

Chapter 10 Character Strings

10.1 String Construction

In MATLAB, text is stored as character strings, or strings for short. Strings are numerical arrays of ascii values which are displayed as the equivalent characters. Strings are created by enclosing text within single quotes.

Example 10.1.1

```
clear
s=' Hello there.'
x=size(s)
greeting='Welcome to the University of Central Florida!'
y=size(greeting)
whos

s =
  Hello there.

x =  1    13

greeting =
Welcome to the University of Central Florida!

y =  1    45

  Name          Size          Bytes  Class

greeting      1x45             90  char array
s              1x13             26  char array
x              1x2              16  double array
y              1x2              16  double array

Grand total is 62 elements using 148 bytes
```

The elements of a string array are the individual characters, each of which requires 2 bytes of storage compared to 8 bytes for elements in numerical, i.e. double arrays. The ascii representation of a string is obtained by making it the argument of the function 'double'. The function 'char' restores the character string.

Example 10.1.2

```
s1='text are character strings'
size_s1=size(s1)
z=double(s1) % Convert string to ascii representation
size_z=size(z)
s2=char(z) % Convert numerical array of ascii values back to string
s3=char(100:107) % Convert ascii values 100-107 to characters

s1 =
text are character strings
```

```

size_s1 = 1    26

z =
  Columns 1 through 12
    116    101    120    116     32     97    114    101     32     99    104     97
  Columns 13 through 24
    114     97     99    116    101    114     32    115    116    114    105    110
  Columns 25 through 26
    103    115

size_z = 1    26

s2 =
text are character strings

s3 =
defghijk

```

Strings can be addressed and manipulated the same as arrays. Strings are also concatenated the same as arrays.

Example 10.1.3

```

s1='Greetings sport fans.'
word2=s1(11:15)
s2='baseball'
s3=['Greetings ' s2 ' fans. ']
s4=s3'

s1 =
Greetings sport fans.

word2 =
sport

s2 =
baseball

s3 =
Greetings baseball fans.

s4 =
G
r
e
e
t
i
n
g
s

b
a
s
e

```

```
b  
a  
l  
l  
  
f  
a  
n  
s  
.
```

To display a string without printing its name, the function `disp()` with the string variable name as argument is used.

Example 10.1.4

```
s1='Joe said 'Have a nice day' ' '  
disp(' ')  
disp(s1)  
disp(' ')  
disp('Happy Birthday')
```

```
s1 =  
Joe said 'Have a nice day'  
  
Joe said 'Have a nice day'  
  
Happy Birthday
```

Character strings arrays can have more than one row provided each row is a string with the same number of elements (characters).

Example 10.1.5

```
list=['Harold Klee';'Chris Bauer';'Joe Smith  '];  
    % Two blanks are added to the end of Joe Smith to make it 1 ^ 11,  
    % the same size as strings Harold Klee and Chris Bauer  
disp(list)  
size_list=size(list)
```

```
Harold Klee  
Chris Bauer  
Joe Smith  
  
size_list = 3    11
```

The MATLAB function `'char()'` is also used to create multiple row string arrays from individual strings of different lengths. The function `'strcat()'` is used to horizontally concatenate string arrays with the same number of rows.

Example 10.1.6

```
NLteams=char('N Y Mets','Atlanta Braves','Milwaukee Brewers');
disp(NLteams),disp(' ')
size_NLteams=size(NLteams),disp(' ')
ALteams=char(' Chicago White Sox',' Seattle Mariners',' Texas Rangers');
disp(ALteams),disp(' ')
size_ALteams=size(ALteams),disp(' ')
teams=strcat(NLteams,ALteams);
disp(teams),disp(' ')
size_teams=size(teams),disp(' ')

```

```
N Y Mets
Atlanta Braves
Milwaukee Brewers

```

```
size_NLteams = 3    17

```

```
Chicago White Sox
Seattle Mariners
Texas Rangers

```

```
size_ALteams = 3    18

```

```
N Y Mets Chicago White Sox
Atlanta Braves Seattle Mariners
Milwaukee Brewers Texas Rangers

```

```
size_teams = 3    31

```

When the function 'char()' is used to create string arrays with multiple rows, individual rows may contain padded zeros to assure that all rows are the same length. The padded zeros can be removed after accessing the individual rows ($1 \times n$ string arrays) by 'deblank'ing them.

