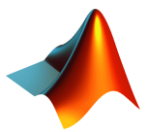
scores3 = ...

...

...

scores20 = ...

→ 이제 각 반의
평균점수를
계산해보자.



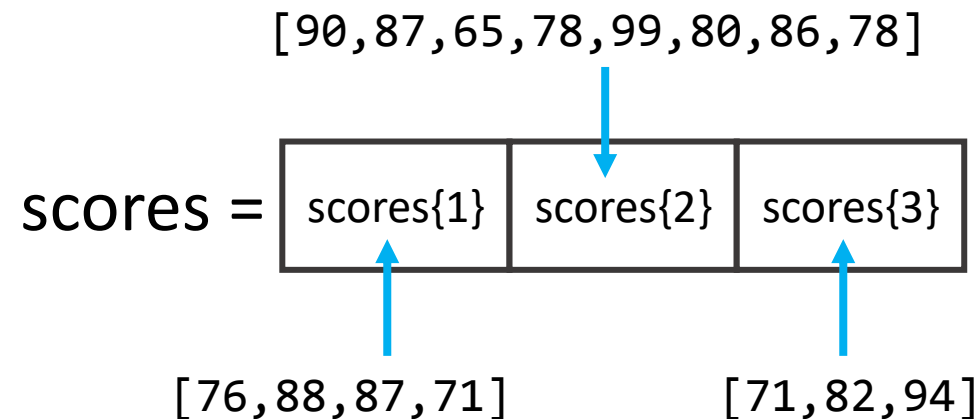
cell 자료형이 이 문제를 해결해준다.

() : parenthesis
{ } : (curly) brace
[] : bracket

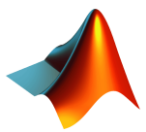
```
scores{1} = [76, 88, 87, 71];  
scores{2} = [90, 87, 65, 78, 99, 80, 86, 78];  
scores{3} = [71, 82, 94];
```

```
>> scores  
scores =  
    1x3 cell 배열  
    {1x4 double}    {1x8 double}    {1x3 double}  
>> disp(scores)  
    [1x4 double]    [1x8 double]    [1x3 double]  
>> celldisp(scores)  
scores{1} =  
    76    88    87    71  
scores{2} =  
    90    87    65    78    99    80    86    78  
scores{3} =  
    71    82    94
```

- 인덱스를 중괄호 {}로 감싼다.
- 각 요소의 길이가 달라도 된다.



- cell array의 내용을 출력하려면?
 - disp가 아닌 celldisp
 - 각 요소가 짧으면 disp도 가능



아니 그래서 도대체 cell array의 정체가!?

numeric array

- 각 요소가 숫자(numeric) 1개
- 즉, A(i, j)에는 숫자 하나(스칼라)만 들어감
- 숫자는 char, double, uint8 등
아무 자료형이나 OK
- 단 모든 요소의 자료형은 같아야 함

$$A = \begin{bmatrix} 14 & 0 & 29 & 29 & 32 \\ 59 & 48 & 11 & 0 & 60 \\ 36 & 0 & 33 & 88 & 70 \\ 3 & 0 & 21 & 0 & 0 \\ 54 & 87 & 0 & 62 & 92 \end{bmatrix}$$

cell array

- 각 요소가 cell
- 즉, C(i, j)에는 1x1의 cell이 들어감
- cell에는 아무 자료형이나 들어갈 수 있음
- cell에는 아무 크기나 들어갈 수 있음
- 즉, 변수로 정의할 수 있는 것은 무엇이든 OK

$$C = \left\{ \begin{array}{|c|c|c|} \hline 0 & \text{rand}(3) & \{2, 3, 4; 5, 6, 7\} \\ \hline \text{'test'} & @(x)\sin(x) & [] \\ \hline \text{struct} & \text{imread('moon.tif')} & \text{magic}(3) > 3 \\ \hline \end{array} \right\}$$

* char array: 각 요소가 character

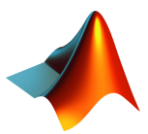
헛갈린다면? 엑셀을 떠올려라!

	A	B	C
1	0	0.123	43%
2	Hongik	HONGIK	0
3	₩3,000	3 1/7	오전 5:19:00

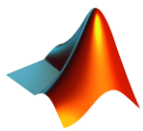
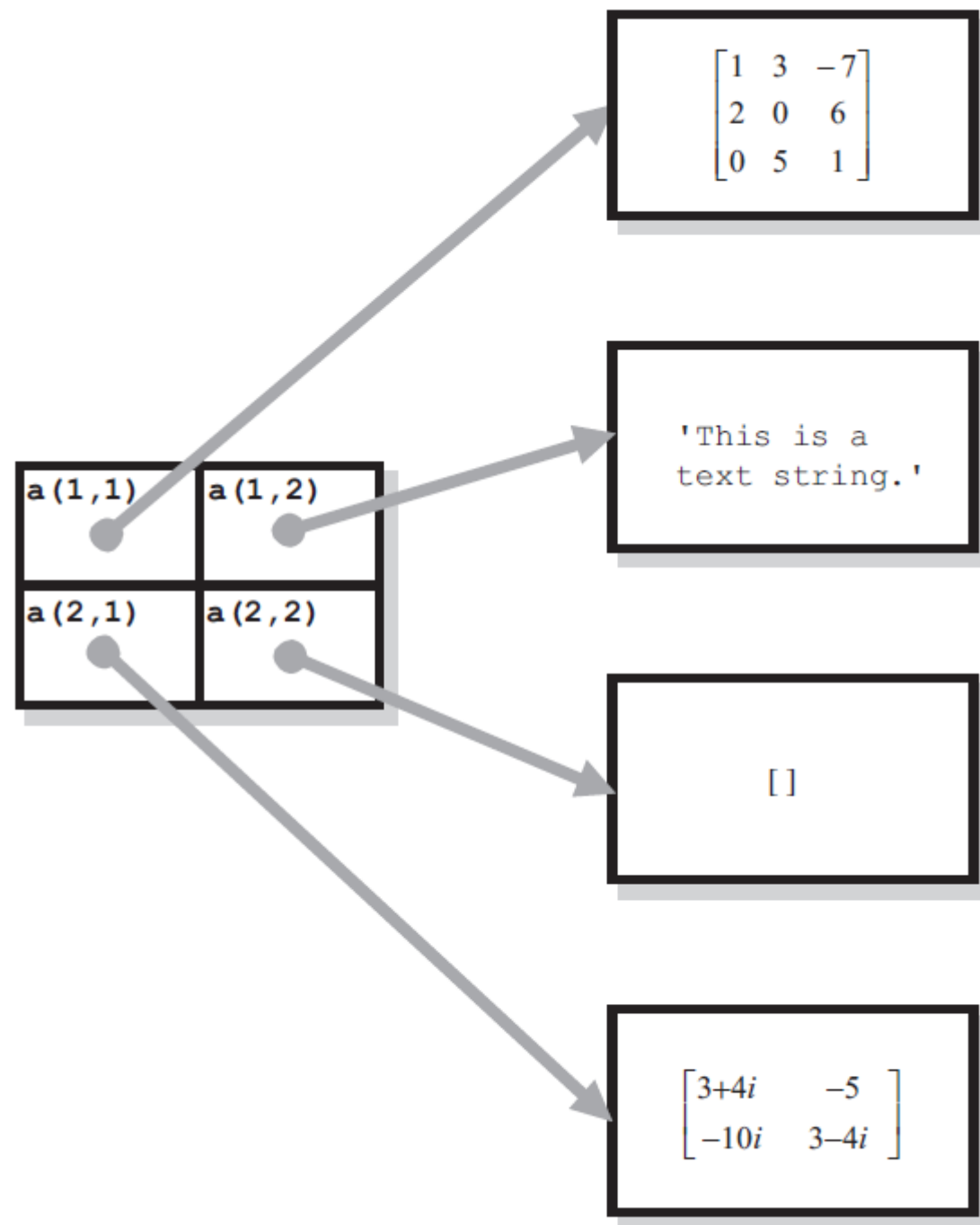
cell array

