Name: SOLUTION

Statement of integrity:

It is a violation of the Code of Academic Integrity to look at any exam other than your own, to look at any reference material outside of this exam packet, to communicate with anyone other than the exam proctors, or to otherwise give or receive any unauthorized help during the exam.

Academic Integrity is expected of all students of Cornell University at all times. **By submitting this exam, you declare that you will not give, use, or receive unauthorized aid in this examination.**

Circle your discussion section:

	Wednesday	Thursday
9:40 AM	Aravind Suresh Babu	Aravind Suresh Babu
11:25 AM	Subham Sahoo	Subham Sahoo
1:00 PM	Claire Liang	-
2:45 PM	Claire Liang	-

Instructions:

- Check that this packet has 8 double-sided sheets.
- 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.
- Do not modify any *given* code unless asked to do so.
- Raise your hand if you have any questions.
- Use the back of the pages if you need additional space.
- Clarity, conciseness, and good programming style count for credit.
- Indicate your final answer. If you supply multiple answers, you may receive a *zero* on that question.
- Use only MATLAB code. No credit for code written in other programming languages.
- Assume there will be no input errors.
- Write user-defined functions and subfunctions only if asked 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`, `rand`, `zeros`, `ones`, `linspace`, `length`, `input`, `fprintf`, `disp`

Examples:
`rem(5,2)` → 1, the remainder of 5 divided by 2
`rand()` → a random real value in the interval (0,1)
`abs(-3)` → 3, absolute value
`floor(6.9)`, `floor(6)` → 6, rounds down to the nearest integer
`ceil(8.1)`, `ceil(9)` → 9, rounds up to the nearest integer
`length([2 4 8])` → 3, length of a vector
`zeros(1,4)` → 1 row 4 columns of zeros
`linspace(3,5,10)` → a vector of 10 real numbers evenly distributed in the interval [3,5]

Question 1 (9 points)

(1.1) What will be printed when the following code is run? If an error would occur during execution, write all the output that would be produced, if any, before the error would occur, followed by the word "error".

```
for a = 1:4
    x = a + 1;
    for b = x:4
        disp(b)
    end
end
```

Solution:

```
2
3
4
3
4
4
```

(1.2) What will be printed when the following script is executed?

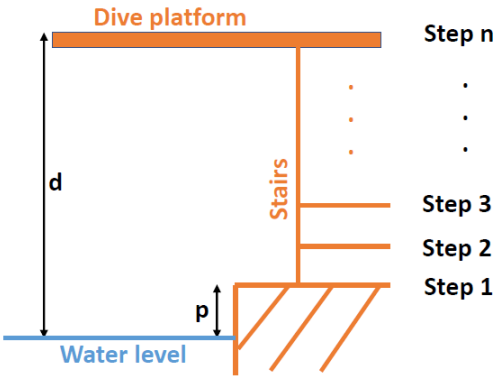
Script	*Function*
`a = 1;`	`function b = woo(b,c)`
`b = 4;`	
`c = 6;`	`b = b - c;`
`d = woo(c,b);`	
`fprintf('a is %d\n', a)`	`a = b;`
`fprintf('b is %d\n', b)`	
`fprintf('d is %d\n', d)`	`fprintf('c is %d\n', c)`

Solution:

```
c is 4
a is 1
b is 4
d is 2
```

Question 2

2.1 (9 points) A dive platform is to be built at a swimming pool. The pool edge (step 1) is p feet above the water level. The top of the dive platform (step n) is d feet above the water level. The distance between the tops of the steps are the same. Complete the script below to print the elevation of the top of each step. You can add code anywhere but do not cross out any given code. Do not use any built-in functions.




```
p= input('Distance in feet from water level to the top edge of the pool: ');
d= input('Distance in feet from water level to the top of the dive platform: ');
n= input('Enter the number of steps, an integer greater than 2: ');




for k =




    fprintf('The top of step %d is %f feet above water level\n', k, elev)




end
```

```
% Example sol: pre-compute height per step, then each iteration do multiples of it
height= (d-p)/(n-1);  % height of each step
for k= 1:n
    elev = (k-1)*height + p;
    fprintf('The top of step %d is %f feet above water level\n', k, elev)
end
```

```
% Example sol: in each iteration compute fractional contribution then combine
for k= 1:n
    frac= (k-1)/(n-1);  % fraction of contribution of bottom edge
    elev= frac*d + (1-frac)*p;
    fprintf('The top of step %d is %f feet above water level\n', k, elev)
end
```

