Name: ______________________________ NetID: __________

(Legibly print last name, first name, middle name)

Statement of integrity: *I did not, and will not, violate the rules of academic integrity on this exam.*

____

(Signature)

Circle your lecture time: 9:05 or 11:15

Circle your discussion instructor's name:

	Tuesday	Wednesday
10:10		Helen Sun
11:15		Kun Dong
12:20	Susie Song	Helen Sun
1:25	Susie Song	Kun Dong
2:30	Matthew Davidow	Noam Eshed
3:35	Matthew Davidow	Noam Eshed

Instructions:

- This is a 90-minute, closed-book exam; no calculators are allowed.
- The exam is worth a total of 100 points, so it's about one point per minute!
- Read each problem completely, including any provided code, before starting it.
- Raise your hand if you have any questions.
- Use the back of the pages or ask for additional sheets of paper as necessary.
- Clarity, conciseness, and good programming style count for credit.
- If you supply multiple answers, we will grade only one.
- Use only MATLAB code. No credit for code written in other programming languages.
- Assume there will be no input errors.
- Do not modify given code unless instructed to do so.
- Do not write user-defined functions or subfunctions unless instructed to do so.
- Do not use `switch`, `try`, `catch`, `break`, `continue`, or `return` statements.
- Do not use built-in functions that have not been discussed in the course.
- You may find the following MATLAB predefined functions useful:
  `abs`, `sqrt`, `rem`, `floor`, `ceil`, `round`, `rand`, `zeros`, `ones`, `linspace`, `length`, `input`, `fprintf`, `disp`, `bar`

Examples:
`rem(5,2)` → 1, the remainder of 5 divided by 2
`rand` → a random real value in the open interval (0,1)
`floor(6.9)`, `floor(6)` → 6, rounds down to the nearest integer
`ceil(8.1)`, `ceil(9)` → 9, rounds up to the nearest integer
`zeros(1,4)` → 1 row 4 columns of zeros
`length([2 4 8])` → 3, length of a vector

Question 1: (15 points)

(a) In each of the following cases, is it better to use a `for`-loop or a `while`-loop? Circle only one choice (`for` or `while`) for each case. By "better," first consider run-time efficiency and then compactness of the code. Recall that the `break` keyword is not allowed.

`for` / `while` **Case 1** Calculate the first 100 Fibonacci numbers.

`for` / `while` **Case 2** Prompt the user to input a value until a negative value is entered.

`for` / `while` **Case 3** Find the smallest value in a vector.

`for` / `while` **Case 4** Find the first instance of the value 5 in a vector of integers.

(b) Write one expression on the blank so that `b` is a uniformly random real value generated in the interval (-14.1,5). The only built-in function allowed is `rand`.

`b= ______________________________________________________________________`

(c) Write one expression on the blank so that scalar `c` is randomly chosen from the set $[0, 2, 4, \ldots, 100]$ with equal likelihood. (Note that `c` is even.) The only built-in functions allowed are `rand`, `floor`, and `ceil`.

`c= ______________________________________________________________________`

(d) What will be printed when the following script is executed? Use the specified print format.

Script	*Function*
`x = 3;`	`function [a,b] = gobble(y,x)`
`y = 5;`	
`[x,z] = gobble(x,y);`	`a = y - x;`
`fprintf('x is %d\n', x)`	`b = x + 10;`
`fprintf('y is %d\n', y)`	`z = 20;`
`fprintf('z is %d\n', z)`	`fprintf('a is %d\n', a)`

Output:

Exam score:

Q1:	(15)	______	____
Q2:	(10)	______	____
Q3:	(20)	______	____
Q4:	(30)	______	____
Q5:	(25)	______	____
Total:	(100)	______	____

Question 2: (10 points)

A leap year is a year that is divisible by 4 with one exception: years divisible by 100 are not leap years unless they are also divisible by 400. For example, the year 2016 was a leap year, the year 1600 was a leap year, but the year 1700 was not a leap year.

