Armstrong State University
Engineering Studies
MATLAB Marina – Switch-Case Statements Primer

Prerequisites
The Switch-Case Statements Primer assumes knowledge of the MATLAB IDE, MATLAB help, arithmetic operations, built in functions, scripts, variables, arrays, and logic expressions. Material on these topics is covered in the MATLAB Marina Introduction to MATLAB module, MATLAB Marina Variables module, MATLAB Marina Arrays module, and MATLAB Marina Logic Expressions module.

Learning Objectives
1. Be able to use switch-case statements to selectively execute blocks of code.
2. Be able to use switch-case statements to selectively perform operations from a menu.

Terms
logical true, logical false, condition, menu

MATLAB Functions, Keywords, and Operators
switch, case, otherwise, end

Switch-Case Statements
The `switch-case` statement is a conditional structure that is useful for handling multiple cases of the value of a single variable and enables a program to conditionally execute one block of statements from several choices or cases. Generally a `switch-case` statement is preferred over an `if-else` statement if there are four or more cases and if the cases are selected based on a single parameter. The `switch-case` statement has the general form shown in Figure 1a.

```
switch switch expression
   case first case expression
      statements to be executed if first case true
   case second case expression
      statements to be executed if second case true
    :
   otherwise
      statements to be executed if all cases false
end
```

Figure 1a, General Form of switch-case Statement

The result of the MATLAB switch expression must be a scalar, string, or cell arrays of scalars or strings (cell arrays will be covered later in the MATLAB Marina Cell Array module). The result of the switch expression is compared to the result of each case expression until a match is

obtained. The block of statements in the case with the first match is executed. Only one block of statements is executed and the program goes to the statement after the end of the switch-case statement after executing the block of statements. If the result of the switch expression does not match any of the cases, the statements in the otherwise block are executed. The otherwise block is optional and without an otherwise block it is possible for no statements in the switch-case blocks to be executed.

Multiple cases can be handled with the same block of statements using a cell array of case expressions as shown in Figure 1b. All the cases for the block of statements are provided as a comma separated list enclosed in braces. For a case expression with a cell array result, the result of the switch expression needs to match at least one of the results of the case expression.

```
case {case expression 1, case expression 2, …}
   statements to be executed if any of cases in {} true
```

Figure 1b, case Block for Multiple Cases

A common use of the `switch-case` statement is to conditionally execute a block of statements based on a menu choice. The program of Figure 2 computes either the sine, cosine, or tangent of an angle based on a user menu choice.

```
% read in angle and user choice or trig function
angleDegrees = input('Enter angle in degrees: ');
choice = menu('','sine','cosine','tangent');

% compute selected trigonometric function
switch (choice)
    case {1}
        result = sind(angleDegrees);
    case {2}
        result = cosd(angleDegrees);
    case {3}
        result = tand(angleDegrees);
end

disp(result)
```

Figure 2, Program to Compute Sine, Cosine, or Tangent Based on Menu Choice

The `menu` function will return an integer corresponding to the number of the button selected. The variable `choice` will be assigned the value corresponding to the menu button pressed. In the example of Figure 2, the switch-case statement does not contain an `otherwise` block as the only options of the menu are the three cases.

The program of Figure 3a indicates whether a day of the week is a week day or weekend day. The day of the week is read in from the user as a string. Monday through Friday are assumed to be week days and Saturday and Sunday are assumed to be weekend days. Figure 3b shows an alternative implementation with all of the week day cases combined and all of the weekend cases combined.

```
% read in day of week
dayOfWeek = input('Enter day of week: ', 's');

% determine if day is week day or weekend day
switch (dayOfWeek)
    case {'sunday', 'Sunday'}
        disp('weekend day')
    case {'monday', 'Monday'}
        disp('week day');
    case {'tuesday', 'Tuesday'}
        disp('week day');
    case {'wednesday', 'Wednesday'}
        disp('week day');
    case {'thursday', 'Thursday'}
        disp('week day');
    case {'friday', 'Friday'}
        disp('week day');
    case {'saturday', 'Saturday'}
        disp('weekend end')
    otherwise
        disp('Invalid day of week');
end
```

Figure 3a, Week Day or Weekend Day Program

Generally `switch-case` statements should only be used when selecting a case based on an integer, a character, or a string. They should not be used when selecting a case based on a real number result as real numbers are rarely equal and matches for the cases may not occur. The case expression cannot include relational operators (result will be a Boolean which is not a legal result for a case expression). Comparisons based on relational operators should be performed using `if-else` statements.

Only one case of a `switch-case` statement should evaluate to true. Only one condition of an `if-else` statement should evaluate to true also. It is ok for no `switch-case` cases to be matched or no `if-else` conditions to be true.

Be careful when defining new variables within conditional blocks (case, otherwise, if, elseif, else blocks). If the block defining the variable is not executed, that variable will not be available later in the program unless it was previously defined.

```
% define week days and weekend days
weekDays = {'monday', 'Monday', 'tuesday', 'Tuesday', ...
    'wednesday', 'Wednesday', 'thursday', 'Thursday', ...
    'friday', 'Friday'};
weekendDays = {'sunday', 'Sunday', 'saturday', 'Saturday'};

% read in day of week
dayOfWeek = input('Enter day of week: ', 's');

% determine if day is week day or weekend day
switch (dayOfWeek)
    case weekDays
        disp('week day');
    case weekendDays
        disp('weekend day')
    otherwise
        disp('Invalid day of week');
end
```

Figure 3b, Week Day or Weekend Day Program

The program of Figure 4a has a potential runtime error. When the user chooses quit from the menu, the variable `f` will not be created (`t` will exist since it is created before the `switch-case` statement). When the plot function is encountered after a quit choice, a runtime error (Undefined function or variable 'f') occurs since the variable `f` was never defined.

```
% read in plot type from user
choice = menu('','sine', 'cosine', 'quit');

% generate time and trig function vectors
t = (0.0 : 0.05 : 1.0);
switch (choice)
    case 1  % sine
        f = sin(2*pi*t);
    case 2  % cosine
        f = cos(2*pi*t);
    otherwise
        disp('Quit');
end

figure(1);
plot(t,f);
```

Figure 4a, Program to Create Sine or Cosine Plot

The program of Figure 4b is a corrected version of the program of Figure 4a. A Boolean variable `shouldPlot` is initialized to true before the `switch-case` statement and if the quit option is selected the `shouldPlot` flag is set to false. The `figure` and `plot` statements are executed only if `shouldPlot` is true thus avoiding the `plot` statement when the variable `f` is not defined.

```
% read in plot type from user
choice = menu('','sine', 'cosine', 'quit');

% generate time and trig function vectors
t = (0.0 : 0.05 : 1.0);
shouldPlot = true;
switch (choice)
    case 1  % sine
        f = sin(2*pi*t);
    case 2  % cosine
        f = cos(2*pi*t);
    otherwise
        shouldPlot = false;
        disp('Quit');
end

if (shouldPlot)
    figure(1);
    plot(t,f);
end
```

Figure 4b, Program to Create Sine or Cosine Plot

Variables should generally be defined outside of conditional structures such as `switch-case` and `if-else` blocks although they may be modified inside a block of a conditional structure.

Last modified Tuesday, September 09, 2014