CS1112 Fall 2018 Prelim 2 Nov 8 7:30–9:00pm

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:

	Tuesday	Wednesday
10:10		Chinasa Okolo
11:15		Chinasa Okolo
12:20	Susie Song	Chinasa Okolo
1:25	Susie Song	Ian Delbridge
2:30	Matthew Davidow	Ian Delbridge
3:35	Ian Delbridge	Matthew Davidow

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

  Examples:
  `rand(1,4)` → a length 4 row vector of random type `double` values, each in the interval (0,1)
  `zeros(2,4,3)` → a 2 rows by 4 columns by 3 layers array of zeros, type `double`
  `cell(3,2)` → a 3-by-2 cell array, each cell is the empty vector `[]`
  `strcmp('cat','Cat')` → 0, the two strings are not identical
  `struct('a',1,'b',0)` → a structure with 2 fields: `a` has value 1, `b` has value 0
  `uint8(4.7)` → the integer (type `uint8`) value 5

Question 1: (15 points)

(a) What is the output from executing the following script? If the program doesn't terminate or if there will be an error during execution, write the word "error" instead of the output.

```
s = 'almostweekend';
n = 13; % no. of characters in s
j = 1;
k = n-j+1;
while  j<k
    s(k)= s(j);
    j = j+1;
    k = n-j+1;
end
disp(s)
```

Solution:

```
almostwtsomla
```

(b) Suppose variables `g` and `h` each stores a `uint8` scalar. Complete the statement below so that `d` stores the absolute value of the difference between `g` and `h`. `d` should be of type `uint8`.

```
d =   uint8(     abs(  double(g) - double(h)  )     )

d =                          (g-h) + (h-g)
```

(c) If executed, the script below displays the phrases "Event A", "Event B", and "Event C". Modify the script such that if executed, it will display "Event A" with 10% chance (probability 0.1), "Event B" with 30% chance, or "Event C" with 60% chance. Only one event should be displayed.

```
    r= rand;  % rand ints ok

    if  r < 0.1            % any way to get 1 in 10 chance just for A

          disp('Event A')

    elseif  r < 0.1 + 0.3  % any way to get 3 in 10 chance excluding previous branch
                           % e.g., also correct:  r > 0.7

          disp('Event B')

    else

          disp('Event C')

    end
```

Suppose B is an n rows by 3 columns cell array, where $n > 10$. Each row of B stores information on a course: the first cell is the department abbreviation as a `char` row vector, e.g., `'CS'`; the second cell is the course number as a `char` row vector, e.g., `'1112'`; the third cell is the number of students enrolled in the course as a type `double` scalar, e.g., 230.

(a) Given cell array B as described above, write a code fragment to store in a length 10 cell array D the complete name of the first 10 courses represented in B. The complete name of a course is the department abbreviation followed by the course number, e.g., `'CS1112'`. Each cell of D is one course name, a `char` row vector.

Example solution:

```
for r= 1:10
    D{r}= [B{r,1} B{r,2}];
end
```

(b) Given cell array B as described above, write a code fragment to create a 1-d struct array F for storing the information of the courses with at least 200 students enrolled. Each struct in F has a field `'dept'` storing the department abbreviation (`char` row vector), a field `'id'` storing the course number as a `char` row vector, and a field `'count'` storing the number of students enrolled (type `double` scalar). The length of F is the number of courses with at least 200 students; assume that there is at least one such course.