- 각 요소가 cell
- 즉, C(i, j)에는 1x1의 cell이 들어감
- cell에는 아무 자료형이나 들어갈 수 있음
- cell에는 아무 크기나 들어갈 수 있음
- 즉, 변수로 정의할 수 있는 것은 무엇이든 OK

$$C = \left\{ \begin{array}{|c|c|c|} \hline 0 & \text{rand}(3) & \{2, 3, 4; \\ & & 5, 6, 7\} \\ \hline \text{'test'} & @(x)\sin(x) & [] \\ \hline \text{struct} & \text{imread} \\ & \text{('moon.tif')} & \text{magic}(3) > 3 \\ \hline \end{array} \right\}$$



cell 1,1 $\begin{bmatrix} 1 & 3 & -7 \\ 2 & 0 & 6 \\ 0 & 5 & 1 \end{bmatrix}$	cell 1,2 'This is a text string.'
cell 2,1 $\begin{bmatrix} 3+i4 & -5 \\ -i10 & 3-i4 \end{bmatrix}$	cell 2,2 $[]$



cell array를 만드는 방법

% 참고

a(:) = 1; % numeric array 일괄변경
C(:) = {zeros(3)}; % cell array 일괄변경

방법 1. { } 이용

```
>> C1 = {0, [1,2]; rand(3), magic(5); 1>0, {1,2}}
C1 =
3x2 cell 배열
    {[      0]}    {1x2 double}
    {3x3 double}    {5x5 double}
    {[      1]}    {1x2 cell }
```

세미콜론
= 새 행

방법 2. cell 함수 이용

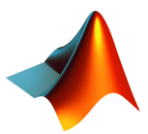
```
>> C2 = cell(4,2)
C2 =
4x2 cell 배열
    {0x0 double}    {0x0 double}
    {0x0 double}    {0x0 double}
    {0x0 double}    {0x0 double}
    {0x0 double}    {0x0 double}
```

빈 셀 배열
C = {};

방법 3. num2cell 이용 (numeric array -> cell array)

```
>> A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
>> C = num2cell(A)
C =
3x3 cell 배열
    {[8]}    {[1]}    {[6]}
    {[3]}    {[5]}    {[7]}
    {[4]}    {[9]}    {[2]}
```

```
>> S = ['hongik'; 'matlab'; '2020-2']
S =
3x6 char 배열
'hongik'
'matlab'
'2020-2'
>> C = num2cell(S)
C =
3x6 cell 배열
    {'h'}    {'o'}    {'n'}    {'g'}    {'i'}    {'k'}
    {'m'}    {'a'}    {'t'}    {'l'}    {'a'}    {'b'}
    {'2'}    {'0'}    {'2'}    {'0'}    {'-'}    {'2'}
```



cell array의 인덱싱 - 값을 읽을 때

괄호 ()를 이용한 경우

- cell array의 각 요소인 cell을 가져온다

```
>> scores(1)
ans =
    1x1 cell 배열
    {1x4 double}
>> scores(1:2)
ans =
    1x2 cell 배열
    {1x4 double}    {1x8 double}
```

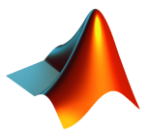
중괄호 { }를 이용한 경우

- cell 안의 "내용물"을 가져온다.

```
>> scores{1}
ans =
    76    88    87    71
>> scores{1:2}
ans =
    76    88    87    71
ans =
    90    87    65    78    99    80    86    78
```

- 헛갈린다면 엑셀을 다시 떠올리자!
 - () 사용 == 엑셀의 셀
 - { } 사용 == 엑셀 셀의 내용

	A	B	C
1	0	0.123	43%
2	Hongik	HONGIK	0
3	₩3,000	3 1/7	오전 5:19:00



cell array의 인덱싱 - 값을 쓸 때

```
C = {'one', 'two', 'three';  
    1, 2, 3};
```

```
upperLeft = C(1:2,1:2);
```

```
C(1,1:3) = {'first', 'second', 'third'};
```

```
row_1st = cell2mat(C(1,1:3));
```

```
row_2nd = cell2mat(C(2,1:3));
```

```
last = C{2,3};
```

```
A(:,1:2) = 100; % numeric  
C(:,1:2) = {zeros(3)}; % cell
```

```
C{2,3} = 300;
```

실습: workspace에서 확인

```
C =  
    2x3 cell 배열  
    {'one'}    {'two'}    {'three'}  
    {[ 1]}    {[ 2]}    {[ 3]}
```

```
upperLeft =  
    2x2 cell 배열  
    {'one'}    {'two'}  
    {[ 1]}    {[ 2]}
```

```
C =  
    2x3 cell 배열  
    {'first'}    {'second'}    {'third'}  
    {[ 1]}    {[ 2]}    {[ 3]}
```

```
row_1st =  
    'firstsecondthird'
```

```
row_2nd =  
     1     2     3
```

```
last =  
     3
```

```
C =  
    2x3 cell 배열  
    {'first'}    {'second'}    {'third'}  
    {[ 1]}    {[ 2]}    {[ 300]}
```

cell array의 인덱싱 - 2중 인덱싱

```
scores{1} = [76, 88, 87, 71];  
scores{2} = [90, 87, 65, 78, 99, 80, 86, 78];  
scores{3} = [71, 82, 94];  
disp(scores{1}(2:3)) % [88, 87]  
disp(mean(scores{2}(3:5))) % mean([65, 78, 99]) = 80.6667
```

```
C{1} = [1, 10, 100, 1000];  
C{2} = magic(3);  
C{3} = 'hongik matlab';  
disp(C{1}(3)) % 100  
disp(C{2}(2:3,2:3)) % [5, 7; 9, 2]  
disp(C{3}(6:8)) % 'k m'
```

```
c1 = {1,2,3,4};  
c2 = {5,6,7,8};  
c = {c1, c2};  
disp(c{1,2})  
disp(c{1}{2})
```

※ 주의: $c\{1,2\}$ 와 $c\{1\}\{2\}$ 는 다르다.

```
>> magic(3)
```

```
ans =
```

8	1	6
3	5	7
4	9	2

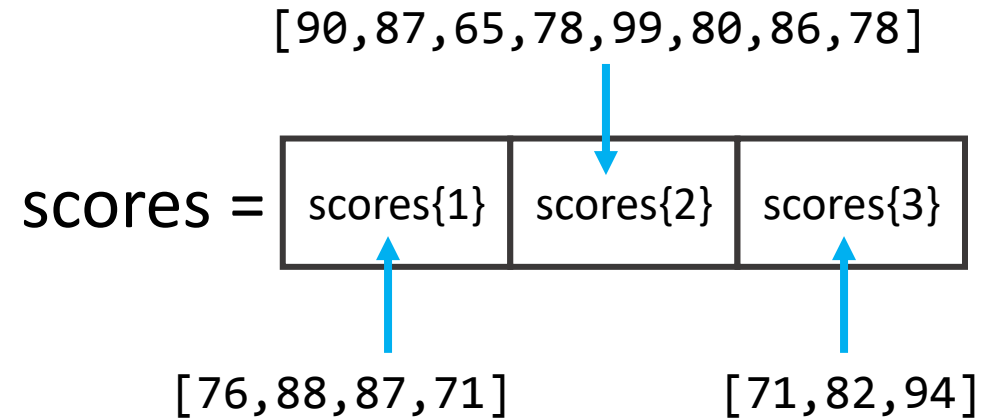
```
>> magic(3)(2:3,2:3)
```

```
Error: ()-indexing must appear last in an index  
expression.
```

이제 각 반의 평균을 내보자.

```
scores{1} = [76, 88, 87, 71];  
scores{2} = [90, 87, 65, 78, 99, 80, 86, 78];  
scores{3} = [71, 82, 94];  
  
for i=1:length(scores)  
    fprintf('class %d:\n', i)  
    fprintf('\t%d students, ', length(scores{i}))  
    fprintf('average: %4.1f\n', mean(scores{i}))  
end
```

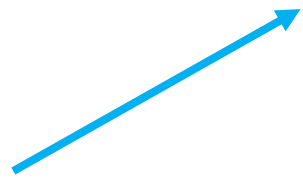
```
class 1:  
    4 students, average: 80.5  
class 2:  
    8 students, average: 82.9  
class 3:  
    3 students, average: 82.3
```



이왕 하는 거 반 번호, 이름까지 다 넣어보자.

classes =

classes{1}	classes{2}	classes{3}
------------	------------	------------



classes{1}

'class1'	'Kang'	76
'class1'	'Na'	88
'class1'	'Park'	87
'class1'	'Jo'	71

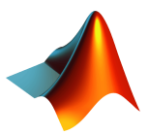
classes{2}

'class2'	'Go'	90
'class2'	'Ra'	87
'class2'	'Ma'	65
'class2'	'Seo'	78
'class2'	'Shin'	99
'class2'	'I'	80
'class2'	'You'	86
'class2'	'Jang'	78

classes{3}

'class3'	'Kim'	71
'class3'	'Jang'	82
'class3'	'Hwang'	94

```
class1
    'class1'      'Kang'      [76]
    'class1'      'Na'        [88]
    'class1'      'Park'     [87]
    'class1'      'Jo'        [71]
class2
    'class2'      'Go'        [90]
    'class2'      'Ra'        [87]
    'class2'      'Ma'        [65]
    'class2'      'Seo'       [78]
    'class2'      'Shin'     [99]
    'class2'      'I'         [80]
    'class2'      'You'       [86]
    'class2'      'Jang'     [78]
class3
    'class3'      'Kim'       [71]
    'class3'      'Jang'     [82]
    'class3'      'Hwang'    [94]
```



그럼 cell array가 만능일까?

