Name: ___________________________ NetID: __________
(Legibly print Surname, 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:

<table>
<thead>
<tr>
<th></th>
<th>Tuesday</th>
<th>Wednesday</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:10</td>
<td>Sara Venkatraman</td>
<td></td>
</tr>
<tr>
<td>11:15</td>
<td>Ethan Keller</td>
<td></td>
</tr>
<tr>
<td>12:20</td>
<td>Ethan Keller</td>
<td>Sara Venkatraman</td>
</tr>
<tr>
<td>1:25</td>
<td>Pooja Nath</td>
<td>Rachael Van Pelt</td>
</tr>
<tr>
<td>2:30</td>
<td>Rohit Biswas</td>
<td>Pooja Nath</td>
</tr>
<tr>
<td>3:35</td>
<td>Rohit Biswas</td>
<td>Wayne Uy</td>
</tr>
</tbody>
</table>

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 backs of 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.
• Write user-defined functions only if asked to do so.
• Do not write subfunctions.
• Do not use switch, try, catch, break, or continue statements.
• You may find the following MATLAB predefined functions useful:
  abs, sqrt, rem, mod, floor, ceil, rand, zeros, ones, length, size, input, fprintf, disp, bar,
  uint8, double, char, strcmp, str2double, fopen, fclose, fgetl, feof, isempty, cell, struct

Examples:
floor(6.9), floor(6) \to 6, rounds down to the nearest integer
zeros(2,4) \to a 2-by-4 matrix of zeros, type double
bar(1:4, [3 0 9 6]) \to a bar graph with 4 bars: 1st bar has height 3, 2nd bar has height 0, ...
cell(3,2) \to a 3-by-2 cell array, each cell is the empty numeric vector []
strcmp(’cat’, ’Cat’) \to 0, the two strings are not identical
str2double(’-2.6’) \to -2.6, a type double scalar
struct(’a’, 1, ’b’, 0) \to a structure with 2 fields: a has value 1, b has value 0
Question 1: (20 points)

(a) What is the output from executing the following script? If the program results in an error, instead of the program output write “error 1” if the error occurs in Statement 1 or “error 2” if the error occurs in Statement 2.

```
tf = 'cat' == 'dog'; % Statement 1
disp(tf) % Statement 2
```

Output:

(b) Assuming that a user enters input correctly, complete the script below by writing an appropriate expression on the blank. *No built-in function is allowed.*

```
ch = input('Type one character enclosed in single quotes: '); % Assume that ch stores a single character.
if _________________________________________________ % NO built-in function allowed
disp('Variable ch stores a lower case letter')
else
disp('Variable ch does not store a lower case letter')
end
```

(c) Assign to variable d the positive difference between uint8 values f and g. d must have the type uint8. Use two different approaches as specified below. Do both parts, do not just choose one.

<table>
<thead>
<tr>
<th>Approach 1: Do not use any built-in functions</th>
<th>Approach 2: Use uint8, double, abs appropriately</th>
</tr>
</thead>
<tbody>
<tr>
<td>f = uint8(ceil(rand*200));</td>
<td>f = uint8(ceil(rand*200));</td>
</tr>
<tr>
<td>g = uint8(ceil(rand*200));</td>
<td>g = uint8(ceil(rand*200));</td>
</tr>
<tr>
<td>% Compute the positive difference d</td>
<td>% Compute the positive difference d</td>
</tr>
</tbody>
</table>

(d) Suppose the Workspace is empty and the statements below are to be typed into the Command Window. On each blank below, write the word “ok” if the statement on the left would execute without error; otherwise write the word “no”.

```
A.netID = 'bda45';
A.major = 'CS';
B.netID = 'zaq8';
C.major = B.netID; %__________
Student(1) = A; %__________
Student(2) = B; %__________
```

Exam score:

<table>
<thead>
<tr>
<th>Q1: (20)</th>
<th>Q2: (18)</th>
<th>Q3: (17)</th>
<th>Q4: (25)</th>
<th>Q5: (20)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total: (100)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Question 2: (18 points)

In a tennis club with \( n \) players where \( n > 2 \), each player has played at least one match against each of the other players. An \( n \)-by-\( n \) matrix \( H \) stores the “head to head” records of these club players such that \( H(i,j) \) is the number of times that player \( i \) has beaten player \( j \).