Complete the script below to determine whether the given variable `y` corresponds to a leap year. The script should display the word "leap" if `y` is a leap year; otherwise "not leap" should be displayed.

```
y= input('Enter a year:  ');  % Assume y is an integer > 0
% Determine whether y is a leap year
```

Question 3: (20 points)

Implement the following function as specified:

```
function idx = whereGreater(v, w)
% Find the indices of the values in vector w that are greater than all the values in v.
% v, w:  each is a non-empty vector of type double values.
% idx:   a vector of the indices of w where w is strictly greater than all the values
%        in v; idx may be empty.
% Example: If v is [2 3 0] and w is [5 3 -6 9 2], then idx is [1 4] because w(1) and w(4)
%          are greater than all the values in v.
% The only built-in function allowed is length.
% Be run-time efficient for full credit.
```

Question 4: (30 points)

(a) Implement the following function as specified:

```
function [pFinal, hFinal] = doubleGame(pStart, hStart)
% Simulate the "double game," a betting game between a player and a host.
% pStart: a positive number of chips with which the player starts the game.
% hStart: a positive number of chips with which the host starts the game.
% A game consists of 1 or more rounds.  The betting starts at 1 chip.  In each round of
%   the game Player flips a coin:  heads means Player wins the bet from Host and the
%   game ends (no more rounds); tails means Player loses the bet to Host but can start
%   another round that doubles the bet if Player and Host each has enough chips for the
%   bet.  The game ends when Player wins a bet or when Player or Host does not have
%   enough chips for the bet.
% pFinal, hFinal: the number of chips that Player and Host have, respectively, at the
%   end of the game.
```

Question 4, continued.

(b) Assume that function `doubleGame` from Part (a) has been correctly implemented; make effective use of it in order to implement the following function as specified:

```
function [count, playerAve] = manyDoubleGames(n,pStart,hStart)
% Simulate the "double game" n times, each time with Player starting with pStart chips
%   and Host starting with hStart chips.  n, pStart, and hStart are each a positive
%   integer.
% count is a vector of appropriate length such that count(k) is the number of times
%   that Player ends the game with k-1 chips.  I.e., count(1) is the number of times
%   that Player ends the game with 0 chip, count(2) is the number of times that Player
%   ends the game with 1 chip, ..., etc.
% playerAve is the average number of chips with which Player ends the game.
% The only built-in function allowed is zeros.
% Be run-time efficient for full credit.
```

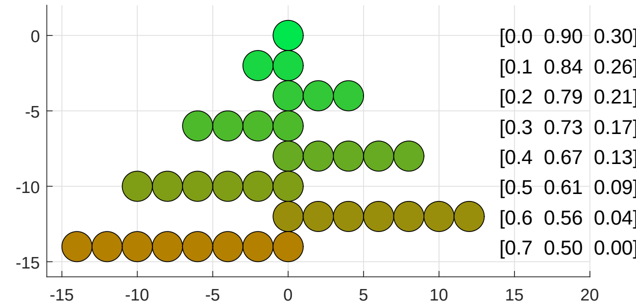
Question 5: (25 points)

Complete the function below as specified. *Do not* use any built-in functions other than `rem`, `length` and `zeros`. The diagram on the right shows an example graphic produced by the following statements:

```
green=[0 .9 .3]; brown=[.7 .5 0];
treePlot(0, 0, 8, 1, green, brown)
```

Assume the availability of the function `DrawDisk`. For example, the command

```
DrawDisk(3, 2, .5, [1 0 0])
```



draws a red disk of radius 0.5 centered at (3,2). Your code draws only the disks. The grid lines and the rgb values are shown for your convenience; do not draw them.

```
function treePlot(xc, yc, n, r, green, brown)
% Draw a "tree" where the nth row has n leaves.  Each leaf is a disk of radius r.
% The top leaf of the tree is centered at xc, yc.
% The remaining rows of leaves grow alternately to the left and to the right.
% The top leaf has the color green; the bottom row of leaves has the color brown;
%   the rows in between vary uniformly in color (linearly interpolated).

close all; figure; axis equal; hold on
```

```
hold off
```

Hint: DECOMPOSE! First work on drawing the disks at the correct locations all in one color; then revise your code to deal with the color interpolation.