- cell array의 치명적 단점: 연산을 할 수 없다.

```
c1 = {1,2,3};  
c2 = {4,5,6};  
  
disp(c1+c2) % error!  
  
disp(c1.*c2) % error!  
  
disp(cell2mat(c1)+cell2mat(c2))  
  
disp(cell2mat(c1).*cell2mat(c2))
```

연산자 '+'은(는) 'cell'형 입력 인수에 대해 정의되지 않았습니다.

연산자 '.*'은(는) 'cell'형 입력 인수에 대해 정의되지 않았습니다.

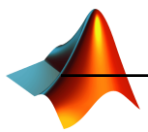
5 7 9

4 10 18

- 모든 연산자는 cell array에 대해서는 정의되어 있지 않음
- mean, std, factorial, sin, exp 등 많은 함수들도 cell array를 입력으로 받지 못함
- 연산을 위해서는 numeric array로 바꿔주어야 함
- cell array는 필요할 때만 쓰자.

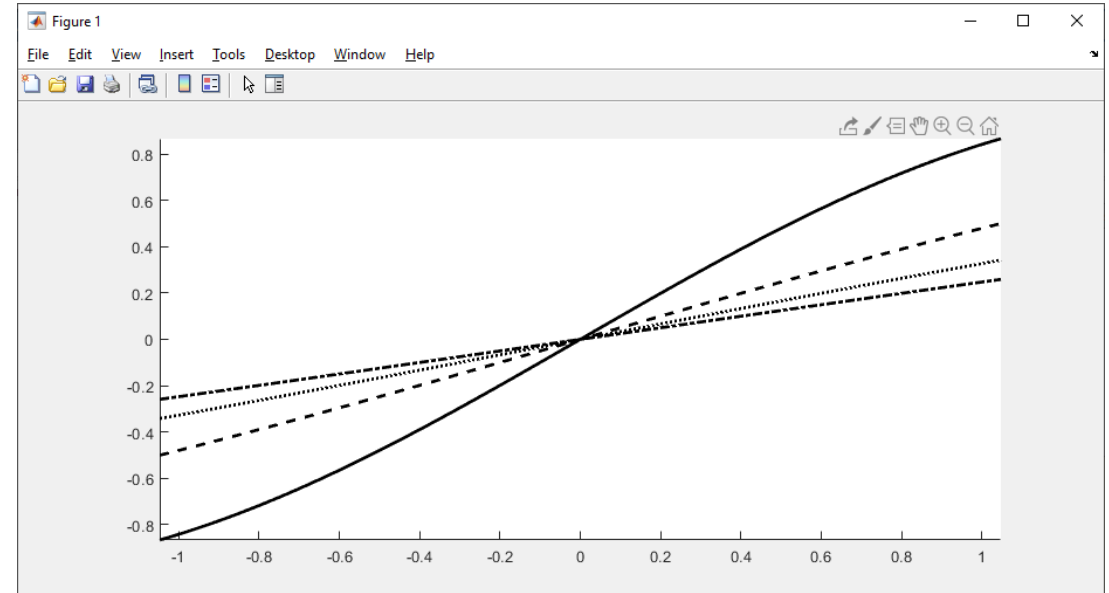
cell array에 쓸 수 있는 함수들

함수명	동작	예시	참고
length size numel	numeric array의 동작과 같음	length({1,2,3,4,5}) size({1,2,3; 4,5,6}) numel({1,2,3; 4,5,6})	
num2cell	numeric array의 각 원소를 cell의 내용으로 하는 cell array 생성	num2cell([1,2,3;4,5,6]) num2cell(['matlab'; 'hongik'])	
cell2mat	cell의 모든 내용을 분해하여 numeric array를 생성	C = {[1], [2 3 4]; [5; 9], [6 7 8; 10 11 12]} cell2mat(C)	cell의 각 원소는 행렬로 이어붙일 수 있는 size를 가져야 하며, 모든 내용은 자료형이 같아야 함
iscell	cell array인지 아닌지 반환	iscell(C)	
celldisp	cell의 모든 내용을 출력	celldisp(C)	
cellplot	cell의 형태를 figure로 보여줌	cellplot(C)	

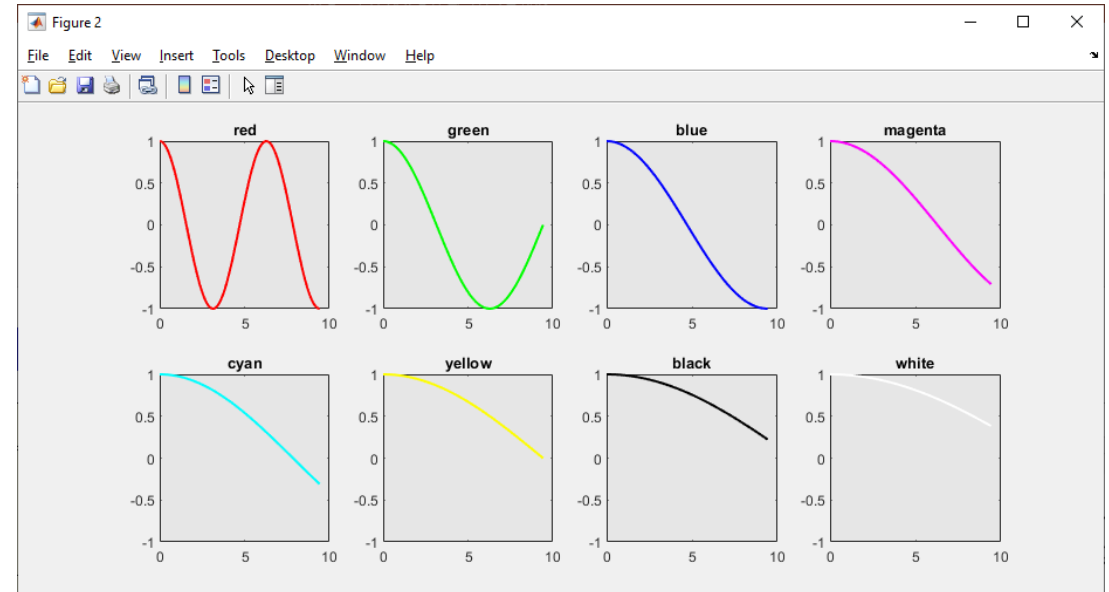


cell array의 활용 - 길이가 다른 문자열

```
figure, hold on,  
styles = {'-', '--', ':', '-.'};  
  
for m = 1:4  
    x = linspace(-pi/3, pi/3, 200);  
    y = sin(x/m);  
    plot(x, y, styles{m}, 'color', 'k', 'linewidth', 2);  
end  
xlim([-1 1])  
axis tight
```



```
figure,  
colors = 'rbgmcyk';  
titles = {'red', 'green', 'blue', 'magenta', ...  
         'cyan', 'yellow', 'black', 'white'};  
x = linspace(0, 3*pi);  
for n = 1:length(colors)  
    subplot(2,4,n)  
    y = cos(x/n);  
    plot(x, y, colors(n), 'linewidth', 1.5);  
    title(titles{n})  
    set(gca, 'color', [.9 .9 .9])  
    ylim([-1 1])  
end
```



cell array 사용 시 주의할 점

% Cell array is still a matrix.