Example 10.1.7

```
schools=char('Central Florida','Harvard','Florida State');
disp(schools),disp(' ')
FLschools=[schools(1,:);schools(3,:)];
disp(FLschools), disp(' ')
other1=schools(2,:)
size_other1=size(other1),disp(' ')
other2=deblank(schools(2,:))
size_other2=size(other2)

```

```
Central Florida
Harvard
Florida State

```

```
Central Florida
Florida State

```

```
other1 =
Harvard

```

size_other1 = 1 15

other2 =
Harvard

size_other2 = 1 7

10.2 Numbers to Strings to Numbers

Often times it is desirable to convert information from one data type to another. In particular, it is common to transform numerical data into text strings for establishing the flow of executable commands within MATLAB's programming environment. In addition, numbers are sometimes converted to strings for annotation of charts and graphs. The essential MATLAB functions for accomplishing data type conversions involving integer and floating point numbers are 'int2str', 'num2str', and 'sprintf'.

Example 10.2.1

```
x=[1:3,5:7,9:11] % x is an integer array
disp(class(x))
size_x=size(x),disp(' ')
xs=int2str(x) % xs is a character array
disp(class(xs))
size_xs=size(xs),disp(' ')
xs10=xs(1:10) % xs10 is a string array of the first 10 characters of xs
```

```
x = 1     2     3     5     6     7     9    10    11
```

```
double
```

```
size_x = 1     9
```

```
xs =
```

```
1  2  3  5  6  7  9 10 11
```

```
char
```

```
size_xs = 1    33
```

```
xs10 =
```

```
1  2  3
```

```
A=25+5*randn(2,3) % A is a floating point array
```

```
disp(class(A))
```

```
size_A=size(A), disp(' ')
```

```
As=num2str(A) % As is a character array
```

```
disp(class(As))
```

```
size_As=size(As)
```

```
As2=As(2,10:end) % As2 is a portion of the second row of As
```

```
A = 22.8372    25.6267    19.2676
     16.6721    26.4384    30.9546
```

```
double
```

```
size_A = 2     3
```

```
As =
```

```
22.8372    25.6267    19.2676
16.6721    26.4384    30.9546
```

```
char
```

```
size_As = 2 33
```

```
As2 =  
    26.4384    30.9546
```

The function 'sprintf' is similar to the ANSI C function and is used for creating character strings which can be displayed or manipulated. Very often the string is comprised of text and numerical data stored in variables which is converted to character strings according to a specified format conversion.

Example 10.2.2

```
format long  
p=pi*1000, disp(' ')  
p_s1=sprintf('%.5e\n',p) % Exponential conversion and display 5  
                        % places after the decimal point  
p_s2=sprintf('%.5f\n',p) % Fixed point conversion and display 5  
                        % places after decimal point  
p_s3=sprintf('%.5g\n',p) % General (e or f) conversion and display a  
                        % total of 5 digits
```

```
p = 3.141592653589793e+003
```

```
p_s1 = 3.14159e+003
```

```
p_s2 = 3141.59265
```

```
p_s3 = 3141.6
```

Example 10.2.3

```
n=500;  
x=rand(1,n);  
ave=mean(x);  
s=sprintf('The average of %.0f random numbers is %.3f.',n,ave);  
disp(s)  
disp(' ')
```

```
The average of 500 random numbers is 0.507.
```

There are occasions when it may necessary to convert from a character string to a number. The MATLAB funtions 'str2num' and 'eval' are applicable.

Example 10.2.4

```
s='2500'  
disp(class(s))  
size_s=size(s)  
x=str2num(s)  
disp(class(x))  
size_x=size(x)
```

```

s =
2500

char

size_s = 1      4

x = 2500

double

size_x = 1      1

```

The argument of 'str2num' must be a number, however 'eval' accepts expressions involving MATLAB variables or other valid MATLAB statements.

Example 10.2.5

```

n=11;
xs='1:2:n'
disp(class(xs))
size_xs=size(xs),disp(' ')
x1=str2num(xs) % xs is not a number
x2=eval(xs)
disp(class(x2))
size_x2=size(x2),disp(' ')

xs =
1:2:n

char

size_xs = 1      5

x1 = []

x2 = 1      3      5      7      9      11

double

size_x2 = 1      6

```

In addition, 'eval' can be called with two string arguments where the second is evaluated if the first produces an error.