Implement the following function to convert the head to head records to “winning fractions.” For example, if player \( x \) has beaten player \( y \) 4 times and player \( y \) has beaten player \( x \) 1 time, then the player \( x \) vs. player \( y \) winning fraction is 0.8 (based on the 5 times that they played against one another) while the player \( y \) vs. player \( x \) winning fraction is 0.2.

```matlab
function W = tennisStats(H)
% Determine the winning fractions of tennis players given the head to head records.
% H is a square matrix; H(i,j) is the number of times player i has beaten player j.
% Each value on the main diagonal of H is zero because a player does not play
% against herself/himself.
% W is a 2-d cell array the same size as H and stores the winning fractions:
% - Off the main diagonal, the cell in row i column j stores the winning fraction
%   of player i vs. player j.
% - On the main diagonal, each cell stores the string 'NA' (to indicate that no
%   winning fraction can be calculated for a player playing against
%   herself/himself.)
```

\[ \text{function } W = \text{tennisStats}(H) \]
\[ \% \text{Determine the winning fractions of tennis players given the head to head records.} \]
\[ \% \text{H is a square matrix; H(i,j) is the number of times player i has beaten player j.} \]
\[ \% \text{Each value on the main diagonal of H is zero because a player does not play} \]
\[ \% \text{against herself/himself.} \]
\[ \% \text{W is a 2-d cell array the same size as H and stores the winning fractions:} \]
\[ \% \quad \text{- Off the main diagonal, the cell in row i column j stores the winning fraction} \]
\[ \% \quad \text{of player i vs. player j.} \]
\[ \% \quad \text{- On the main diagonal, each cell stores the string 'NA' (to indicate that no} \]
\[ \% \quad \text{winning fraction can be calculated for a player playing against} \]
\[ \% \quad \text{herself/himself.)} \]
Question 3: (17 points)

Implement the following function as specified:

```matlab
function Q = myFilter(P,d)

% P is a 3-d array of type uint8 corresponding to a color jpeg image.
% Q is a 3-d array of type uint8 the same size as P. Q is a modified version of P.
% d is a positive integer. d<nr/2 and d<nc/2 where nr and nc are the number of rows
% of pixels and the number of columns of pixels, respectively, in P. The d pixels
% from each edge of the image will NOT be modified. Each of the remaining
% pixels, called interior pixels, will be modified as follows:
% Each interior pixel (i,j) has four diagonal neighbors--northwest, northeast,
% southeast, and southwest. In each layer, replace the color intensity value of
% pixel (i,j) with the average color intensity value of the four diagonal neighbors.
% Built-in functions sum and mean are NOT allowed.
```
Question 4: (25 points)

A 3-city cycling race is to be organized given the data of \( n \) cities, \( n > 3 \). Each struct in the length \( n \) struct array \( S \) stores the data of an individual city and has these fields:

- **cname**: the name of the city, a string
- **budget**: the amount of money that the city is willing to contribute to the race, a positive scalar

An \( n \)-by-\( n \) matrix \( D \) stores the distances between cities: \( D(i,j) \) is the distance between cities \( i \) and \( j \) in miles. \( D \) is symmetric, so \( D(i,j) = D(j,i) \).

In a 3-city cycling race, the cyclists start in one city, ride to a second city, ride to a third city, and then ride directly to the starting city. Given our symmetric matrix \( D \), for a particular combination of three cities the race distance is the same regardless of the itinerary (the order of the three cities in the race). Elevation is not a concern in this race!

Given the struct array \( S \) and distance matrix \( D \), consider all possible 3-city combinations that give a race distance of at least 100 miles and no more than 200 miles. Among those combinations find the one that has the largest total budget. (Assume that only one combination has this largest total budget value.) Store the data of this combination in a 1-d cell array \( R \) of length 4 where the first 3 cells store the names of the three cities and the 4th cell stores the total budget. If no 3-city combination meets the distance criteria, then \( R \) is an empty cell array.

For full credit, your code should be efficient. Built-in functions \texttt{max} and \texttt{min} are not allowed.

\%
Assume that struct array \( S \) and matrix \( D \) are given and are as described above.
\%
Write your code below. Functions max and min are NOT allowed.

Save this question for \underline{final exam review}.

\underline{NOT} for Prelim 2.
Question 5: (20 points)

A MATLAB script or function file contains plain text (ASCII) characters only. Such a file contains comments as well as programming statements. On a line in the file that contains a comment, the comment begins with the character ‘%’ and continues until the end of the line. A comment can begin anywhere on a line. We will consider only MATLAB files in which the ‘%’ character is used solely for indicating the beginning of a comment. I.e., the file does not contain any statement (e.g., fprintf) that may uses the the ‘%’ character other than for indicating the beginning of a comment.

Complete the following function as specified. Note the example block of code given at the bottom of the page to remind you how to use the file handling functions.

```matlab
function co = extractComments(mf)
    % Store in cell array co all the comments in a Matlab script or function file.
    % mf: a string; it is the name of a Matlab script or function file.
    % co: 1-d cell array of strings; each cell stores one comment line, including the
    %     comment symbol '%', from the file named by mf. The number of cells in co is
    %     the number of comment lines in the file. If the file does not contain any
    %     comments, co is an empty cell array.
    % Built-in functions find and strfind are NOT allowed.

    % A reminder on how to read
    % every line of a file 'foo.m'
   (fid = fopen('foo.m','r'));
    while ~feof(fid)
        s = fgetl(fid);
    end
    fclose(fid);
```