Problem 1 of Review Questions

```
amt = input('Enter the Income: ');

if (amt > 60000)
    tax = (25/100)*30000+(30/100)*30000+(40/100)*(amt-60000);
elseif (amt > 30000)
    tax = (25/100)*30000+(30/100)*(amt-30000);
else
    tax = (25/100)*(amt);
end

fprintf('The tax to be paid for an income of %d is %d',amt,tax);
```

Problem 2 of Review Questions

```
x = input('Enter the first number: ');
y = input('Enter the second number: ');
z = input('Enter the third number: ');

if (x > y)
    if (x > z)
        fprintf('The maximum of numbers %d, %d, %d is %d',x,y,z,x);
    else
        fprintf('The maximum of numbers %d, %d, %d is %d',x,y,z,z);
    end
else
    if (y > z)
        fprintf('The maximum of numbers %d, %d, %d is %d',x,y,z,y);
    else
        fprintf('The maximum of numbers %d, %d, %d is %d',x,y,z,z);
    end
end
```

Problem 3 of Review Questions

```
% Declare sum,count and set max, min to the first value entered.
sum = 0.0;
count = 0;
num = input('Enter a number(-ve value to terminate): ');
max = num;
min = num;

while( num >= 0 )
        if (num > max)
            max = num;
        end

        if (num < min)
            min = num;
        end
    count = count+1;
    sum = sum+num;
    num = input('Enter another: ');
end

if (count == 0)
    fprintf('The sequence cannot start with a negative value');
else
    mean = sum/(count);
    fprintf('statistics of the %d numbers are\n',count);
    fprintf('sum = %f\nmean = %f\nmaximum = %f\nminimum = %f', ...
            sum,mean,max,min);
end
```

Problem 4 of Review Questions

```
% Assume variables nBig and nSm are given

% Outer loop to iterate from nBig to nSm
while ( nBig >= nSm )

    % Check nBig to see if it is prime
      d=2;  % divisor
      % Iterate until first proper divisor is found
      while (mod(nBig,d) ~= 0);
          d = d + 1;
      end
      if (d == nBig)
          fprintf('%d is a prime\n',nBig);
      else
          fprintf('%d\n',nBig);
      end

    nBig = nBig – 1;
end
```

Problem 5 of Review Questions

Part A

```
function seconds = hms2s( h, m, s );
%H2MS    Converts a time expressed in  hours, minutes, seconds to a time in seconds
%     H2MS( H, M, S ) returns 3600*H + 60*M + S
%     H  = hours
%     M  = minutes
%     S = seconds
seconds = 3600*h+60*m+s;
```

Part B

```
function [h, m, s] = s2hms( seconds );
%S2HMS   Takes an input argument in seconds and returns the hour, minutes,
%and second equivalent
%     [H, M, S] = s2hms( SECONDS )
%     SECONDS = number of input seconds
%     H  = maximum number of hours in SECONDS
%     M  = maximum number of minutes in SECONDS, after max hours are removed
%     S = number of seconds remaining after max hours and max minutes removed

h = floor( seconds/3600 );
m = floor( mod( seconds, 3600 )/60 );
s = mod( mod(seconds, 3600), 60 );
```

Part C Command Window entries and output

```
s  = hms2s( 7, 45, 13 )
s =
      27913
[h, m, s ] = s2hms( 3682 )
h =
    1
m =
    1
s =
   22
```

Problem 6 of Review Questions

```
function [ volume, surfaceArea ] = myCylinder( ratio );
% MYCYLINDER  Calculate volume and surface area of a randomly generated cylinder.
% [V, S] = MYCYLINDER(RATIO)  randomly generates a real number in the range (0, 1)
%        as the diameter d of the circular end of the cylinder.
%        The height of the cylinder is d*ratio.
%        volume = the volume of the cylinder
%        surfaceArea = the surface area of the cylinder

d = rand(1);    % generate diameter in the range (0, 1)
height = d*ratio;
surfaceArea = 2*0.25*pi*d^2 + pi*d*height;
volume = 0.25*pi*d^2*height;
```

Problem 7 of Review Questions

```
% Compute mode from user-entered, non-decreasing, non-negative sequence
% terminated by a negative number.  If multiple modes exist, take the
% first one.

grade = input('Enter grade (negative number to end): ');
  % grade is current grade to process
prevgrade = grade;       % Previous grade entered
freq = 0;                % Frequency of current grade set to
                         %   zero since dummy prevgrade used
                         %   to start the sequence
mode = grade;            % Mode found so far
modeFreq = -1;           % Frequency of mode found so far,
                         %   set to -'ve number to test for
                         %   degenerate case
while (grade>=0)
  % Track frequency
    if (grade==prevgrade)
        freq = freq + 1;
    else
        freq = 1;
    end
  % Is there a new mode?
    if (freq>modeFreq)
        mode = grade;
        modeFreq = freq;
    end
  % Update
    prevgrade = grade;
    grade = input('Enter grade (negative number to end): ');
end

if (modeFreq<0)
    fprintf('Mode not found--no grades entered')
else
    fprintf('Mode is %.0f\n', mode);
end
```