MATLAB 빠른 참조

배열, 행렬, 플로팅, 파일 I/O, 제어 흐름

기본

커맨드 창

```
x = 5;           % assign (semicolon suppresses output)
x = 5            % assign and display result
disp('Hello')    % print to console
clc              % clear command window
clear            % clear all variables
```

도움말 및 정보

```
help sin         % quick help for function
doc sin          % open documentation
who              % list variables in workspace
whos             % list with details (size, type)
```

연산자

+ - * / ^	산술 (행렬 연산)
.* ./ .^	요소별 연산
== ~= < > <= >=	비교 연산자
&& || ~	논리 AND, OR, NOT (스칼라)
& | ~	요소별 논리 (배열)

변수 및 타입

숫자 타입

```
x = 3.14;            % double (default)
n = int32(42);       % 32-bit integer
z = 2 + 3i;          % complex number
tf = true;           % logical
```

타입 확인

class(x)	타입 이름을 문자열로 반환
isa(x, 'double')	x가 특정 타입인지 확인
isnumeric(x)	숫자 타입이면 true
ischar(x)	문자 배열이면 true
islogical(x)	논리 타입이면 true

특수 상수

pi	3.14159...
Inf / -Inf	무한대
NaN	숫자가 아님
eps	머신 엡실론 (~2.2e-16)
i / j	허수 단위

배열 및 행렬

배열 생성

```
v = [1 2 3 4 5];         % row vector
v = [1; 2; 3];           % column vector
A = [1 2; 3 4];          % 2x2 matrix
r = 1:5;                 % [1 2 3 4 5]
r = 0:0.5:2;             % [0 0.5 1 1.5 2]
```

내장 생성자

```
zeros(3)        % 3x3 of zeros
ones(2, 4)      % 2x4 of ones
eye(3)          % 3x3 identity
rand(2, 3)      % 2x3 uniform random
linspace(0,1,5) % 5 evenly spaced [0..1]
```

인덱싱 및 슬라이싱

```
A(2, 3)          % row 2, col 3
A(1, :)          % entire first row
A(:, 2)          % entire second column
A(1:2, 1:2)      % submatrix
A(end, :)        % last row
```

행렬 연산

A'	전치 (켤레)
A.'	전치 (켤레 없음)
inv(A)	역행렬
det(A)	행렬식
eig(A)	고유값 및 고유벡터
A \ b	Ax = b 풀기
size(A)	차원 [행 열]
numel(A)	전체 요소 수

제어 흐름

if / elseif / else

```
if x > 0
    disp('positive')
elseif x == 0
    disp('zero')
else
    disp('negative')
end
```

for 및 while

```
for i = 1:10
    fprintf('i = %d\n', i);
end
while x > 0
    x = x - 1;
end
```

switch

```
switch grade
    case 'A'
        disp('Excellent')
    case {'B', 'C'}
        disp('Good')
    otherwise
        disp('Try harder')
end
```

반복문 제어

break	가장 안쪽 반복문 종료
continue	다음 반복으로 건너뛰기
return	함수 즉시 종료

함수

함수 파일

```
% Save as myfunc.m
function result = myfunc(x, y)
    result = x.^2 + y.^2;
end
```

다중 출력

```
function [mn, mx] = minmax(v)
    mn = min(v);
    mx = max(v);
end
[lo, hi] = minmax([3 1 4 1 5]);
```

익명 함수

```
f = @(x) x.^2 + 1;
f(3)                    % returns 10
g = @(x,y) x + y;
arrayfun(f, [1 2 3])    % apply to each element
```

유용한 내장 함수

sum(v)	요소의 합
mean(v)	평균값
max(v) / min(v)	최대 / 최소
sort(v)	오름차순 정렬
find(v > 3)	조건이 참인 인덱스
length(v)	벡터 길이

플로팅

2D 플롯

```
x = 0:0.1:2*pi;
plot(x, sin(x), 'r-', 'LineWidth', 2)
xlabel('x'); ylabel('sin(x)')
title('Sine Wave'); grid on
legend('sin(x)')
```

여러 플롯

```
hold on
plot(x, sin(x), 'b-')
plot(x, cos(x), 'r--')
hold off
subplot(1,2,1); plot(x, sin(x))
subplot(1,2,2); plot(x, cos(x))
```

기타 플롯 유형

bar(x, y)	막대 차트
histogram(data)	히스토그램
scatter(x, y)	산점도
pie(data)	파이 차트
surf(X, Y, Z)	3D 표면 플롯
imagesc(A)	행렬을 이미지로 표시

그림 저장

```
saveas(gcf, 'plot.png')
exportgraphics(gcf, 'plot.pdf')
```

파일 I/O

텍스트 파일

```
data = readmatrix('data.csv');
writematrix(A, 'output.csv')
T = readtable('data.csv');
writetable(T, 'output.csv')
```

MAT 파일

```
save('workspace.mat')           % save all variables
save('data.mat', 'x', 'y')     % save specific vars
load('data.mat')                % load into workspace
S = load('data.mat');           % load into struct
```

저수준 파일 I/O

```
fid = fopen('log.txt', 'w');
fprintf(fid, 'Value: %f\n', 3.14);
fclose(fid);
lines = readlines('log.txt');
```

MATLAB 빠른 참조

문자열 연산

문자열 vs 문자 배열

```
s = "Hello";        % string (double quotes)
c = 'Hello';        % char array (single quotes)
s + " World"        % "Hello World" (string)
[c, ' World']       % 'Hello World' (char concat)
```

문자열 함수

strlength(s)	문자열 길이
upper(s) / lower(s)	대/소문자 변환
contains(s, pat)	패턴 발견 시 true
replace(s, old, new)	부분 문자열 교체
split(s, delim)	배열로 분리
join(arr, delim)	문자열 배열 결합
strip(s)	앞뒤 공백 제거

형식화

```
sprintf('x = %.2f', 3.14159)   % "x = 3.14"
fprintf('i = %d\n', 42)         % print to console
num2str(3.14)                   % number to string
str2double("3.14")              % string to number
```

셀 및 구조체

셀 배열

```
C = {1, 'hello', [1 2 3]};  % mixed types
C{2}                         % access: 'hello'
C{end+1} = true;            % append element
cellfun(@length, C)          % apply func to each
```

구조체

```
s.name = 'Alice';
s.age = 30;
s.scores = [90 85 92];
fieldnames(s)           % {'name','age','scores'}
rmfield(s, 'age')      % remove field
```

구조체 배열

```
people(1).name = 'Alice'; people(1).age = 30;
people(2).name = 'Bob';   people(2).age = 25;
{people.name}           % {'Alice', 'Bob'}
[people.age]            % [30, 25]
```

주요 패턴

벡터화 연산

```
% Avoid loops — use vectorization
v = 1:1000;
result = sum(v.^2);          % fast
idx = v(v > 500 & v < 600);  % logical indexing
```

테이블 연산

```
T = table([25;30], ["A";"B"], 'VariableNames', ...
          {'Age','Grade'});
T.Age             % access column
T(T.Age > 25, :)  % filter rows
```

에러 처리

```
try
    result = riskyFunction(x);
catch ME
    fprintf('Error: %s\n', ME.message);
end
```

코드 타이밍

```
tic
heavyComputation();
toc              % prints elapsed time
```