CS368 MATLAB Programming
Lecture 4

Young Wu

Based on lecture slides by Michael O’Neill and Beck Hasti

February 16, 2022
Surface Plotting, Bowl

Quiz
Surface Plotting, Bowl

Quiz
ASCII Code

ASCII stands for American Standard Code for Information Interchange.

Each character is stored as an integer (its ASCII code).

- 0 to 9 are stored as 48 to 57.
- A to Z are stored as 65 to 90.
- a to z are stored as 97 to 122.
A string is a list of characters.

A string is stored as a (row) vector of integers with `char` variable type in MATLAB.

'Hello World!' is a string, and

`char([72 101 108 108 111 32 87 111 114 108 100 33])` represents the same string.
Combining Strings

Code

- Two strings can be combined the same way two vectors are combined, for example, `['Hello' 'World' '! ']` is the same as `'Hello World!'`.

- `append(x, y, ...)` also combines the strings `x, y, ...`, for example, `append('Hello ', 'World', '! ')` returns `'Hello World!'`.

- `strcat(x, y, ...)` combines (or conCATenate) the strings `x, y, ...`, and removes trailing spaces, for example, `strcat ('Hello ', 'World', '!' )` returns `'HelloWorld!'`. 
String Conversion

Code

- **num2str(x, n)** converts a number x (not ASCII code) to a string, rounded to n significant digits (different from n decimal places), for example, **num2str(pi, 4)** is the same as '3.142' or **char([51 46 49 52 50])**.

- **str2num(x)** converts a string back to a number or a matrix, for example, **str2num('3.142')** returns the number 3.142 and **str2num('1 2; 3 4')** returns the matrix $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$.
String Operations, Addition

Quiz
String Operations, Number Conversion

Quiz
String Operations, as Vectors

Quiz
In MATLAB, there is `string` variable type that stores multiple characters as a single object so that multiple strings can be stored in a vector without getting combined into one, for example, `['a' 'b' 'c']` is the same as `'abc'` but `['a' 'b' 'c']` stays a vector and `''a'' + ''b'' + ''c''` is the same as `''abc''`.

To convert between the two types of strings, `string ('abc')` becomes `''abc''` and `char('''abc''')` becomes `'abc'`. 
Useful String Functions, Comparison

Code

- `strcmp(x, y)` compares two strings `x` and `y` and returns 1 if they are the same and 0 otherwise, for example, `strcmp('abc', ['a' 'b' 'c'])` returns 1 and `strcmp('abc', 'AbC')` returns 0.

- `strcmpi(x, y)` compares two strings `x` and `y` ignoring cases, for example, `strcmpi ('abc', 'AbC')` returns 1.

- `upper(x)` and `lower(x)` converts the string `x` to upper and lower cases.
Useful String Functions, Find and Replace

**Code**

- `strfind(x, y)` finds the indices of all occurrences of `y` in `x`, for example, `strfind('aabb', 'a')` return `[1 2]`.
- `strrep(x, o, n)` or `replace(x, o, n)` replaces all occurrences of `o` in `x` by `n` and returns the new string, for example, `strrep('aabb', 'b', 'c')` returns `'aacc'` and `replace('aabb', ['a', 'b'], ['c', 'd'])` returns `'ccdd'`. 
Special Text Symbols

**Code**

- `blanks(n)` creates a string with `n` spaces.
- `''` (two single quotation marks, not one double quotation mark) is `.``
- `%%` is `%`.
- `\` is backslash `.``
- `\n` is new line.
- `\t` is tab.
disp(x) displays the string x. It does not store x in the variable ans.

fprintf(x, v1, v2, ...) displays a string with %s (string), %i (integer), %f (floating point), %e (scientific notation) replaced by v1, v2, ...
Add a number after % to set the field width (text length) for the string, for example, %5s and %—5i make sure that the displayed string has length ≥ 5 by adding spaces when necessary. A positive number means added spaces are on the left and a negative number means added spaces are on the right.

Add a . followed by a number for %f to set the precision, the number of digits after the decimal point, for example, %.4f rounds the number to 4 decimal places, adding 0s when necessary.
Text Formatting, Integer

Quiz
Text Formatting, Floating Quiz
Text Input

Code

- \texttt{input(x)} gets a user input in MATLAB syntax. String \(x\) is the prompt.
- \texttt{input(x, 's')} gets a user input as a string.
- Sometimes the user input may need to be validated or reformatted before being used in subsequent computations. More details in a later lecture.
- \texttt{menu(x, c_1, c_2, \ldots)} or \texttt{listdlg('ListString', c_1, c_2, \ldots, 'PromptString', x)} gets a user input from a list of choices \(c_1, c_2, \ldots\), and returns the index.
Text Input, Input

Code
Text Input, Multiple Inputs

Code
Text Input, Options

Code
File Input

Code

- `load(x, '— ascii ')` loads the text file with name `x`.
- `load(x)` can load a `.mat` binary file.
- `readmatrix(x)` loads the text or spreadsheet file with name `x` into a single matrix.
- Under the ”HOME” tab, there is a ”Import Data” tool that can be used to import data in various formats from files.
save(x, v, ..., '— ascii ') saves the variables with names v, ..., to the file with name x.

save(x, v, ...) saves the variables in a .mat binary file, not human-readable.

writematrix (v, x) saves the variable v to the file with name x.
String File Input Output

Code

- **fileread** \((x)\) reads the text file with name \(x\) as a string with *char* type.
- **readlines** \((x)\) reads the text file with name \(x\) as a vector of lines, each line has the *string* type.
- **fopen**\((x, \ 'w'\);  **fprintf** \((x, v);  **fclose** \((x);\) writes the string \(v\) to the file with name \(x\).