% Every column must have same number of rows.

```
C1 = {rand(2), magic(3);  
      'string', [], 1>0}; % error
```

※ cell array도 행렬이므로 직사각형 형태여야 함

% LHS uses () -> RHS must be cell array with corresponding size

```
C2 = cell(2,3);
```

```
C2(1,1:3) = {'first', 'second', 'third'};
```

※ 좌변에 () 사용: 우변은 cell이어야 하고, 크기가 맞아야 함

```
% C2(1,1:3) = ['first', 'second', 'third']; -> error
```

```
% C2{1,1:3} = {'first', 'second', 'third'}; -> error
```

% LHS uses {} -> RHS can be anything.

```
C2{2,1} = 'string';
```

```
C2{2,2} = magic(5);
```

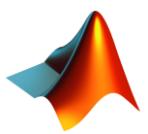
```
C2{2,3} = {'another', 'cell'};
```

※ 좌변에 {} 사용: 우변은 무엇이든 올 수 있음

```
cellplot(C2)
```

※ cell array도 pre-allocation을 이용한 속도 향상 가능

구조체 / 구조체 배열
struct / struct array



cell 자료형의 아쉬운 점

class1 =

'class1'	'Kang'	76
'class1'	'Na'	88
'class1'	'Park'	87
'class1'	'Jo'	71

class

name

score

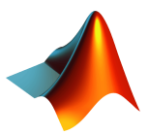
- class1을 사용하려면 다음을 미리 알고 있어야 한다.
 - 첫 번째 column이 class
 - 두 번째 column이 name
 - 세 번째 column이 score

```
class1 = {  
    'class1', 'Kang', 76  
    'class1', 'Na', 88  
    'class1', 'Park', 87  
    'class1', 'Jo', 71  
};
```



```
fprintf('avg. of class1: %4.1f\n', mean(cell2mat(class1(:,3))))
```

- 데이터 접근 시 "이름"으로 접근할 수는 없을까?



지수, 제니, 로제, 리사

Jisoo



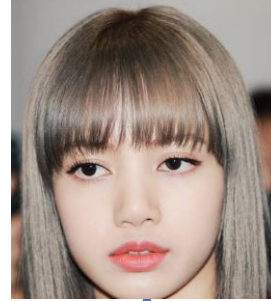
Jennie



Rose



Lisa



```
name = 'Jisoo'
birthyear = 1995
height = 162
bloodtype = 'A'
```

```
name = 'Jennie'
birthyear = 1996
height = 163
bloodtype = 'B'
```

```
name = 'Rose'
birthyear = 1997
height = 168
bloodtype = 'B'
```

```
name = 'Lisa'
birthyear = 1997
height = 167
bloodtype = 'O'
```

```
Jisoo.name = 'Jisoo';
Jisoo.birthyear = 1995;
Jisoo.height = 162;
Jisoo.bloodtype = 'A';
```

```
Jennie.name = 'Jennie';
Jennie.birthyear = 1996;
Jennie.height = 163;
Jennie.bloodtype = 'B';
```

```
Rose.name = 'Rose';
Rose.birthyear = 1997;
Rose.height = 168;
Rose.bloodtype = 'B';
```

```
Lisa.name = 'Lisa';
Lisa.birthyear = 1997;
Lisa.height = 167;
Lisa.bloodtype = 'O';
```

- Jisoo의 키를 출력하고 싶다면?

```
fprintf('Height of Jisoo: %d cm\n', Jisoo.height)
```

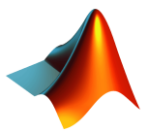
- Rose의 혈액형은?

```
fprintf('Bloodtype of Rose: %s\n', Rose.bloodtype)
```

기본형태: structName.fieldName

Jisoo, Jennie, Rose, Lisa
-> struct (구조체)

birthyear, height, bloodtype
-> field



구조체를 만드는 방법

```
% 1) structName.fieldName
Jisoo.name = 'Jisoo';
Jisoo.birthyear = 1995;
Jisoo.height = 162;
Jisoo.bloodtype = 'A';

% 2) use 'struct' function
Jennie = struct('name', 'Jennie', ...
               'birthyear', 1996, ...
               'height', 163, ...
               'bloodtype', 'B');

% 3) use 'struct' and cell array
c = {'name', 'Rose', ...
     'birthyear', 1997, ...
     'height', 168, ...
     'bloodtype', 'B'};
Rose = struct(c{:});

% 4) empty struct (no fields)
Lisa = struct;
```

Jisoo



name = 'Jisoo'
birthyear = 1995
height = 162
bloodtype = 'A'

Jennie



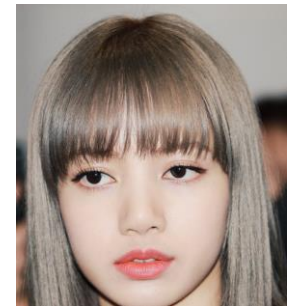
name = 'Jennie'
birthyear = 1996
height = 163
bloodtype = 'B'

Rose



name = 'Rose'
birthyear = 1997
height = 168
bloodtype = 'B'

Lisa



name = 'Lisa'
birthyear = 1997
height = 167
bloodtype = 'O'

구조체 "배열"

```
blackpink = [Jisoo, Jennie, Rose, Lisa];
```

blackpink

```
>> blackpink
blackpink =
    1x4 struct array with fields:
     name
    birthyear
     height
    bloodtype
```

※ workspace 확인

```
>> blackpink(1).name
ans =
    'Jisoo'
>> blackpink(2).birthyear
ans =
    1996
>> blackpink(3).bloodtype
ans =
    'B'
>> blackpink(4).height
ans =
    167
```



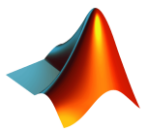
name
= 'Jisoo'
birthyear
= 1995
height
= 162
bloodtype
= 'A'

name
= 'Jennie'
birthyear
= 1996
height
= 163
bloodtype
= 'B'

name
= 'Rose'
birthyear
= 1997
height
= 168
bloodtype
= 'B'

name
= 'Lisa'
birthyear
= 1997
height
= 167
bloodtype
= 'O'

numeric array
-> 각 요소가 숫자 (number)
cell array
-> 각 요소가 cell
struct array
-> 각 요소가 struct



구조체 배열의 활용

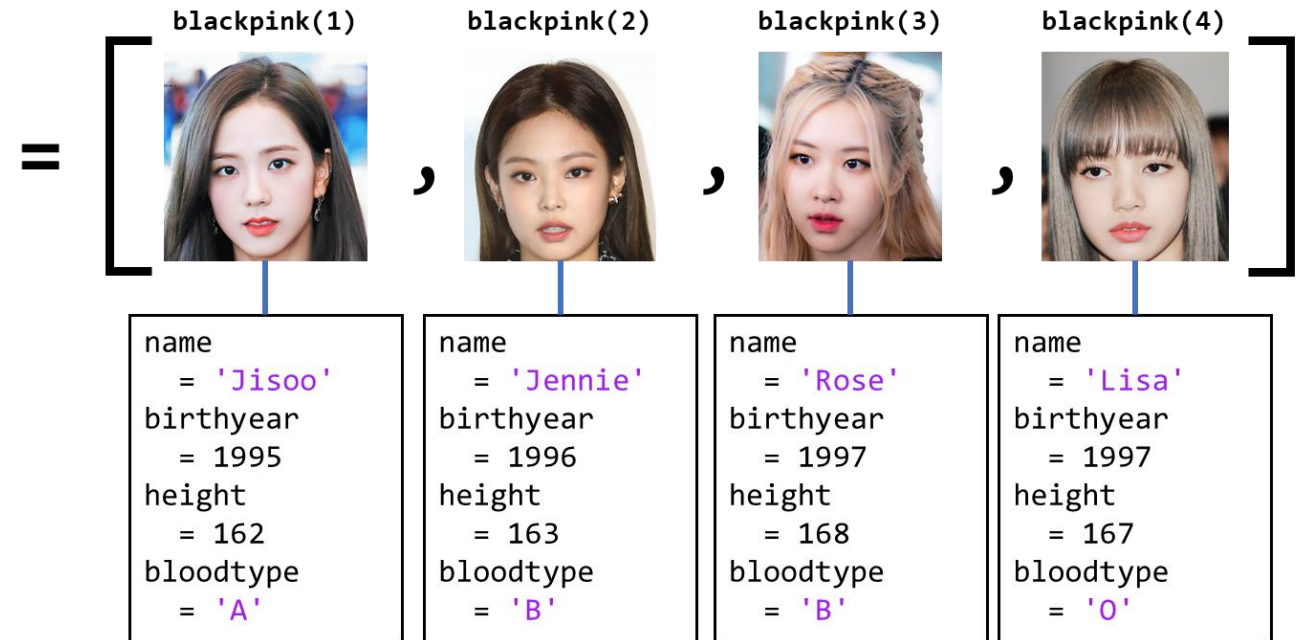
```
blackpink = [Jisoo, Jennie, Rose, Lisa];

% average height
height = 0;
for i=1:length(blackpink)
    height = height + blackpink(i).height;
end
height = height/length(blackpink);
fprintf('Average height is %5.1f cm.\n', height)

% who is oldest?
bys = zeros(1,length(blackpink));
for i=1:length(blackpink)
    bys(i) = blackpink(i).birthyear;
end
[~,idx] = min(bys);
fprintf('Oldest member is %s (born in %d).\n', ...
        blackpink(idx).name, blackpink(idx).birthyear)

% how many bloodtypes of B?
N = 0;
for i=1:length(blackpink)
    if strcmpi(blackpink(i).bloodtype, 'B')
        N = N+1;
    end
end
fprintf('%d members are bloodtypes of B.\n', N)
```

blackpink



Average height is 165.0 cm.
Oldest member is Jisoo (born in 1995).
2 members are bloodtypes of B.