2.2 (11 points) Given a vector, the *two neighbors* of an element are the immediately preceding and following elements. The first and last elements of a vector do not have two neighbors; the interior elements do. We define the "two-neighbor sums" of a vector v to be the vector storing the sums of the two neighbors of all the interior elements of v. For example, vector [2 5 -6 4 -1] has "two-neighbor sums" [-4 9 -7]. Implement the following function as specified. Do not use any built-in functions other than `length` and `zeros`.

```
function [svec, sm]  = twoNeighborSums(v)
% Return the "2-neighbor sums" vector of v and the smallest 2-neighbor sum.
% v is a numeric vector with a length greater than 2.
% svec is the 2-neighbor sums vector.
% sm is the smallest 2-neighbor sum.
% Do not use any built-in functions other than length, zeros.
```

```
% Example solution

n= length(v);
svec= zeros(1,n-2);  % initialization not necessary

sm= inf;
for k= 2:n-1
    neighborSum= v(k-1) + v(k+1);
    svec(k-1)= neighborSum;
    if  neighborSum < sm
        sm= neighborSum;
    end
end
```

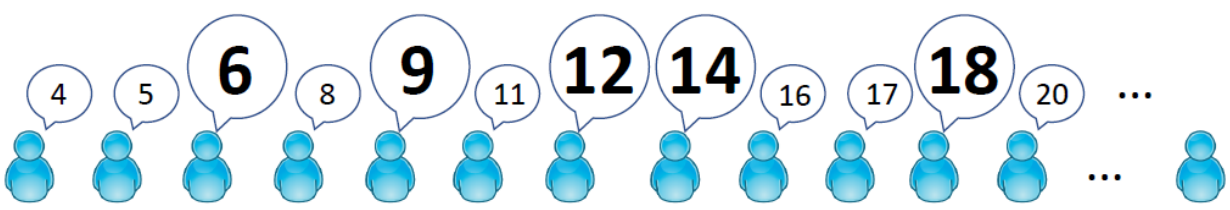
Question 3 (20 points)

Implement the following function as specified.

```
function [spoken, shouted] = countingGame(n, x)
% n people play a game in which they take turns saying a number, starting
%   at x and adding one each time.  However, when they hit a multiple of 3
%   or 7, they must shout the number instead of speaking normally.  After a
%   number has been shouted, the next person skips a number.  The game ends
%   after each person has said--either spoken or shouted--a number.
% n: the number of people, a positive integer greater than 1
% x: the number to start with, an integer
% spoken: a vector of all the numbers spoken, in the order they were spoken
% shouted: a vector of all the numbers shouted, in the order they were shouted
```

```
% Example solution

shouted= [];
spoken= [];
for k= 1:n
    if  rem(x,3)==0  ||  rem(x,7)==0
        shouted= [shouted x];
        x= x + 2;
    else
        spoken= [spoken x];
        x= x + 1;
    end
end
```



Question 4

4.1 (6 points) Implement the following function as specified. The only functions that you may use are `rand`, `round`, `floor`, `ceil`, and `zeros`.

```
function v = randInts(n, lo, hi)
% Return v, a vector of n randomly generated integers where each is
% equally likely to be any integer in the set lo..hi
% n: the number of random values to generate, also the length of v. n>0.
% lo: an integer value
% hi: an integer value greater than lo
```

```
% Example solution 1
v= floor( rand(1,n)*(hi-lo+1) ) + lo;
```

```
% Example solution 2
v=  ceil( rand(1,n)*(hi-lo+1) ) + lo - 1;
```

4.2 (24 points) For Question 4.2, assume that function `randInts` above is correctly implemented and accessible. Assume also the availability of a function `topTwo`, which has the following specification:

```
function [largest, secondLargest] = topTwo(v)
% Return the two largest elements in numeric vector v as scalars. length(v)>=2.
% largest:        a scalar, the largest element in v
% secondLargest:  a scalar, the second largest element in v
% Example 1:  topTwo([5 2 6]) returns two scalars, 6 and 5.
% Example 2:  topTwo([5 2 5]) returns two scalars, 5 and 5.
```

Do *not* implement `topTwo`; you will just call it.

Ada and Bok play a game with seven standard dice with faces that are labelled 1 through 6. A larger face value is "better" in this game. In the first round of the game, Ada has four dice while Bok has three. In each round of the game, the players roll all their dice and then compare their **second best** roll. The player with the higher second best roll wins 2 points. If there is a tie, the player with fewer dice wins 1 point. After a tie in a round, the player with four dice gives one of their dice to the other player so that the other player has more dice, i.e., the two players swap their number of dice.

As an example, suppose in one round Ada rolls 1, 6, 5, 3 and Bok rolls 5, 2, 5. Ada's second best roll is 5 and Bok's seond best roll is also 5. Bok has fewer dice and therefore wins the tie, earning 1 point.