Example 10.2.6

```

n=5;
As='[n^2, rand; pi,cos(pi)]'
disp(' ')
A=eval('rand(2,3)^2',As) % The first argument cannot be evaluated
disp(' ')
size_A=size(A)

```



```
As =  
[n^2, rand; pi,cos(pi)]  
  
A =  
25.000000000000000 0.62113870578822  
3.14159265358979 -1.000000000000000  
  
size_A = 2 2
```

10.3 String Functions

There are an assortment of MATLAB string inquiry and manipulation functions for use with character strings. The table below includes a partial list of functions from the text followed by several illustrative examples.

Function	Description
ischar(S)	True if argument is a character string
isletter(S)	True for indices that contain letters
isspace(S)	True for indices that contain white space
findstr(S1,S2)	Find indices where one string appears in the other
strcmp(S1,S2)	True if two strings are identical
strncmp(S1,S2,n)	True if first n characters of two strings are identical
strmatch(S1,S2)	Return row indices of S2 that contains S1 as first elements
strrep(S1,S2,S3)	Replace all occurrences of S2 in S1 with S3

Example 10.3.1

```

s1='Bon Jour',disp(' ') % Create character string s1
x1=[1 2 3],disp(' ') % Create numeric array x1
check_s1=ischar(s1),disp(' ') % Check if s1 is a character string
check_x1=ischar(x1),disp(' ') % Check if x1 is a character string
s2='1 plus 2 equals 3',disp(' ') % Create string s2
size_s2=size(s2),disp(' ') % Display size of s2
check_s2_l=isletter(s2),disp(' ') % Find indices of letters in s2
check_s2_w=isspace(s2),disp(' ') % Find indices of blanks in s2

s1 =
Bon Jour

x1 = 1 2 3

check_s1 = 1
check_x1 = 0

s2 =
1 plus 2 equals 3

size_s2 = 1 17

check_s2_l =
Columns 1 through 12
0 0 1 1 1 1 0 0 0 1 1 1
Columns 13 through 17
1 1 1 0 0

```

```

check_s2_w =
  Columns 1 through 12
    0    1    0    0    0    0    1    0    1    0    0    0
  Columns 13 through 17
    0    0    0    1    0

```

Example 10.3.2

```

you='Have a really great day',disp(' ') % Create string you
me='Let's all have a great day today',disp(' ') % Create string me
findstr('great day',you),disp(' ') % Find starting index in string you
    % where string 'great day' begins
findstr('day',me),disp(' ') % Find where 'day' starts in string me
strcmp(you,me),disp(' ') % Check if strings you and me are identical
strcmp(you(21:23),me(24:26)) % Check if indices 21-23 in string you are
    % identical to indices 24-26 in string me

```

```

you =
Have a really great day

me =
Let's all have a great day today

ans = 15

ans = 24    30

ans = 0

ans = 1

```

Example 10.3.3

```

x1s=num2str(100+5*rand); % Create character string x1s
x2s=num2str(100+5*rand); % Create character string x2s
disp(x1s)
disp(x2s),disp(' ')
strncmp(x1s,x2s,1) % Check if first character of x1s and x2s are same
strncmp(x1s,x2s,2) %Check if first 2 characters of x1s and x2s are same
strncmp(x1s,x2s,3) %Check if first 3 characters of x1s and x2s are same
strncmp(x1s,x2s,4) %Check if first 4 characters of x1s and x2s are same
strncmp(x1s,x2s,5) %Check if first 5 characters of x1s and x2s are same

102.8011
101.2202

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

```

Example 10.3.4

```
states=char('Virginia', 'New York', 'California', 'New Mexico',  
'Florida', 'New Jersey')  
strmatch('New',states) % Find row indices of character string array  
% states that begin with the string 'New'
```

```
states =  
Virginia  
New York  
California  
New Mexico  
Florida  
New Jersey
```

```
ans =  
     2  
     4  
     6
```

Example 10.3.5

```
lie='You represent the ''best'' of the ''best''! ';  
disp(lie)  
truth=strrep(lie,'best','worst'); % Replace  
disp(truth)
```

```
You represent the 'best' of the 'best'!
```

```
You represent the 'worst' of the 'worst'!
```

10.4 Cell Arrays of Strings

A data type capable of holding different types of data, e.g. numbers and character strings is called a cell array. A cell array consists of cells in a rectangular structure with rows and columns. The cells located in each column contain a specific data type. If a cell contains a character string, its length can differ from character strings in other cells of the same column. Curly brackets { } are used to create cell arrays.