구조체 배열의 활용

```
hydrogen = struct('name', 'hydrogen', ...
    'atomic_number', 1, ...
    'symbol', 'H', ...
    'density', 0.08988, ...
    'boiling_point', 20.271);

helium = struct('name', 'helium', ...
    'atomic_number', 2, ...
    'symbol', 'He', ...
    'density', 0.1786, ...
    'boiling_point', 4.222);

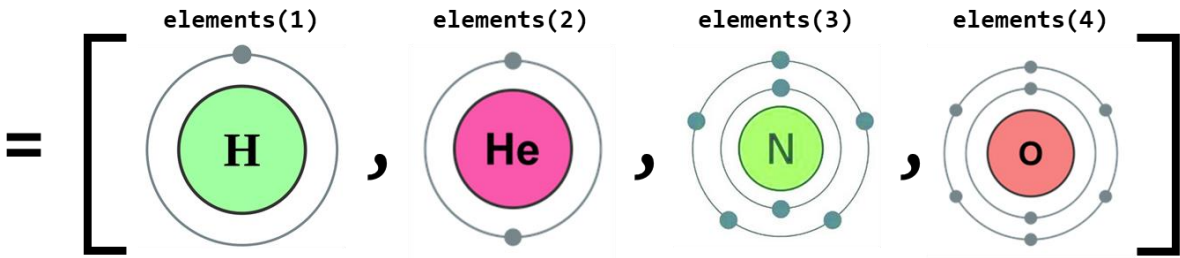
nitrogen = struct('name', 'nitrogen', ...
    'atomic_number', 7, ...
    'symbol', 'N', ...
    'density', 1.2506, ...
    'boiling_point', 77.355);

oxygen = struct('name', 'oxygen', ...
    'atomic_number', 8, ...
    'symbol', 'O', ...
    'density', 1.429, ...
    'boiling_point', 90.188);

elements = [hydrogen, helium, nitrogen, oxygen];
```

원자명	원자번호	기호	밀도 (g/L)	끓는점 (K)
hydrogen	1	H	0.08988	20.271
helium	2	He	0.1786	4.222
nitrogen	7	N	1.2506	77.355
oxygen	8	O	1.429	90.188

elements



name	'hydrogen'	'helium'	'nitrogen'	'oxygen'
atomic_number	1	2	7	8
symbol	'H'	'He'	'N'	'O'
density	0.08988	0.1786	1.2506	1.429
boiling_point	20.271	4.222	77.355	90.188





구조체 배열의 활용

```
files = dir('./test_files/*.*');
Nfiles = length(files);

% size of files
filesizes = zeros(1,Nfiles);
for i=1:Nfiles
    filesizes(i) = files(i).bytes;
end

% extension of files
exts = cell(1,Nfiles);
for i=1:Nfiles
    [~,~,ext] = fileparts(files(i).name);
    exts{i} = ext;
end
```

examples > test_files

Name	Date modified	Type	Size
 bp.xlsx	2020-10-26 오후 5:09	Microsoft Excel 워...	10 KB
 exp1.txt	2020-10-26 오후 9:12	Text Document	1 KB
 foosballraw.tiff	2016-11-23 오전 2:13	TIFF File	11,987 KB
 knee2.dcm	2011-11-09 오전 3:01	DCM File	514 KB

```
>> files
files =
    6x1 struct array with fields:
        name
        folder
        date
        bytes
        isdir
        datenum
>> filesizes
filesizes =
         0
         0
        9832
        240
    12274656
    526330

>> exts
exts =
    6x1 cell array
    {'.'}
    {'.'}
    {'.xlsx'}
    {'.txt'}
    {'.tiff'}
    {'.dcm'}
```


구조체 배열의 활용

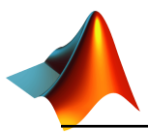
```
a = [1,2,3,4,5];  
b = 'foo';  
c = {magic(3), rand(5), zeros(4)};  
  
save('data.mat');  
  
a = 100;  
b = 'bar';  
c = cell(1,1);  
  
data = load('data.mat');  
% load('test.mat'); -> overwrite the existing variables
```

Workspace		
Name ▲	Value	Size
a	100	1x1
b	'bar'	1x3
c	1x1 cell	1x1
data	1x1 struct	1x1

```
>> data  
data =  
    struct with fields:  
  
    a: [1 2 3 4 5]  
    b: 'foo'  
    c: {[3x3 double] [5x5 double] [4x4 double]}
```

구조체를 다루는 함수들

함수명	동작	예시	참고
fieldnames	구조체의 field 목록을 cell array로 반환	<pre>>> fieldnames(blackpink) ans = 4x1 cell array {'name' } {'birthyear' } {'height' } {'bloodtype' }</pre>	
isfield	구조체에 해당 이름의 field가 있는지를 반환 (있으면 1, 없으면 0)	<pre>>> isfield(blackpink, 'birthday') ans = <u>logical</u> 0</pre>	
rmfield	필드를 없앴	<pre>>> rmfield(blackpink, 'bloodtype') ans = 1x4 struct array with fields: name birthyear height</pre>	필드명은 문자열 원래의 구조체는 바뀌지 않으며, 새 구조체가 반환됨
isstruct	구조체이면 1, 아니면 0	<pre>>> isstruct(blackpink) ans = <u>logical</u> 1</pre>	



구조체 관련 팁

- struct array는 모두 같은 field를 가져야 한다.
 - Jisoo는 name과 height를, Jennie는 height와 birthyear를 갖는 구조체 -> X
- fieldname은 동적으로 접근할 수 있다.

```
Jisoo = struct('name', 'Jisoo', ...  
              'birthyear', 1995, ...  
              'height', 162, ...  
              'bloodtype', 'A');
```

```
n = 'name';  
b = 'birthyear';
```

```
fprintf('%s, %s\n', n, Jisoo.(n))  
fprintf('%s, %d\n', b, Jisoo.(b))
```

Command Window

```
name, Jisoo  
birthyear, 1995
```

```
files = dir('./test_files/*.*');  
sortwith = 'bytes';  
  
[~,idx] = sort([files.(sortwith)]);  
files = files(idx);
```

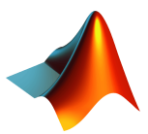
```
>> [files.bytes]  
ans =  
  
      0  
      0  
    10468  
     9832  
      240  
    189924
```



```
>> [files.bytes]  
ans =  
  
      0  
      0  
     240  
     9832  
    10468  
    189924
```

table 자료형

엑셀 읽고 쓰기



간단한 실습 - 엑셀 데이터로 table 만들기

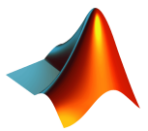
bp.xlsx

	A	B	C	D
1	name	birthyear	height	bloodtype
2	Jisoo	1995	162	A
3	Jennie	1996	163	B
4	Rose	1997	168	B
5	Lisa	1997	167	O

```
t = readtable('bp.xlsx');
```

t =
4x4 table
name birthyear height bloodtype
<hr/>
'Jisoo' 1995 162 'A'
'Jennie' 1996 163 'B'
'Rose' 1997 168 'B'
'Lisa' 1997 167 'O'

- readtable('파일명');
 - 각 column에 이름이 붙는다.
('variable name'이라고 부른다.)
 - 엑셀의 1행이 variable name이 된다.
 - 한 column 내의 데이터는 자료형이 같아야 한다.
 - 문자 -> cell (:: 길이가 다를 수 있으므로)
 - 숫자 -> double (:: 매트랩 기본 자료형)
 - column에 숫자와 문자가 섞여있으면 모두 문자로 인식하고 cell이 된다.
 - 각 column의 row 개수는 같다.



