15. Strings

Operations
Subscripting
Concatenation
Search
Numeric-String Conversions

Built-Ins: int2str, num2str, str2double

Previous Dealings

N = input( 'Enter Degree:' )
title('The Sine Function')
disp( sprintf('N = %2d',N) )

A String is an Array of Characters

'Aa7>*@ x!'

This string has length 9.

Why Important

1. Numerical Data often encoded as strings
2. Genomic calculation/search

Numerical Data is Often encoded in Strings

For example, a file containing Ithaca weather data begins with the string

W07629N4226

Longitude: 76° 29′ West
Latitude: 42° 26′ North

What We Would Like to Do

Get hold of the substring '07629'

Convert it to floating format so that it can be involved in numerical calculations.
Format Issues

9 as an IEEE floating point number:

```
0100000blablahblah01001111000100010010
```

9 as a character:

```
01000otherblalblal
```

Different Representation

Genomic Computations

Looking for patterns in a DNA sequence:

`'ATTCTGACCTCGATC'`

ACCT

Genomic Computations

Quantifying Differences:

ATTCTGACCTCGATC

ATTGCTGACCTCGAT

Remove?

Working With Strings

Strings Can Be Assigned to Variables

```
S = 'N = 2'
N = 2;
S = sprintf('N = %1d',N)
```

sprintf produces a formatted string using fprintf rules

Strings Have a Length

```
s = 'abc';
n = length(s);  % n = 3
s = '';         % the empty string
n = length(s)   % n = 0
s = ' ';        % single blank
n = length(s)   % n = 1
```
**Concatenation**

This:

\[
\begin{align*}
S &= 'abc'; \\
T &= 'xy' \\
R &= [S \ T]
\end{align*}
\]

is the same as this:

\[
R = 'abcxy'
\]

**Repeated Concatenation**

This:

\[
\begin{align*}
s &= ''; \\
\text{for } k=1:5 \\
s &= [s 'z']; \\
\text{end}
\end{align*}
\]

is the same as this:

\[
z = 'zzzzz'
\]

**Replacing and Appending Characters**

\[
\begin{align*}
s &= 'abc'; \\
s(2) &= 'x' \quad \% s = 'axc' \\
t &= 'abc' \\
t(4) &= 'd' \quad \% t = 'abcd' \\
v &= '' \\
v(5) &= 'x' \quad \% v = ' x'
\end{align*}
\]

**Extracting Substrings**

\[
\begin{align*}
s &= 'abcdef'; \\
x &= s(3) \quad \% x = 'c' \\
x &= s(2:4) \quad \% x = 'bcd' \\
x &= s(\text{length}(s)) \quad \% x = 'f'
\end{align*}
\]

**Colon Notation**

\[
s(\begin{array}{cc}
\text{Starting Location} & \text{Ending Location}
\end{array})
\]

**Replacing Substrings**

\[
\begin{align*}
s &= 'abcde'; \\
s(2:4) &= 'xyz' \quad \% s = 'axyze' \\
s &= 'abcde' \\
s(2:4) &= 'wxyz' \quad \% \text{Error}
\end{align*}
\]
**Question Time**

```matlab
s = 'abcde';
for k=1:3
    s = [ s(4:5) s(1:3)];
end

What is the final value of s?
```

A. abcde  B. bcdea  C. eabcd  D. deabc

**Problem: DNA Strand**

x is a string made up of the characters 'A', 'C', 'T', and 'G'.

Construct a string Y obtained from x by replacing each A by T, each T by A, each C by G, and each G by C.

```
x: ACGTTGCAGTTCCATATG
y: TGCAACGTCAAGGTATAC
```

**Comparing Strings**

Built-in function `strcmp`

`strcmp(s1, s2)` is true if the strings `s1` and `s2` are identical.

