Question 1: (10 points)

Part (a): (4 points)

What will be displayed at the end of each fragment below? If there is an error write the word "error" in the box.

```
w = [2 3];
x = w(w(1))
```

Output

```
x =
     3
```

```
z = [2 3 1];
for k = 1:length(z)
   z(k+1) = z(k);
end
disp(z)
```

Output

```
     2  2  2  2
```

Part (b): (6 points)

What will be printed when the following script is executed?

Script	*Function*	*Output*

`a=2;  b=6;  c=3;`
`d= zoo(c,b);`
`fprintf('a is %d\n', a);`
`fprintf('b is %d\n', b);`
`fprintf('d is %d\n', d);`
	`function a = zoo(b,c)`
	`b= b/c;`
	`a= b;`
	`fprintf('c is %d\n', c);`
		`c is 6`
		`a is 2`
		`b is 6`
		`d is 0.5`

Question 2: (20 points)

Complete each of the functions below according to the specifications. Do *not* use function `find`.

Part (a): (10 points)

```
function h = histData(yr, maj)
% h is the data for drawing a bar graph showing the number of UNDERGRADUATE
% students in each of the 90 majors at Cornell.
% yr and maj are vectors of the same length.  For a valid index k:
%   yr(k)  is the year code of student k.  Possible values are integers
%          in [1..13]; values 1,2,3,4 indicate undergraduate.
%   maj(k) is the major code of student k; possible values are integers
%          in [1..90].
% Assume that the length of yr (and maj) is greater than 1.

h= zeros(1,90);  % h(i) will be the number of undergrads in major i


for k= 1:length(yr)

    if yr(k)<5

        h(maj(k)) = h(maj(k)) + 1;
    end
end


bar(1:90, h)
title('Number of UNDERGRADUATE students in each major')
```

Part (b): (10 points)

```
function s = smoothVec(v)
% Smooth vector v by averaging each "interior" value with its left and right
% neighbors.  s is the smoothed vector and is two components shorter than v.
% Example:  If  v=[-2 5 3 4 8]  then  s=[2 4 5]
% Assume that the length of v is greater than 2.


for k= 2:length(v)-1

    s(k-1) = (v(k-1)+v(k)+v(k+1))/3;
end
```

Question 3: (20 points)

Complete each of the functions below according to the specifications. Do *not* use function `find`.

Part (a): (6 points)

```
function r = randInt(lo, hi)
% r is a uniformly random INTEGER in [lo..hi].
% lo and hi are integers and lo < hi.
```

Grading note:
rand(1) generates a number in (0,1), the *open* interval.

```
r = floor(rand(1)*(hi-lo+1)) + lo;

%An alternative:
%   r = ceil(rand(1)*(hi-lo+1)) + (lo-1);
```

Part (b): (14 points)

```
function ind = myFind(x, v)
% ind is the index of the first occurrence of value x in vector v.
% If x is not found in v then ind is 0.
% x is a scalar.  v is a vector with length greater than 1.
% For full credit your code should be efficient--stop as soon as x is found.
```

```
k= 1;  % Next position to check

while k<=length(v) && v(k)~=x

    k= k+1;
end

if k>length(v)  % x not found

    ind= 0;
else
    ind= k;
end
```

Question 4: (25 points)

Write the *function header* for the function below. The function name is **`checkLengths`**. It has two input parameters, **`a`** and **`b`**, and returns two vectors, **`shortV`** and **`longV`**.

```
function [shortV, longV] = checkLengths(a,b)

% a and b are vectors with length>1; assume their lengths are different.
% shortV is the shorter vector between a and b.
% longV is the longer vector between a and b.

if length(a)<length(b)
    shortV= a;  longV= b;
else
    shortV= b;  longV= a;
end
```

Complete the function below to interleave two vectors. You must use function **`checkLengths`** from Part (a) above as part of your solution. *Do not use vectorized code!*