간단한 실습 - 변수로 table 만들기

```
names = {'Jisoo', 'Jennie', 'Rose', 'Lisa'}';  
birthyear = [1995, 1996, 1997, 1997]';  
height = [162 163 168 167]';  
bloodtype = {'A', 'B', 'B', 'O'}';  
  
t = table(names, birthyear, height, bloodtype);  
disp(t)
```

```
t =  
4x4 table  
    name    birthyear    height    bloodtype  
_____  
'Jisoo'    1995         162         'A'  
'Jennie'   1996         163         'B'  
'Rose'     1997         168         'B'  
'Lisa'     1997         167         'O'
```

- table(변수1, 변수2, ...);
 - 변수명이 table의 variable name이 된다.
 - 각 column의 row 개수는 같다.
 - 왼쪽 예제에서 일부만 transpose(')를 하면 에러가 발생한다.
 - 왼쪽 예제에서 모두 transpose를 하지 않으면 에러는 발생하지 않으나 형태가 달라진다.
- 변수명을 쓰지 않고 값을 직접 입력할 경우, variable name은 Var1, Var2, ... 등으로 자동지정된다.

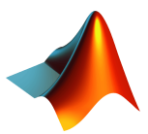


table 자료형의 인덱싱

```
t = readtable('bp.xlsx');  
disp(t)
```

```
names = t.name;  
bys = t.birthyear;  
heights = t.height;  
bts = t.bloodtype;
```

```
t(1, :)  
   name    birthyear    height    bloodtype  
   ----    -  
   'Jisoo'    1995      162      'A'  
  
t(2:3, 3:end)  
   height    bloodtype  
   ----    -  
   163      'B'  
   168      'B'  
  
t(:, 4)  
   bloodtype  
   ----  
   'A'  
   'B'  
   'B'  
   'O'
```

```
>> names  
names =  
   4x1 cell array  
   {'Jisoo' }  
   {'Jennie'}  
   {'Rose' }  
   {'Lisa' }  
  
>> bys  
bys =  
      1995  
      1996  
      1997  
      1997  
  
>> heights  
heights =  
      162  
      163  
      168  
      167  
  
>> bts  
bts =  
   4x1 cell array  
   {'A'}  
   {'B'}  
   {'B'}  
   {'O'}
```

- named indexing
 - tableName.variableName
 - 구조체 field 값 보는 것과 동일
 - 출력은 cell 또는 numeric
- numeric indexing
 - numeric array의 인덱싱과 동일
 - 출력은 table

table 자료형의 인덱싱

```
% named indexing to modify the table
t.name = {'지수', '제니', '로제', '리사'}';

% add variable at the end
t.birthday = {'Jan 01', 'Jan 16', 'Feb 11', 'Mar 27'}';

% add variable after 'name'
engName = {'Jisoo', 'Jennie', 'Rose', 'Lisa'}';
t = addvars(t, engName, 'After', 'name');
% same as addvars(t, engName, 'Before', 'birthyear');

% re-order variables
t = t(:, [1 2 3 6 4 5]);

% delete a variable from a table
t.birthday = {'95 Jan 01';
              '96 Jan 16';
              '97 Feb 11';
              '97 Mar 27'}';

t.birthyear = [];
t(:, end) = [];
```

name	birthyear	height	bloodtype
'지수'	1995	162	'A'
'제니'	1996	163	'B'
'로제'	1997	168	'B'
'리사'	1997	167	'O'

name	birthyear	height	bloodtype	birthday
'지수'	1995	162	'A'	'Jan 01'
'제니'	1996	163	'B'	'Jan 16'
'로제'	1997	168	'B'	'Feb 11'
'리사'	1997	167	'O'	'Mar 27'

name	engName	birthyear	height	bloodtype	birthday
'지수'	'Jisoo'	1995	162	'A'	'Jan 01'
'제니'	'Jennie'	1996	163	'B'	'Jan 16'
'로제'	'Rose'	1997	168	'B'	'Feb 11'
'리사'	'Lisa'	1997	167	'O'	'Mar 27'

name	engName	birthyear	birthday	height	bloodtype
'지수'	'Jisoo'	1995	'Jan 01'	162	'A'
'제니'	'Jennie'	1996	'Jan 16'	163	'B'
'로제'	'Rose'	1997	'Feb 11'	168	'B'
'리사'	'Lisa'	1997	'Mar 27'	167	'O'

name	engName	birthday	height
'지수'	'Jisoo'	'95 Jan 01'	162
'제니'	'Jennie'	'96 Jan 16'	163
'로제'	'Rose'	'97 Feb 11'	168
'리사'	'Lisa'	'97 Mar 27'	167

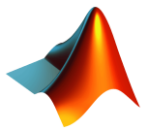


table 자료형의 인덱싱

```
% add row (=new member)
t(end+1,:) = {'미미', 'MiMi', '15 Jun 01', 35};

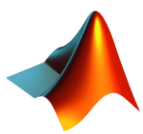
% delete row
t(end,:) = [];
% same as t = t(1:end-1,:);

% row re-order
t = t([3,4,2,1],:);
```

name	engName	birthday	height
'지수'	'Jisoo'	'95 Jan 01'	162
'제니'	'Jennie'	'96 Jan 16'	163
'로제'	'Rose'	'97 Feb 11'	168
'리사'	'Lisa'	'97 Mar 27'	167
'미미'	'MiMi'	'15 Jun 01'	35

name	engName	birthday	height
'지수'	'Jisoo'	'95 Jan 01'	162
'제니'	'Jennie'	'96 Jan 16'	163
'로제'	'Rose'	'97 Feb 11'	168
'리사'	'Lisa'	'97 Mar 27'	167

name	engName	birthday	height
'로제'	'Rose'	'97 Feb 11'	168
'리사'	'Lisa'	'97 Mar 27'	167
'제니'	'Jennie'	'96 Jan 16'	163
'지수'	'Jisoo'	'95 Jan 01'	162



readtable로 엑셀 파일 읽어오기

```
% read row names
t = readtable('bp.xlsx', 'ReadRowNames', true);

% read specific range
t = readtable('bp.xlsx', 'Range', 'A1:C4');

% read specific sheet
t = readtable('bp.xlsx', 'Sheet', 'itzy');
```

인덱싱은 row name이 없는 경우와 동일

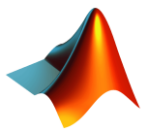
	A	B	C	D
1	name	birthyear	height	bloodtype
2	Jisoo	1995	162	A
3	Jennie	1996	163	B
4	Rose	1997	168	B
5	Lisa	1997	167	O
6				
7				
8				
9				

	A	B	C	D
1	name	birthyear	height	bloodtype
2	Yeji	2000	167	A
3	Ryujin	2001	164	B
4	Chaeryeong	2001	167	B
5	Lia	2000	162	AB
6	Yuna	2003	169	A
7				
8				
9				

	birthyear	height	bloodtype
Jisoo	1995	162	'A'
Jennie	1996	163	'B'
Rose	1997	168	'B'
Lisa	1997	167	'O'

name	birthyear	height
'Jisoo'	1995	162
'Jennie'	1996	163
'Rose'	1997	168

name	birthyear	height	bloodtype
'Yeji'	2000	167	'A'
'Ryujin'	2001	164	'B'
'Chaeryeong'	2001	167	'B'
'Lia'	2000	162	'AB'
'Yuna'	2003	169	'A'



UI로 엑셀 파일 읽어오기

- 현재폴더에서 더블클릭
- 우클릭-열기
- 명령창에서 uiopen
 - 확장자 지정: uiopen('*.xlsx')
- UI 기반으로 파일 읽을 때
문자열은 string 자료형
(≠ char array)

