CN208
Introductory Computer Programming

Week 3:- Programming Concept

By

Dr. Piya Techateerawat

Programming Concept

- **Programming Overview**
- Objects
- Program Statement Types

Programming Overview

- Why?
  - Problem is very hard to solve.
  - Problem is very complex to solve.
  - Problem is repeating to solve.
  - Problem needs AI to solve.
  - Etc.

Programming Overview

- What computer can do ?
  - Computer Hardware is Stupid!.
  - Mainly, it can only store and do calculation.
  - In calculation is not much different in ability comparing to scientific calculator.
- So why we still need computer ?
  - Because programming can combine memory, function, method and calculation to solve the problems (complex, difficult, repeating and AI).

Programming Overview

- What programmer does ?
  - Understand the problem
  - Break the problems to small pieces
  - Design the method to solve each piece of problem
  - Coding by manipulate syntax, function, procedure, ...
  - Test each solution individually
  - Test the combine solution
  - Documentation

Programming Overview

- To able to solve the problem, programmer must understand the tools of programming.
- So, programmer can design the proper solution to the proper problem.
- To design the solution without understanding the tools, it can be misguided and lost.

Programming Overview

Abstraction

- Mr. Smith bakes a cake

- Data Abstraction
  - Mr. Smith
  - A cake

- Procedural Abstraction
  - bakes

Programming Overview

Functional Programing: Lisp

Procedural Programing: C Language

Object Oriented: C++

Programming Concept

- Programming Overview
- **Objects**
- Program Statement Types

Objects

- Objects is data items passed by the computer .
  - Single-bit integer
  - Multi-bit signed integer
  - Multi-bit unsigned integer
  - Floating –point numbers
  - Character
  - String
  - Pointer

Objects

- Objects is data items passed by the computer .
  - Single-bit integer
  - Multi-bit signed integer
  - Multi-bit unsigned integer
  - Floating –point numbers
  - Character
  - String
  - Pointer

Programming Concept

- Programming Overview
- Objects
- **Program Statement Types**
- Data Structures

Programming Statement Types

- Programming Statement is procedure or action that manipulate the objects

  e.g. store, moving, modify or logical operations.
  - Constants, variables and data types.
  - Arithmetic, string and logical operations
  - Assignment statements
  - Conditional statements
  - Program control functions and statements
  - Input and output functions and statements.

Programming Statement Types

- Constants, variables and data types.
  - Decimal Integer 12345
  - Hexadecimal 1AB34
  - Binary Integer 01101101
  - Floating Point 3.141592
  - Character 'A'
  - String "Ant is an animal"

Programming Statement Types

- Arithmetic, string and logical operations
  - Arithmetic 1 + 2 – 3 x 4 / 5
  - String concatenate "Ant" & "A"
  - Logical NOT, AND, OR, XOR

Programming Statement Types

- Assignment statements
  - A =3
  - B = 4

Programming Statement Types

- Conditional statements
  - If a>0 then a=a+1;
  - Select
  - Case, Switch
  - Etc.

Programming Statement Types

- Program control functions and statements
  - Loop

```
For I=0 to 10
{
    a=a+1;
 }
```

  - Function

```
test(int a);
b= a + 1;
Return(b);

b=test(1)
printf(" Answer is  %i\n", b)

Answer is 2
```

Programming Statement Types

- Input and output functions and statements.

```
printf("Enter an integer: ");
scanf(""%i", &int1);
int1 = int1 * 2;
printf(" Double value is %i", int1);
```

Programming Statement Types

- Constants, variables and data types.
- Arithmetic, string and logical operations
- Assignment statements
- Conditional statements
- Program control functions and statements
- Input and output functions and statements.

Programming Concept

- Programming Overview
- Objects
- Program Statement Types

MATLAB Sample

```
% (1) Basics

% The symbol "%" is used to indicate a comment (for the remainder of
% the line).

% When writing a long Matlab statement that becomes to long for a
% single line use "..." at the end of the line to continue on the next
% line.  E.g.

A = [1, 2; ...
     3, 4];

% A semicolon at the end of a statement means that Matlab will not
% display the result of the evaluated statement. If the ";" is omitted
% then Matlab will display the result.  This is also useful for
% printing the value of variables, e.g.

A

% Matlab's command line is a little like a standard shell:
% - Use the up arrow to recall commands without retyping them (and
%   down arrow to go forward in the command history).
% - C-a moves to beginning of line (C-e for end), C-f moves forward a
%   character and C-b moves back (equivalent to the left and right
%   arrow keys), C-d deletes a character, C-k deletes the rest of the
%   line to the right of the cursor, C-p goes back through the
%   command history and C-n goes forward (equivalent to up and down
%   arrows), Tab tries to complete a command.

% Simple debugging:
% If the command "dbstop if error" is issued before running a script
% or a function that causes a run-time error, the execution will stop
% at the point where the error occurred. Very useful for tracking down
% errors.
```

MATLAB Sample

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% (2) Basic types in Matlab

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (A) The basic types in Matlab are scalars (usually double-precision
% floating point), vectors, and matrices:

A = [1 2; 3 4];              % Creates a 2x2 matrix
B = [1,2; 3,4];              % The simplest way to create a matrix is
                             % to list its entries in square brackets.
                             % The ";" symbol separates rows;
                             % the (optional) "," separates columns.