**How y is Built Up**

```
x: ACGTTGCAGTTCCATATG
y: TGCAACGTCAAGGTATAC
```

Start: y: '
After 1 pass: y: T
After 2 passes: y: TG
After 3 passes: y: TGC

```matlab
function y = DNA(x)
    % x is a string consisting of
    % the characters A, C, T, and G.
    % y is a string obtained by
    % replacing A by T, T by A, 
    % C by G and G by C.

    x = 'ACGTTGCAGTTCCATATG';
    y = '';
    for k=1:length(x)
        if strcmp(x(k),'A')
            y = [y 'T'];
        elseif strcmp(x(k),'T')
            y = [y 'A'];
        elseif strcmp(x(k),'C')
            y = [y 'G'];
        else
            y = [y 'C'];
        end
    end
end
```
A DNA Search Problem

Suppose $S$ and $T$ are strings, e.g.,

$S$: ‘ACCT’

$T$: ‘ATGACCTGA’

We’d like to know if $S$ is a substring of $T$ and if so, where is the first occurrence?

function $k = \text{FindCopy}(S,T)$

% $S$ and $T$ are strings.
% If $S$ is not a substring of $T,$
% then $k=0.$
% Otherwise, $k$ is the smallest integer so that $S$ is identical
% to $T(k:k+\text{length}(S)-1).$

A DNA Search Problem

\text{strcmp}(S,T(1:4)) \quad \text{False}

A DNA Search Problem

\text{strcmp}(S,T(2:5)) \quad \text{False}

A DNA Search Problem

\text{strcmp}(S,T(3:6)) \quad \text{False}

A DNA Search Problem

\text{strcmp}(S,T(4:7)) \quad \text{True}
Pseudocode

First = 1; Last = length(S)
while S is not identical to T(First;Last)
    First = First + 1;
    Last = Last + 1
end

Subscript Error

S: ‘ACCT’
T: ‘ATGACTGA’

strcmp(S,T(6:9))

There’s a problem if S is not a substring of T.

Pseudocode

First = 1; Last = length(s)
while Last<=length(T) && ~strcmp(S,T(First:Last))
    First = First + 1;
    Last = Last + 1
end

Post-Loop Processing

Loop ends when this is false:
Last<=length(T) && ~strcmp(S,T(First:Last))

Post-Loop Processing

if Last>length(T)
    % No Match found
    k=0;
else
    % There was a match
    k=First;
end

The loop ends for one of two reasons.

Numeric/String Conversion
**String-to-Numeric Conversion**

An example...

Convention:

```
W07629N4226
```

Longitude: 76° 29’ West
Latitude: 42° 26’ North

---

```
S = 'W07629N4226'
s1 = s(2:4);
x1 = str2double(s1);
s2 = s(5:6);
x2 = str2double(s2);
Longitude = x1 + x2/60
```

There are 60 minutes in a degree.

---

**Numeric-to-String Conversion**

```
x = 1234;
s = int2str(x);         % s = '1234'
x = pi;
s = num2str(x,'%5.3f');   % s = '3.142'
```

---

**Problem**

Given a date in the format

‘mm/dd’

specify the next day in the same format

---

**Y = Tomorrow(x)**

```
x             y
02/28         03/01
07/13         07/14
12/31         01/01
```

---

**Get the Day and Month**

```
month = str2double(x(1:2));
day = str2double(x(4:5));
```

Thus, if `x = '02/28'` then month is assigned the numerical value of 2 and day is assigned the numerical value of 28.
L = [31 28 31 30 31 30 31 31 30 31 30 31];
if day+1<=L(month)
    % Tomorrow is in the same month
    newDay = day+1;
    newMonth = month;
else
    % Tomorrow is in the next month
    newDay = 1;
    if month<12
        newMonth = month+1;
    else
        newMonth = 1;
    end
end

The New Day String
Compute newDay (numerical) and convert...

d = int2str(newDay);
if length(d)==1
    d = ['0' d];
end

The New Month String
Compute newMonth (numerical) and convert...

m = int2str(newMonth);
if length(m)==1
    m = ['0' m];
end

The Final Concatenation

y = [m '/' d];