Example solution:

```
[nr, nc]= size(B);
k= 0;
for r= 1:nr
    if  B{r,3} >= 200
        k= k + 1;
        F(k)= struct('dept', B{r,1}, 'id', B{r,2}, 'count', B{r,3});
        % Also correct:
        %    F(k).dept= B{r,1}; F(k).id= B{r,2}; F(k).count= B{r,3};
    end
end
```

Question 3: (25 points)

(a) Implement the following function as specified.

```
function Z = rgb2blackwhite(X,t)
% Convert color image data to black-white image data.
% X: 3-d uint8 array storing color image data; X is not empty.
% Z: 2-d uint8 array where each element is either 0 (black) or 255 (white);
%    Z has the same number of rows and columns as X.
% t: uint8 vector of length 3; the 3 values are the thresholds for red,
%    green, and blue, respectively.
% For each pixel, if at least two of its red, green, and blue values are
%   strictly less than their respective thresholds, then the pixel becomes
%   black; otherwise the pixel becomes white.
% Do NOT use vectorized code.
```

Solution:

```
[nr, nc, np]= size(X);
Z= uint8(zeros(nr,nc));  % Not necessary but ok
for r= 1:nr
    for c= 1:nc
        under= 0;  % number of colors under threshold
        for p= 1:np
            if X(r,c,p) < t(p)
                under= under + 1;
            end
        end
        if under <= 1
            Z(r,c)= uint8(255);  % if Z initialized to uint8 outside loop,
                                 % then no need to cast 255 to uint8.
        end
    end
end
% If all previous ops in type double, then must cast Z to uint8 at the end.
```

Part (b) appears on the next page.

Question 3, continued

(b) In image processing, "dilation" refers to expanding or thickening the features in a black-and-white image. Suppose the elements in a 2-d array have only the values 0 and 255, the dilation operation will change a 0 to 255 if the 0 is immediately north, south, east, or west of an element with the value 255. For example,

$$\text{dilating} \begin{bmatrix} 0 & 0 & 255 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 255 & 255 \\ 255 & 0 & 0 & 255 & 0 & 255 \end{bmatrix} \text{gives} \begin{bmatrix} 0 & 255 & 255 & 255 & 0 & 0 \\ 0 & 0 & 255 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 255 & 255 \\ 255 & 0 & 0 & 255 & 255 & 255 \\ 255 & 255 & 255 & 255 & 255 & 255 \end{bmatrix}.$$

Implement the following function as specified.

```
function D = dilate(Z)
% Dilate a black-white image whose data is stored in 2-d uint8 array Z.
%   Each value in Z is either 0 or 255.  Z is not empty.
% D is a 2-d type double array storing the result of dilating Z.  D has the
%   same number of rows and columns as Z.
% Do NOT use vectorized code.
```

Solution:

```
 [nr,nc]= size(Z);
 % Example solution 1: if any neighbor is 255; turn itself into 255
 for r= 1:nr
     for c= 1:nc
         % if any N,E,S,W neighbor of pixel (r,c) is white, make pixel (r,c)
         % white
         if r>1 && Z(r-1,c)==255 || r<nr && Z(r+1,c)==255 || ...
            c>1 && Z(r,c-1)==255 || c<nc && Z(r,c+1)==255
             D(r,c)= 255;             % type double
         else
             D(r,c)= double(Z(r,c));  % assigning to 0 is wrong
         end
     end
 end
 % If any previous assignment in uint8, then must cast D to double at the end.

% Example solution 2: if itself is 255, turn its N,E,S,W neighbors to 255
[nr,nc]= size(Z);
D= zeros(nr,nc);  % this is ok, but not  D=Z  b/c vectorized code
for r= 1:nr
    for c= 1:nc
        if Z(r,c)==255
            D(r,c)= Z(r,c);
            if r>1
                D(r-1,c)= 255;
            end
            if r<nr
                D(r+1,c)= 255;
            end
            if c>1
                D(r,c-1)= 255;
            end
            if c<nc
                D(r,c+1)= 255;
            end
        end
    end
end
% If D was initialized as uint8, then must cast D to double at the end.
```

Question 4: (20 points)

