Solutions to Homework 5

Help Center

Here are the "official" solutions. Note that there are multiple ways to solve any non-trivial problem, so these are just representative examples.

Problem generationXYZ

```
function gen = generationXYZ(year)
    if year < 1966
        gen = 'O';
    elseif year < 1981
        gen = 'X';
    elseif year < 2000
        gen = 'Y';
    elseif year < 2013
        gen = 'Z';
    else
        gen = 'K';
    end
end
```

Problem generationXYZ (alternative solution)

Using no if-statements

```
function gen = generationXYZ(yr)
    opts = {'O','X','Y','Z','K'};  % Create cell array of options
    idx = 1 + sum(yr >= [1966,1981,2000,2013]); % Calculate index by comparing year to
edge values
    gen = opts{idx};
end
```

Problem letter_grade

```
function G = letter_grade(score)
    if score >= 91
        G = 'A';
    elseif score >= 81
        G = 'B';
    elseif score >= 71
        G = 'C';
    elseif score >= 61
        G = 'D';
    else
        G = 'F';
    end
end
```

Problem sort3

Using no built-in functions

```
function v = sort3(a, b, c)
    if a <= b
        v = [a b];
    else
        v = [b a];
    end
    if c >= v(2)            % a and b in v are ordered. Where to insert c?
        v = [v c];          % at the end
    elseif c <= v(1)
        v = [c v];          % at the beginning
    else
        v = [v(1) c v(2)];  % in the middle
    end
end
```

Problem sort3

Using no if-statements.

```
function v = sort3 (a,b,c)
    v = [a b c];                                    % unordered
    v = [min(v(1),v(3)) v(2) max(v(1),v(3))];   % the 1st and 3rd are in order
    v = [min(v(1),v(2)) max(v(1),v(2)) v(3)];   % move 2nd left if necessary
    v = [v(1) min(v(2),v(3)) max(v(2),v(3))];   % move 2nd right if necessary
end
```

Problem classify

```
function c = classify(x)
    [row, col] = size(x);
    if row == 0 || col == 0         % any dim == 0 -> empty
        c = -1;
    elseif row == 1 && col == 1     % both dim == 1 -> scalar
        c = 0;
    elseif row == 1 || col == 1     % none of the above, but one dim == 1 -> vector
        c = 1;
    else
        c = 2;
    end
```

```
end
```

Problem classify (alternative solution)

```
function c = classify(x)
    mindim = min(size(x));
    maxdim = max(size(x));
    if mindim == 0      % if one dim == 0, it must be empty
        c = -1
    elseif maxdim == 1  % otherwise, both dim == 1 (since max == 1) -> scalar
        c = 0;
    elseif mindim == 1  % otherwise, if the smaller dim == 1 -> vector
        c = 1;
    else
        c = 2;
    end
end
```

Problem classify (alternative solution)

Using no if-statements

```
function y = classify (x)
    d = size(x);
    p = prod(d);                        % multiplies the two dims
    y = -1 +(p>=1) +(p>1) +(min(d)>1)   % each added condition increases the answer by
one
end

% Note that the first two solutions are longer but easier to read and understand than t
his one.
```

Problem older

```
function a = older(y1,m1,d1,y2,m2,d2)
    a = 1;
    if y1 == y2 && m1 == m2 && d1 == d2
        a = 0;
    elseif (y1 > y2) || (y1 == y2 && m1 > m2) || (y1 == y2 && m1 == m2 && d1 > d2)
        a = -1;
    end
end
```

Problem older (alternative solution)

Using no if-statements

```
function a = older(y1,m1,d1,y2,m2,d2)
    a1 = y1 * 366 + m1 * 31 + d1;     % does not have to be exact date in days...
    a2 = y2 * 366 + m2 * 31 + d2;     % it simply makes a1 and a2 comparable
    a = sign(a2 - a1);                % sign() returns -1, 0 or 1, just what is needed
end

% multiplying by 366 or greater is needed because of leap years
```

Problem movies

```
function cando = movies(hr1,min1,durmin1,hr2,min2,durmin2)
    cando = false;
    endtime =   hr1*60 + min1 + durmin1;    % convert times to minutes
    starttime = hr2*60 + min2;
    if endtime <= starttime && endtime + 30 >= starttime  % so we can compare them
        cando = true;
    end
end
```

Problem movies (alternative solution)

Using no if-statement

```
function cando = movies(h1,m1,d1,h2,m2,d2)
    end1 = h1*60 + m1 + d1;
    st2  = h2*60 + m2;
    cando = (end1 <= st2 && end1+30 >= st2);
end
```

Problem sines

```
function [s1 s2 sums] = sines(pts,amp,f1,f2)
    if nargin < 1, pts = 1000;    end
    if nargin < 2, amp = 1;       end
    if nargin < 3, f1 = 100;      end
    if nargin < 4, f2 = f1*1.05;  end
    t = 0 : 2*pi/(pts-1) : 2*pi;
    s1 = amp * sin(f1*t);
    s2 = amp * sin(f2*t);
    sums = s1 + s2;
end

% The sin() function has a full period between 0 and 2*pi.
% To set up the vector t, dividing by (pts-1) is needed
% because n points in a line define (n-1) consecutive segments
% and not n. For example, two points define a single line segment.
% The function call sin(f1*t) will create exactly f1 full periods
% using vector t defined above.
```

Problem moving average

```
function a = moving_average(x)
    persistent xp;
    if isempty(xp)
        xp = x;                 % first time, the buffer simply contains x
    elseif length(xp) < 25
        xp(end+1) = x;          % while fewer than 25 elements, keep adding x to the bu
ffer
    else
        xp = [xp(2:end),x];     % replace first (oldest) element by shifting to the lef
t
    end                         % and inserting x at the end
    a = mean(xp);
end
```

Problem moving average (alternative solution)

Using no if-statement

```
function avg = moving_average (in)
    persistent buffer;
    buffer = [in buffer(1:end-(length(buffer) == 25))];
    avg = mean(buffer);
end

% This is an illustration of a short, but tricky solution. However,
% a longer, but more readable solution is always preferred, therefore,
% the first solution is better!
%
% This one works by realizing that we do not need to check whether the
% buffer is empty or not, since [x buffer] will work either way.
% The tricky part is how the length is handled. While the buffer is
% shorter than 25, buffer(1:end) is used. Once it reaches 25, it turns
% into buffer(1:end-1), exactly what is needed.
```

Published with MATLAB® R2014a

Created Wed 20 May 2015 1:44 PM PDT

Last Modified Tue 26 May 2015 9:11 AM PDT