

CS368 MATLAB Programming

Lecture 11

Young Wu

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

April 13, 2022

Errors

Math

- Syntax error is an error in spelling or grammar.
- MATLAB displays red messages for syntax errors, so they are easy to find and fix.
- Semantic error is an error in meaning or logic.
- For small programs, compare the program outputs with expected outputs computed by hand to find and fix the semantic error.
- For large programs, break into smaller programs and debug each one.

Debugger

Code

- A debugger can set break points.
- A break point stops the program so that the current variable values can be viewed in *Workspace*.
- It is useful to check if loops and conditionals are written correctly.

User Input Validation

Code

- Use a while loop to keep asking for user input until the input is valid.

```
1 valid = 0;  
2 while ~valid  
3     x = input (...)  
4     if ... % check valid  
5         valid = 1;  
6     end  
7 end
```

Input Validation Membership Functions

Code

- *isletter* (*x*) or *isstrprop* (*x*, 'alpha') checks if the string *x* contains characters that are letters 'a' to 'z' or 'A' to 'Z'.
- *isstrprop* (*x*, 'digit') checks if the string *x* contains characters that are numbers '0' to '9'.
- *ismember*(*x*, *y*) checks if the string *x* contains characters that are characters in the other string *y*.
- These functions treat *x* as a vector and returns a Boolean vector specifying whether each character in *x* belongs to some set of characters.

Input Validation Substring Functions

Code

- *contains(x, y)* checks if the string x contains a substring y .
- *count(x, y)* counts the number of occurrences of y in the string x .
- *startsWith(x, y)* and *endsWith(x, y)* checks if the string x starts with or ends with the substring y .
- These functions treat x as a single string and checks if it contains certain substrings.

Input Validation, Check Contain

Quiz

Input Validation, Check Combination Quiz

Input Validation, Check Permutation

Quiz

Input Argument Validation

Code

- The inputs to a function can be validated so that an input that does not satisfy the conditions will cause an error instead of incorrect outputs.

- 1 *function f(x)*
- 2 *arguments*
- 3 *x (size) class {functions} = default value*
- 4 *end*
- 5 *... % actual function*
- 6 *end*

Input Argument Size

Code

- In x (*size*) *class* {*functions*}, the size is specified by a comma-separated list.
- x (n , m) ... requires x to be an $n \times m$ matrix.
- x (n , :) ... requires x to be a matrix with n rows or a vector with n elements.
- x (:, m) ... requires x to be a matrix with m columns.

Input Argument Class

Code

- In `x (size) class {functions}`, the class is specified by its class name.
- `x char ...` and `x string ...` require `x` to be a string.
- `x single ...` and `x double ...` require `x` to be a number.
- `x logical ...` requires `x` to be a Boolean variable.

Input Argument Validation Function

Code

- In x (*size*) *class* {*functions*}, the functions are special functions that raise an error when some conditions are not satisfied.
- For example, x {*mustBeGreaterThanOrEqualTo*(x , l), *mustBeLessThanOrEqualTo*(x , u)} ... requires x to be between l and u , and raises an error when $x < l$ or $x > u$.
- For example, x {*mustBeMember*(x , [u v w])} requires x to be one of u or v or w .

Function of Functions

Math

- A functional (noun.) is a function can take another function as an input, or returns another function as an output. Functionals are also called higher-order functions.
- The differential and integral operators are functionals: they take a function as input and outputs another function (or a scalar).

Function Handle

Code

- $y = @f$ creates a variable y that represents the function f . The variable y is a function handle.
- Function handles provide a way to pass a function as an input argument to another function.

Anonymous Functions

Code

- $y = @(x) f(x)$ creates an anonymous function and stores it in the variable named y .
- Anonymous functions provide a way to write a function handle without defining a separate file for the function.

Function Max at Value Example

Code

- Another version of the max function can be defined to find the maximum of two functions f and g at a point x .
- ① `function mfg = maxFun(f, g, x)`
- ② `mfg = max(f(x), g(x));`
- ③ `end`
- For example, `maxFun(@sin, @(x)(sin(2 * x)), 1)` finds the maximum between $\sin(1)$ and $\sin(2 \cdot 1)$.

Function Max Example

Code

- Another version of the max function can be defined to find the maximum of two functions f and g and return a function.

```
1 function mfg = maxFun(f, g)
2   mfg = @(x)(max(f(x), g(x)));
3 end
```

- For example, if $h = \text{maxFun}(@\sin, @(x)(\sin(2 * x)))$, then $h(1)$ finds the maximum between $\sin(1)$ and $\sin(2 \cdot 1)$.

Function Handle, Max

Quiz

Function Handle, Noise

Quiz

Function Handle, Random Noise

Quiz

Blank Slide