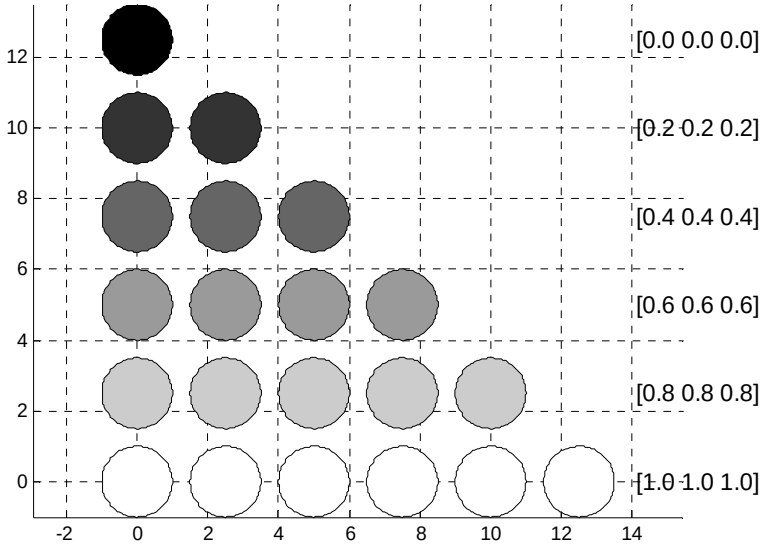
```
function v = interleave(a,b)
% Interleave the values from vectors a and b to form vector v.
% a and b are vectors with length > 1; assume their lengths are different.
% The first value in v comes from the longer vector of a and b.
% The "leftover" values from the longer vector are copied to the end of v.
% For example, if  a=[10 90 30]  and  b=[8 4 5 2 4]
%            then  v=[8 10 4 90 5 30 2 4]
% NO VECTORIZED CODE!

[S, L]= checkLengths(a,b);  % S is shorter than L

nS= length(S);
nL= length(L);
v= zeros(1,nS+nL);   % not necessary to pre-allocate v
iv= 1;   % next index position of vector v

% Interleave up to length of short vector
for k= 1:nS
    v(iv)= L(k);
    iv= iv+1;
    v(iv)= S(k);
    iv= iv+1;
end

% Copy over leftovers from long vector
for k= nS+1:nL
    v(iv)= L(k);
    iv= iv+1;
end
```

Question 5: (25 points)

Complete the function below to draw a set of grayscale disks arranged in a triangle. Read the specifications in the function comment. An *example* figure is shown on the right with **n**=6, **s**=0.5. Assume the availability of function **DrawDisk** and recall that you can specify a color in Matlab using a vector of length 3:

```
colr = [1 1 1]; %white
DrawDisk(5,0,1, colr )
```

draws a white disk with radius 1 centered at (5,0). The grid lines and "color values" are shown on the diagram on the right for your convenience; you do not have to draw them.



```
function grayness(n,s)
% Draw a triangle of disks; there are n disks on each side of the triangle.
% The disk in row 1 is black [0 0 0]; the disks in row n are white [1 1 1];
% the rows in between vary uniformly in grayness.
% The disks have unit radius and are spaced s units apart.
% The center of the lower left disk is at (0,0).

close all;  figure;  axis equal;  hold on

d= 2+s;       % distance from center to center
g= 1/(n-1);   % "distance" between grayness

for r= n:-1:1

    y= (n-r)*d;

    colr= (r-1)*g;  % color for this row

    for c= 1:r

        x= (c-1)*d;

        DrawDisk(x,y,1,colr*ones(1,3))

    end
end


hold off
```

```
% An alternate solution:

% x and y are all coordinates of the centers of the disks
  d= 2+s;  % distance from center to center
  x= linspace(0, (n-1)*d, n);
  y= linspace(0, (n-1)*d, n);  % or just y=x

% colorVec stores all color values
  g= 1/(n-1);  % "distance" between grayness
  colorVec= linspace(0,1,n);

% Draw pattern
  for r= n:-1:1

      yIndex= n-r+1;

      colr= colorVec(r);

      for c= 1:r
          % xIndex= c
          DrawDisk(x(c),y(yIndex),1,colr*ones(1,3))
      end
  end
```