CS100M/CIS121/EAS121
Introduction to Computer Programming

Spring 2004
Lecture 3
MATLAB Language

Announcements
- CS100M sections start today
- http://courses.cs.cornell.edu/cs100m
- Staff reminders: consultants (Carpenter B101, TAs Upson 328, Course Admin patwell@cs)
- Prelim 1 conflicts: e-mail Kelly
- readings reminder
- when to e-mail DIS?
- A1 due Weds, 11:59pm

Objectives
- Need for written programming language
- So, you get more MATLAB!
  - alphabet (character set)
  - words (tokens)
  - sentences (statements/commands)
  - paragraphs (functions: more later)
  - documents (programs: what you're doing!)
- IMPORTANT: Readings from CHAPTER 2!

Why Programming Language?
- Problems are stated given, often written
- Could just go ahead and solve problem
  - Very useful to remember what you did
  - Want to reproduce
  - Want someone/something else also to solve
- How do we record our solution process?
  - what did we do last time?
  - solve problem once and not worry about implementation (languages often change)
Language

- So, algorithms are great.
- But what about when you do need to implement?
- Computers need very precise instructions
  - Algorithms good for humans, bad for computers
  - Also, want way to solve problem once and implement as new languages come along
- Need for precise written language
  - Programming involves translation!

Elements of Language

- **Alphabet**
  - look at your keyboard for *character set*
  - typical: ASCII, UNICODE
- **Words**:
  - meaningful clumps of characters
  - *tokens*
  - separate with *whitespace* and *punctuation*

Language Elements (continued)

- **Sentences**:
  - clumps of words and separators
  - *statements*: specific instructions
- **Paragraphs and documents**:
  - clumps of sentences
  - modules/functions/other structures...

Study of Languages

- **Syntax**
  - spelling
  - position/structure
  - eg) Spyre blorg the !
- **Semantics**
  - meaning
  - eg) I eat food!
- **Rules?**
Comments

- Not really a token
- Inert, program won't act on
- way to provide messages for other programmers
- good style!
- use % at beginning of line
- MATLAB ignores everything after the %

Characters

- MATLAB character set
- ASCII
  - ASCII has 128 characters
  - see kb.gif for some name
  - some are nonprinting -- see ascii.txt for full list
  - also, try entering char(32:127)
- UNICODE

Tokens

- Rem: words in a language
- eg)

\[
\text{disp(}'Hello, world!'\text{'}) \\
\text{x = 1:4;} \\
\text{plot(x)}
\]

spot the tokens? categories?
- values
- operators
- names
- more!

Tokens: whitespace

- Whitespace
  - blank spaces before, between, and after tokens
  - use as much as you want, but do not split a token
  - eg) he  lp
- See also comments
Tokens: numbers

• Numbers
  - problems often have quantities
• Examples:
  - 1 % integer (whole number)
  - 1.1 % floating point number
  - 1e2 % floating point number
  - 27i % imaginary
• see also format

Tokens: special values

• Special values:
  - pi
  - i, j
  - Inf
  - Nan
  - eps
  - clock, date
  - ans
• See Chapman 2.5

Tokens: logic and character values

• Logical values:
  - 0: false
  - 1: true
• Characters and strings:
  - ASCII characters
  - eg) x = 'd'
  - eg) y = 'abcd'
  - eg) double('a')
  - eg) char(97)

Tokens: names

• clumps of characters that have special purpose
  - Want to name something...
  - Identify/store/associate values and statements
• Values:
  - identifiers: variables that can hold different values
  - eg) x = 17 , y = 's'
• Statements:
  - give names to scripts and functions
  - eg) x + sqrt(4)
  - eg) name of M-Files
Some rules for names

- must begin with letter
- can contain other letters, numbers, underscore
- case-sensitive!
- max length? see `namelengthmax`

Tokens: keywords

- Review of “clumps of letters”:
  - characters and strings, which are values
  - identifiers and function/script names
  - but there’s one more kind...
- **keyword**
  - special, unchangeable word: part of language!
  - languages use most common pseudocode
  - MATLAB: `if else elseif for while`
    see `iskeyword` for full list
  - eg) `for ii=1:4, disp('Slim Shady'), end;`

Tokens: operators

- Operators
  - like functions, but built-in actions
  - act on values
- Common categories
  - arithmetic: `x, +, -, *, /` and more!
  - logic: `&&, ||, ~, ==, >, <` and more!
  - assignment: `=`
  - eg) `18 == 19 || 20 >= 20`
- `help ops`

Tokens: punctuation

- Some “operators” really just separate tokens
- arithmetic? use parens
  eg) `(1 + 2) / 3`
- suppress output: use semicolon
  eg) `x = 1 ;`
- separate statements: use semicolon, comma, return
  eg) `x = 1; y = 2, z = 3 % return d = 2`
Everything in MATLAB is actually an array

- Even scalars are arrays
  - examples:
    - [1 2 3 4]
    - [1 2 3 ; 4 5 6]
    - 1:4
    - 0:2:10
    - 10:-2:0

Tokens: arrays

Some Statements

- Expression
  - empty
  - arithmetic
  - string
  - boolean
  - function call
- More?
  - assignment
  - input/output
  - selection (choosing)
  - repetition (repeating)