Example 10.4.1

```
clear
format bank
C{1,1}='Derek Jeter'; % Name
C{1,2}='NY Yankees'; % Team
C{1,3}='ss'; % Position
C{1,4}=332; % Batting Ave
C{1,5}=[17,87]; % Home runs and rbi's
C{2,1}='Mike Piazza';
C{2,2}='NY Mets';
C{2,3}='c';
C{2,4}=351;
C{2,5}=[34,105];
C{3,1}='Sammy Sosa';
C{3,2}='Chicago Cubs';
C{3,3}='rf';
C{3,4}=305;
C{3,5}=[48,118];
C{4,1}='Mark McGwire';
C{4,2}='St Louis Cardinals';
C{4,3}='1b';
C{4,4}=287;
C{4,5}=[29,83];
C,disp(' ')
size(C)

C =
    'Derek Jeter'      'NY Yankees'      'ss'      [332.00]      [1x2
double]
    'Mike Piazza'     'NY Mets'         'c'       [351.00]      [1x2
double]
    'Sammy Sosa'      'Chicago Cubs'    'rf'      [305.00]      [1x2
double]
    'Mark McGwire'    'St Louis Cardinals' '1b'      [287.00]      [1x2
double]

ans = 4.00          5.00
```

The cells are addressed using parenthesis () similar to an ordinary array.

Example 10.4.2

```
Player_1_Name=C(1,1) % Address cell (1,1) and store in Player_1_Name
disp(class(Player_1_Name)) % Player_1_Name is a cell array array also
```

```

Player_1_Name =
    'Derek Jeter'

cell

```

The contents of a cell are addressed using the curly brackets.

Example 10.4.3

```

Player_2_Name=C{2,1} % Retrieve contents of cell (2,1) and store in
                    % Player_2_Name
disp(class(Player_2_Name)) % Player_2_Name is a character string
size(Player_2_Name) % Number of characters in Player_2_Name

Player_2_Name =
Mike Piazza

char

ans =  1.00      11.00

Player_3_BA=C{3,4} % Retrieve contents of cell (3,4) and store in
                % Player_3_BA
disp(class(Player_3_BA)) % Player_3_BA is a double
size(Player_3_BA) % Size of array Player_3_BA

Player_3_BA =  305.00

double

ans =  1.00      1.00

Player_4_Hr_rbi=C{4,5} % Retrieve contents of cell (4,5) and store
                    % in Player_4_Hr_rbi
disp(class(Player_4_Hr_rbi)) % Player_4_Hr_rbi is a double
size(Player_4_Hr_rbi) % Size of array Player_4_Hr_rbi

Player_4_Hr_rbi =  29.00      83.00

double

ans =  1.00      2.00

ave=(C{1,4}+C{2,4}+C{3,4}+C{4,4})/4 % Calculate average batting ave

ave =  318.75

```

If a given cell contains a character string or a numeric array, other than a 1x1, a portion of the cell's contents can be addressed.

Example 10.4.4

```
A{1,1}='Monday';
A{1,2}= [75,92]; % Low and high temperature for day 1
A{2,1}='Friday';
A{2,2}=[25,38]; % Low and high temperature for day 2
A, disp(' ')
Day_1=A{1,1}(1:3),disp(' ') % Find the first 3 letters of the day
Day_1_Low_temp=A{1,2}(1) % Find the 1st component of numeric array
                    % in cell (1,2)
Day_2_High_temp=A{2,2}(2) % Find the 1st component of numeric array
                    % in cell (2,2)
```

```
A =
    'Monday'    [1x2 double]
    'Friday'    [1x2 double]
```

```
Day_1 =
Mon
```

```
Day_1_Low_temp =    75.00
Day_2_High_temp =    38.00
```