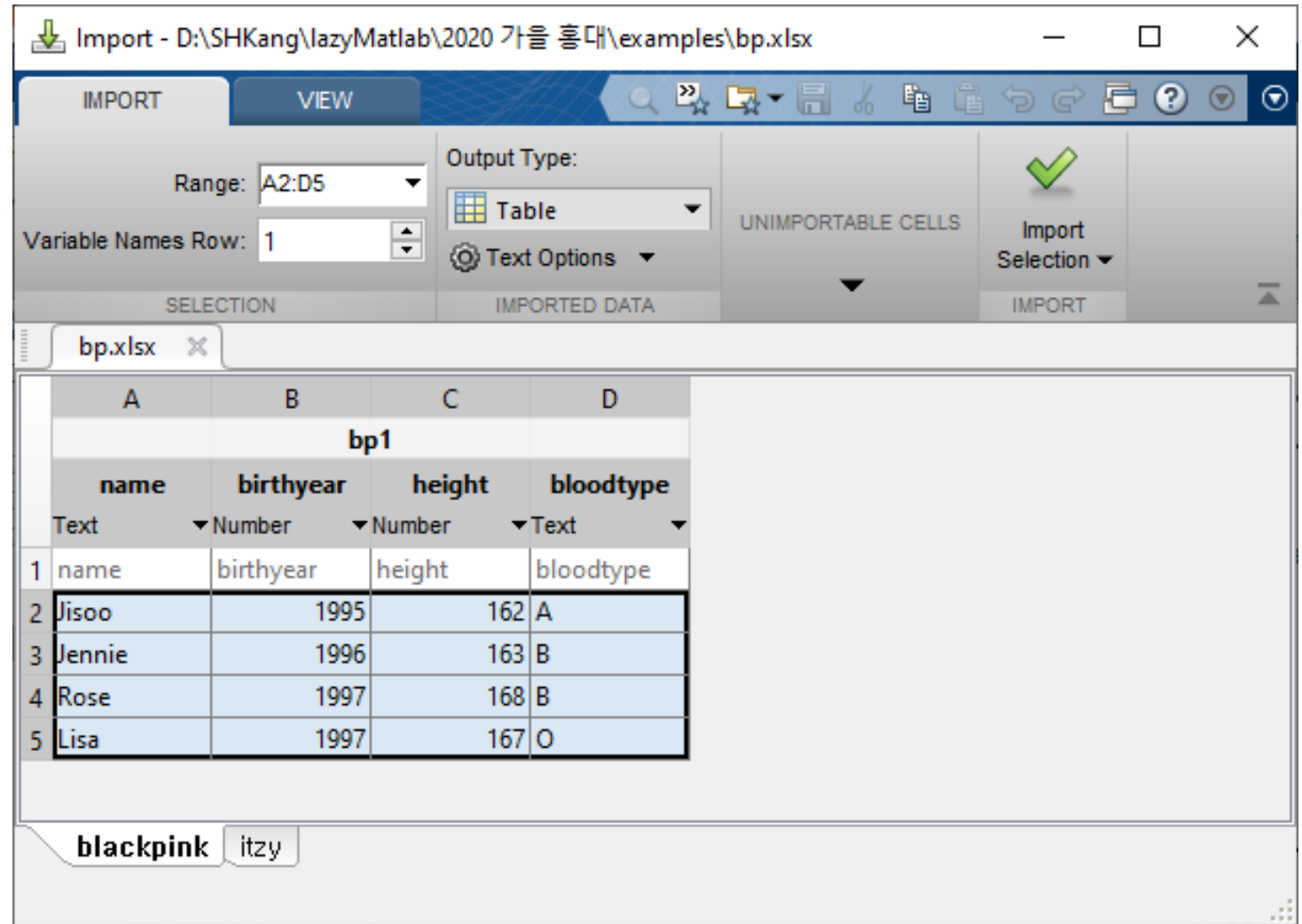


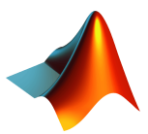
table ↔ struct array

```
t = readtable('bp.xlsx');  
disp(t)  
  
s = table2struct(t);  
  
t2 = struct2table(s);
```

t					
4x4 table					
	1	2	3	4	5
	name	birthyear	height	bloodtype	
1	'Jisoo'	1995	162	'A'	
2	'Jennie'	1996	163	'B'	
3	'Rose'	1997	168	'B'	
4	'Lisa'	1997	167	'O'	

s					
4x1 struct with 4 fields					
Fields	name	birthyear	height	bloodtype	
1	'Jisoo'	1995	162	'A'	
2	'Jennie'	1996	163	'B'	
3	'Rose'	1997	168	'B'	
4	'Lisa'	1997	167	'O'	

t2					
4x4 table					
	1	2	3	4	5
	name	birthyear	height	bloodtype	
1	'Jisoo'	1995	162	'A'	
2	'Jennie'	1996	163	'B'	
3	'Rose'	1997	168	'B'	
4	'Lisa'	1997	167	'O'	

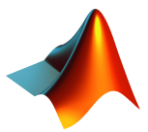


writetable로 엑셀 파일 쓰기

```
names = {'YURA', 'SOJIN', 'MINAH', 'HYERI'}';  
birthyear = [1992, 1986, 1993, 1994]';  
height = [170 166 164 167]';  
bloodtype = {'O', 'AB', 'O', 'AB'}';  
  
t = table(names, birthyear, height, bloodtype);  
disp(t)  
  
writetable(t, 'gd.xlsx');
```

names	birthyear	height	bloodtype
'YURA'	1992	170	'O'
'SOJIN'	1986	166	'AB'
'MINAH'	1993	164	'O'
'HYERI'	1994	167	'AB'

	A	B	C	D
1	names	birthyear	height	bloodtype
2	YURA	1992	170	O
3	SOJIN	1986	166	AB
4	MINAH	1993	164	O
5	HYERI	1994	167	AB



elements - table

```
names = {'hydrogen', 'helium', 'nitrogen', 'oxygen'}';  
atomic_numbers = [1, 2, 7, 8]';  
symbols = {'H', 'He', 'N', 'O'}';  
densities = [0.08988, 0.1786, 1.2506, 1.429]';  
boiling_points = [20.271, 4.222, 77.355, 90.188]';  
  
elements = table(names, ...  
                 atomic_numbers, ...  
                 symbols, ...  
                 densities, ...  
                 boiling_points);  
  
disp(elements)
```

원자명	원자번호	기호	밀도 (g/L)	끓는점 (K)
hydrogen	1	H	0.08988	20.271
helium	2	He	0.1786	4.222
nitrogen	7	N	1.2506	77.355
oxygen	8	O	1.429	90.188

names	atomic_numbers	symbols	densities	boiling_points
'hydrogen'	1	'H'	0.08988	20.271
'helium'	2	'He'	0.1786	4.222
'nitrogen'	7	'N'	1.2506	77.355
'oxygen'	8	'O'	1.429	90.188

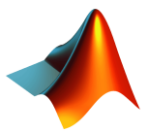


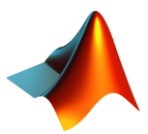
table 자료형 - 주의할 점

- 각 column은 같은 row 개수를 가짐
 - 엑셀에서 없는 cell을 불러오면 빈 문자열 또는 NaN
- table의 variable name은 문자여야 함
- 한 column 내의 데이터는 자료형이 같아야 한다.
 - 숫자가 있는 위치에 cell -> 에러
 - 숫자가 있는 위치에 char, int8 등
-> double로 자동 변환

	A	B	C	D
1	1	2	3	4
2	Jisoo	1995	162	A
3	Jennie	1996	163	B
4		1997	168	B
5	Lisa		167	O
6				

x1	x2	x3	x4
'Jisoo'	1995	162	'A'
'Jennie'	1996	163	'B'
''	1997	168	'B'
'Lisa'	NaN	167	'O'

string 자료형



char array vs string array

```
ch = 'Hongik University';  
  
str = ["Hongik", "University"];  
  
str2 = ["Hongik"; "University"];  
% ['Hongik'; 'University'] throws an error.
```

세미콜론
= 새 행

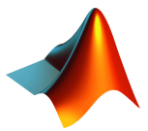
- char array

- 각 요소가 1x1 character인 행렬
- character = 문자 하나, "로 묶음
- 'matlab'의 크기는 1x6

- string array

- 각 요소가 1x1 string인 행렬
- string = ""로 묶인 문자열
- "matlab"의 크기는 1x1

```
>> ch  
  
ch =  
  
    'Hongik University'  
  
>> str  
  
str =  
  
    1x2 string array  
  
    "Hongik"    "University"  
  
>> str2  
  
str2 =  
  
    2x1 string array  
  
    "Hongik"  
    "University"
```



string의 활용

```
s = "I am " + "Groot.";
c = ['I am ', 'Groot.'];

s = "I am Quill. " + 38 + " years old.";
c = ["I am Quill. ", 38, " years old."];
c2 = ['I am Quill. ', num2str(38), ' years old.'];

s = "Where " + 'is ' + 'Gamora' + "?";
c = ["Where ", 'is ', 'Gamora', "?"];

s = "The answer will be ""I am Groot.""";
c = 'The answer will be "I am Groot."';

s = "I am " + {'Quill', 'Groot', 'Gamora'};

s = "Hongik" + "Matlab";
c = 'Hongik' + 'Matlab';
% c = 'Hongik' + 'Univ' + 'Matlab'; -> error
```

```
s =
    "I am Groot."
c =
    'I am Groot.'
s =
    "I am Quill. 38 years old."
c =
    1x3 string array
    "I am Quill. "    "38"    " years old."
c2 =
    'I am Quill. 38 years old.'
s =
    "Where is Gamora?"
c =
    1x4 string array
    "Where "    "is "    "Gamora"    "?"
s =
    "The answer will be "I am Groot.""
c =
    'The answer will be "I am Groot."'
s =
    1x3 string array
    "I am Quill"    "I am Groot"    "I am Gamora"
s =
    "HongikMatlab"
c =
    149    208    226    211    202    205
```

string의 활용

```
words = ["Fabulous", "Amazing", "Tremendous";  
         "Extraordinary", "Stunning", "Incredible";  
         "Astonishing", "Unbelievable", "Awesome"];
```

```
size(words)
```

```
strlength(words)
```

```
count(words, "ing")
```

```
count(words, "e")
```

```
erase(words, "e")
```

```
insertBefore(words, "e", ".")
```

```
insertBefore(words, 3, "-")
```

```
ans =  
      3      3
```

```
ans =  
      8      7     10  
     13      8     10  
     11     12      7
```