Implement the following function as specified. Be efficient for full credit.

```
function [minDis, delivIdx] = deliverTwo(P)
% Choose from a set of delivery locations two that result in the shortest travel distance.
%   A delivery van departs a warehouse at x-y coordinates (0,0), travels to one location,
%   travels to another location, then returns to the warehouse. Estimate the travel
%   distance as the straight-line distance between two points. For example, if the
%   delivery is to be made to locations 2 and 5, the two possible paths
%       warehouse --> location 2 --> location 5 --> warehouse
%       warehouse --> location 5 --> location 2 --> warehouse
%   have the same travel distance.
% P: 1-d array of point structs. Each point struct has the fields 'x' and 'y' storing
%   the x- and y-coordinates, respectively, of a delivery location. P contains at least
%   two structs, no two locations are the same, and none is the location of the warehouse.
% minDis: the minimum total distance (type double scalar) traveled by the van to make
%   two deliveries and return
% delivIdx: vector of length 2 storing the indices of the two delivery locations in P
%   that result in the shortest travel distance. The order of the two values in delivIdx
%   does not matter.
% Be efficient for full credit.
```

Example solution:

```
n= length(P);
minDis= inf;
delivInx= zeros(1,2);  % not necessary to initialize

for j= 1:n-1  % letting j go to n is ok

    dis1= sqrt(P(j).x^2 + P(j).y^2);  % (0,0) to P(j)

    for k= j+1:n
        dis2= sqrt( (P(k).x-P(j).x)^2 + ...
                    (P(k).y-P(j).y)^2 );  % P(j) to P(k)

        dis3= sqrt(P(k).x^2 + P(k).y^2);  % P(k) to (0,0)

        totalDis= dis1 + dis2 + dis3;

        if totalDis < minDis

            minDis= totalDis;
            delivIdx= [j k];
        end
    end
end
```

Recall that the distance between two points (x_a, y_a) and (x_b, y_b) is $\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$

Question 5: (25 points)

Implement the following function as specified.

```
function counts = wordBag(text, vocab)
% Determine how many times each word of a vocabulary appears in a sentence.
% text:  The sentence. A period terminated char row vector containing words in lower-
%   case. Words are separated by a single space and there is no punctuation mark other
%   than the period at the end.
% vocab:  A 1-d cell array of distinct words (char row vectors) representing a vocabulary
% counts:  A type double vector the same length as vocab storing the number of times
%   each word of vocab appears in text.
% Example:  Suppose    text= 'the red stone and red walls.'
%           and        vocab= {'red', 'blue', 'wall', 'stone'}
%           Then       wordBag(text, vocab)
%           returns    [2 0 0 1]
%   because the first word of vocab appears twice in text, the second word of vocab
%   appears 0 times in text, ..., and so forth.
% The only built-in functions allowed are zeros, cell, length, size, strcmp
```

Example Solution 1:

```
nv= length(vocab);
counts= zeros(1,nv);
curWord= '';
for k= 1:length(text)
    if  text(k)~=' ' && text(k)~='.'  % a char of current word
        curWord= [curWord text(k)];
    else  % a word has ended and is in curWord
        % Search for curWord in vocab
        j= 1;  % note re-init here
        while  j<=nv && ~strcmp(vocab{j}, curWord)
            j= j + 1;
        end
        if j<=nv
            counts(j)= counts(j) + 1;
        end
        curWord= '';  % must reset curWord to length 0
    end
end
```

Example Solution 2:

```
nv= length(vocab);
counts= zeros(1,nv);
wordStart= 1;  % starting index of a word
wordIdx= 0;
% Find all words and store them in cell array allWords
for k= 1:length(text)
    end
    if  text(k)==’ ’  ||  k==length(text)
        % a word has ended
        curWord= text(wordStart:k-1);
        wordIdx= wordIdx + 1;
        allWords{wordIdx}= curWord;
        wordStart= k+1;
    end
end

% Search for each word of allWords in vocab
for k= 1:wordIdx
    j= 1;  % note re-init here
    while  j<=nv && ~strcmp(vocab{j}, allWords{k})
        j= j + 1;
    end
    if j<=nv
        counts(j)= counts(j) + 1;
    end
end
```