MATLAB Marina: Conditional Statements, switch Examples

switch Statement Examples

The program of Figure 1a indicates whether a day of the week is a weekday or weekend day. The day of the week is read in from the user as a string. Monday through Friday are assumed to be weekdays and Saturday and Sunday are assumed to be weekend days.

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

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

Figure 1a. Weekday or Weekend Day Program

Figure 1b shows an alternative implementation with all of the weekday cases combined and all of the weekend cases combined.

```
%% determine if day is weekday or weekend day
clear; clc;
% define weekdays 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
        fprintf('weekday\n');
    case weekendDays
        fprintf('weekend end\n')
    otherwise
        fprintf('Invalid day of week\n');
end
```

Figure 1b. Weekday or Weekend Day Program

Conditional Results, Define Results Beforehand

The program of Figure 2a 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` 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 function arrays
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
        fprintf('Quit\n');
end

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

Figure 2a. Program to Create Sine or Cosine Plot

Variables that would otherwise be created as part of conditional statement should generally be defined before the conditional statement. They can be modified in the conditional statement, but if the conditional statement does not assign something to them or modify them they will exist later in the program.

The program of Figure 2b is a corrected version of the program of Figure 2a. A Boolean variable shouldPlot is initialized to false before the `switch` statement. If f is generated (cosine or sine is chosen), the shouldPlot flag is set to true. The `figure` and `plot` statements are enclosed by a `if` statement and are only executed if shouldPlot is true thus avoiding the `plot` statement when the variable f is not defined.

```
clear; clc; close all;
% read in plot type from user
choice = menu('','sine', 'cosine', 'quit');
shouldPlot = false;

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

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

Figure 2b. Program to Create Sine or Cosine Plot

Last modified Monday, September 28, 2020