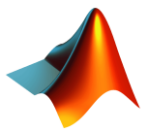
```
ans =  
      0      1      0  
      0      1      0  
      1      0      0
```

```
ans =  
      0      0      2  
      0      0      2  
      0      3      2
```

```
ans =  
3x3 string array  
    "Fabulous"      "Amazing"      "Trmndous"  
    "Extraordinary" "Stunning"      "Incrdibl"  
    "Astonishing"   "Unblivabl"     "Awsom"
```

```
ans =  
3x3 string array  
    "Fabulous"      "Amazing"      "Tr.em.endous"  
    "Extraordinary" "Stunning"      "Incr.edibl.e"  
    "Astonishing"   "Unb.eli.evabl.e" "Aw.esom.e"
```

```
ans =  
3x3 string array  
    "Fa-bulous"      "Am-azing"      "Tr-emendous"  
    "Ex-traordinary" "St-unning"      "In-credible"  
    "As-tonishing"   "Un-believable" "Aw-esome"
```



string의 활용

```
words = ["Fabulous", "Amazing", "Tremendous";  
         "Extraordinary", "Stunning", "Incredible";  
         "Astonishing", "Unbelievable", "Awesome"];
```

```
insertAfter(words, "e", ".")
```

```
insertAfter(words, 3, "-")
```

```
join(words)
```

```
join(words, ', ')
```

```
replace(words, "e", "E")
```

% also a function for char array

```
split("I am Groot")
```

```
split("I am Groot", " ")
```

```
split("I-am-Groot", "-")
```

% also a function for char array (-> cell array)

```
ans =  
3x3 string array  
"Fabulous"      "Amazing"      "Tre.me.ndous"  
"Extraordinary" "Stunning"     "Incre.dible."  
"Astonishing"   "Unbe.lie.vable." "Awe.some."
```

```
ans =  
3x3 string array  
"Fab-ulous"      "Ama-zing"      "Tre-mendous"  
"Ext-raordinary" "Stu-nning"     "Inc-redible"  
"Ast-onishing"   "Unb-elievable" "Awe-some"
```

```
ans =  
3x1 string array  
"Fabulous Amazing Tremendous"  
"Extraordinary Stunning Incredible"  
"Astonishing Unbelievable Awesome"
```

```
ans =  
3x1 string array  
"Fabulous, Amazing, Tremendous"  
"Extraordinary, Stunning, Incredible"  
"Astonishing, Unbelievable, Awesome"
```

```
ans =  
3x3 string array  
"Fabulous"      "Amazing"      "TrEmEndous"  
"Extraordinary" "Stunning"     "IncrEdible"  
"Astonishing"   "UnbElieEvablE" "AwEsomE"
```

```
ans =  
3x1 string array  
"I"  
"am"  
"Groot"
```

string의 활용 - string 메서드

```
>>
>>
>>
>> words
words =
  3x3 str array
    "Fabu"    "azing"    "Tremendous"
    "Extr"    "unning"    "Incredible"
    "Asto"    "believable" "Awesome"
fx >> words.
```

```
words = ["Fabulous", "Amazing", "Tremendous";
         "Extraordinary", "Stunning", "Incredible";
         "Astonishing", "Unbelievable", "Awesome"];
words.append("!")
words.contains("ous")
words.count("i")
words.endsWith("g")
words.extractBefore("ing")
words.sort()
words.strlength()
words.upper()
```

```
ans =
  3x3 string array
    "Fabulous!"    "Amazing!"    "Tremendous!"
    "Extraordinary!" "Stunning!"    "Incredible!"
    "Astonishing!" "Unbelievable!" "Awesome!"

ans =
  3x3 logical array
    1    0    1
    0    0    0
    0    0    0

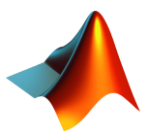
ans =
  3x3 logical array
    0    1    0
    0    1    0
    1    0    0

ans =
  3x3 string array
    <missing>    "Amaz"    <missing>
    <missing>    "Stunn"    <missing>
    "Astonish"    <missing>    <missing>

ans =
  3x3 string array
    "Astonishing"    "Amazing"    "Awesome"
    "Extraordinary"    "Stunning"    "Incredible"
    "Fabulous"    "Unbelievable"    "Tremendous"
```

정리

- 텍스트 파일 쓰고 읽기
 - fprintf: 포맷에 맞춰 파일에 텍스트 쓰기
 - fscanf: 포맷에 맞춰 파일에서 텍스트 읽어오기
 - fopen: 파일을 읽거나 쓰기 위한 경로 생성
 - 'r': 읽기 권한, 'w': 쓰기 권한 (기존 내용 삭제), 'a': 쓰기 권한 (기존 내용 뒤에)
 - fclose: 파일과 연결된 경로 삭제
- 이미지도 행렬이다.
 - 정수자료형 (예. uint8 -> 0-255 범위의 정수)
 - 흑백이미지 (M x N uint8): 0=black, 255=white
 - 컬러이미지 (M x N x 3 uint8): RGB의 조합으로 색을 표현
 - imshow(img,[imin, imax])로 표시범위 조절 가능 (imin 이하=black, imax이상=white)
 - uint16, double로도 이미지를 표현할 수 있다.




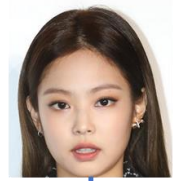
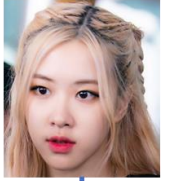
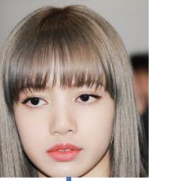
정리

C =	0	rand(3)	{2, 3, 4; 5, 6, 7}
	'test'	@(x)sin(x)	[]
	struct	imread ('moon.tif')	magic(3) > 3

cell array

- 각 요소가 1x1 cell
- cell에는 아무 자료형이나 아무 크기로 들어갈 수 있음
- 인덱싱: ()는 cell 형태의 subarray를, {}는 셀의 내용을 가져옴
- 연산을 할 수 없고 mean, std 등의 함수도 사용할 수 없음

blackpink

blackpink(1)	blackpink(2)	blackpink(3)	blackpink(4)
			
name = 'Jisoo' birthyear = 1995 height = 162 bloodtype = 'A'	name = 'Jennie' birthyear = 1996 height = 163 bloodtype = 'B'	name = 'Rose' birthyear = 1997 height = 168 bloodtype = 'B'	name = 'Lisa' birthyear = 1997 height = 167 bloodtype = 'O'

struct array

- 각 요소가 1x1 struct
- struct에 속한 값은 숫자 인덱스가 아닌 field 이름으로 접근
- field에 접근방법: structName.fieldname
- 배열의 모든 요소는 같은 field를 가져야 함

정리

```
t =  
4x4 table  
      name      birthyear      height      bloodtype  
-----  
'Jisoo'      1995      162      'A'  
'Jennie'     1996      163      'B'  
'Rose'       1997      168      'B'  
'Lisa'       1997      167      'O'
```

table

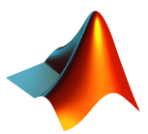
- 각 column에 이름이 붙음
- named indexing (tableName.variableName)
-> column을 가져옴
- numeric indexing -> numeric array와 동일
- readtable, writetable로 엑셀파일 읽고 쓰기 가능

```
str = ["Hongik", "University"];  
str2 = ["Hongik"; "University"];
```

```
s = "I am " + "Groot.";  
s = "I am Quill. " + 38 + " years old.";
```

string

- char와 다르게 "Hongik"은 1x1 string임
- 1x1 string이므로 길이가 다른 문자열을 한 행렬에 넣을 수 있음
- + 연산자로 string을 더할 수 있음
- + 연산자 사용 시 숫자도 string으로 자동변환됨
- 함수를 .functionName 형태로 쓸 수 있음



Q&A