N = 5                        % A scalar
v = [1 0 0]                  % A row vector
v = [1; 2; 3]                % A column vector
v = v'                       % Transpose a vector (row to column or
                             %   column to row)
v = 1:.5:3                   % A vector filled in a specified range:
v = pi*[-4:4]/4              %   [start:stepsize:end]
                             %   (brackets are optional)
v = []                       % Empty vector
```

MATLAB Sample

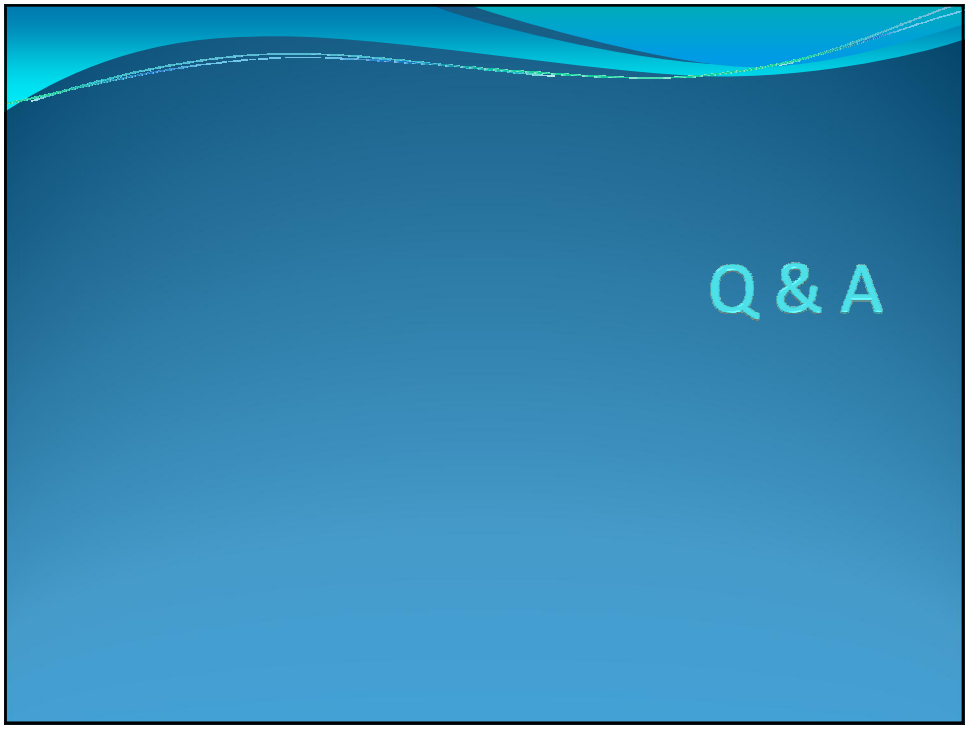
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (B) Creating special matrices: 1ST parameter is ROWS,
%   2ND parameter is COLS

m = zeros(2, 3)              % Creates a 2x3 matrix of zeros
v = ones(1, 3)               % Creates a 1x3 matrix (row vector) of ones
m = eye(3)                   % Identity matrix (3x3)
v = rand(3, 1)               % Randomly filled 3x1 matrix (column
                             % vector); see also randn

                             % But watch out:
m = zeros(3)                 % Creates a 3x3 matrix (!) of zeros
```

MATLAB Sample

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (C) Indexing vectors and matrices.
% Warning: Indices always start at 1 and *NOT* at 0!

v = [1 2 3];
v(3)                        % Access a vector element


m = [1 2 3 4; 5 7 8 8; 9 10 11 12; 13 14 15 16]
m(1, 3)                     % Access a matrix element
                            %       matrix(ROW #, COLUMN #)
m(2, :)                     % Access a whole matrix row (2nd row)
m(:, 1)                     % Access a whole matrix column (1st column)

m(1, 1:3)                   % Access elements 1 through 3 of the 1st row
m(2:3, 2)                   % Access elements 2 through 3 of the
                            %   2nd column
m(2:end, 3)                 % Keyword "end" accesses the remainder of a
                            %   column or row

m = [1 2 3; 4 5 6]
size(m)                     % Returns the size of a matrix
size(m, 1)                  % Number of rows
size(m, 2)                  % Number of columns

m1 = zeros(size(m))         % Create a new matrix with the size of m

who                         % List variables in workspace
whos                        % List variables w/ info about size, type, etc.
```

Reference

- http://www.cs.brown.edu/courses/cs143/MatlabTutorialCode.html@ 20 OCT 2008


Q & A