Complete the script on the next page to simulate Ada and Bok playing this game, stopping as soon as either player has accumulated at least 50 points. Display the end result of the game: "Ada won" or "Bok won". For full credit, make effective use of the functions `randInts` and `topTwo`.

Add your code below for Question **4.2**.

```
% 4 suggested variables that you can choose to use or not use
scores= zeros(1,2);  % Index 1 for Ada; index 2 for Bok
aDice= 4;      % Ada has 4 dice at beginning of game
bDice= 3;      % Bok has 3 dice at beginning of game
tieWinner= 2;  % Bok has fewer dice now and will win if there's a tie in a round
```

```
% Example solution

while  scores(1) < 50  &&  scores(2) < 50

    aRolls= randInts(aDice, 1, 6);
    bRolls= randInts(bDice, 1, 6);
    [aBest, aHigh2]= topTwo(aRolls);  % OR  [~, aHigh2]= topTwo(aRolls)
    [bBest, bHigh2]= topTwo(bRolls);

    % Compare 2nd best roll
    if aHigh2 > bHigh2
        scores(1)= scores(1) + 2;
    elseif bHigh2 > aHigh2
        scores(2)= scores(2) + 2;
    else
        scores(tieWinner)= scores(tieWinner) + 1;
        % Swap aDice and bDice, update tieWinner
        tmp= aDice;
        aDice= bDice;
        bDice= tmp;
        if tieWinner==1
            tieWinner= 2;
        else
            tieWinner= 1;
        end
          % Can replace above if with   tieWinner= min(tieWinner+1, 2)
    end
end
if scores(1) > scores(2)
    disp('Ada won')
else
    disp('Bok won')
end
```

Question 5 (21 points)

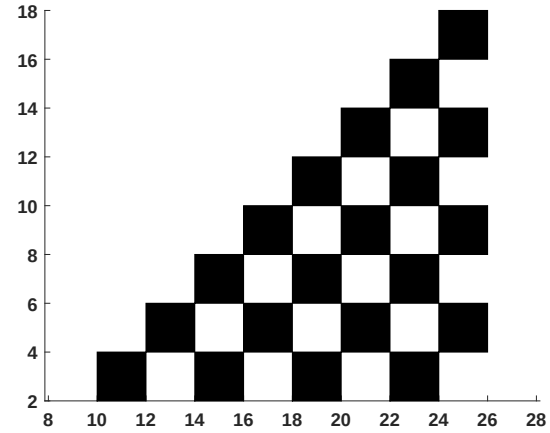
Implement the following function as specified. Assume the availability of a function `DrawRect(x,y,w,h,c)`, which draws a rectangle with its lowerleft corner at the coordinates (x,y), width `w`, height `h`, and filled with the RGB color specified by the 3-vector `c`.

```
function pattern(n, s, a, b)
% Make a checkerboard pattern forming a triangle by drawing black squares.
% Starting from the bottom, the first two rows each has n black squares, offset
% to form a checkboard pattern.  The next two rows each has n-1 black squares,
% and next two rows each has n-2 black squares, and so forth, continuing the
% checkerboard pattern and forming a triangle with a vertical right edge as
% shown in the example diagram.  The top two rows each has one black square.
% n: the number of black squares in the bottom row, an integer greater than 1
% s: the side length of each black square in the pattern
% The lowest x- and y-coordinates of the pattern (lowerleft corner of black
% square) are the point (a,b).
% The only built-in functions allowed are zeros, length, rem

figure; axis equal; hold on
black= [0 0 0];  % rgb vector for black

hold off
```

Example output from `pattern(4,2,10,2)`



```
% Example solution 1 : Draw 2 rows at a time
yLowRow= b;  % ycoord of lowest row
for k= n:-1:1
    % Draw 2 rows, each with k squares, offset for checkerboard
    yNextRow= yLowRow+s;  % ycoord of the higher of two rows
    x= a + (n-k)*2*s;     % xcoord of first square in lower row
    for c= 1:k
        DrawRect(x, yLowRow, s, s, black)
        x= x + s;
        DrawRect(x, yNextRow, s, s, black)
        x= x + s;
    end
    yLowRow= yLowRow + 2*s;
end
```

```
% Example solution 2: Draw n "diagonal lines" of sqrs
xStart= a;
yStart= b;
for d= n:-1:1
    % Draw 2*d black sqrs on a diagonal
    for k= 1:2*d
        x= xStart + (k-1)*s;
        y= yStart + (k-1)*s;
        DrawRect(x, y, s, s, black)
    end
    xStart= xStart + 2*s;
end
```