W1005 - Fall 2014
Homework 2

- Due by Friday 4pm (Oct. 3rd).
- See submission instructions.
- Always include your name and UNI at the top of your submitted files.

1. A MATLAB guru challenges you for the following game: you start by choosing any positive integer (1,2,….), call it X.
   If X is an even integer you divide X by 2.
   If X is an odd integer you multiply X by 3 and add 1.
   You continue this procedure until at some point you get 1 as your next integer. At that point you stop.

   For example: suppose you choose 5 as your integer. Then you will generate the following sequence: 5, 16, 8, 4, 2, 1.

   The guru claims that this is true for ANY integer you choose. That is, you will always end up with 1 at some point.

   a) Guess whether the claim is correct or not, and give a brief justification (NOTE: the question is ONLY asking for a guess, not a proof).

   b) Open a blank script and call it '**game1.m**'.
   Write a script that solves the problem above.

   - The first few lines of your script should include comments which answer part (a),
   - Assume that the value of X is hardcoded as the next line of your script (after the comments).
   - Hint: you should use loops.
   - Store the sequence you generate in the variable S (a column vector). Display S and its length.

   c) Open a blank file and call it '**game2.m**'
   Write a function that solves the problem above.

- Your function should take one input argument (the integer X)
- Your function should have one output argument (the vector S)
- You can copy code from part (b)

2.

a) Type ‘help switch’ and read about switch-case statements.
b) Create a script named ‘**descent.m**’. Give a brief but concrete example when switch-case statements should be preferred to if-elseif statements. Use block comments to enter your answer in the script.
c) Your script should contain code that solves the following problem: A user hard codes two variables, N and C (both positive integers). Write a switch-case statement where C is the switch-expression.

   If C = 1: you should display all the EVEN values starting at N going down to 1 (use “for loop”).
   If C = 2: you should display all the EVEN values starting at N going down to 1 (use “while loop”).
   If C = 3: you should display all the ODD values starting at N going down to 1, but skip the numbers 9 and 5 (use “for loop”).
   If C > 3, you should display an error message of your choice.

3.

a) Write a function named ‘**edges**’ which has 4 input parameters {numR, numC, C, THR} and outputs one parameter {T}. Your function should do the following:
b) Create a random matrix M of size numR x numC (the elements of M are random values between 0 and 1)
c) Multiply each element in M by the constant C and compute the floor. For example, if M(1,1) * C = 2.3 then the floor would be 2. You should initialize a new matrix M2 which holds the new (floor-ed) values.
d) Suppose M2 represents a directed graph. If M2(i,j) = K > 0, then there exists a directed edge from i to j and the length of this edge is K (if K = 0, there is no edge). For every edge, whose length is at least THR, print the word ‘edge’ followed by the vertices of the edge and its length.

e) The function should return the total length of all edges (whose length is at least THR) in the graph in the variable T.

4.

a) Write a function named ‘**my_stats.m**’ which has three input parameters {M1, M2, M3} and one output parameter T. We shall assume that M1,M2,M3 are three matrices of different dimensions.
b) First check to see that each of the input parameters is indeed a matrix (not a vector). If it is not, display an error message of your choice, set T to -1 and terminate the function (hint: use ‘return’).
c) Assuming the condition in part (b) has been satisfied, compute the following stats on your matrices:
    1. The average of all elements (in all three matrices).
    2. The total sum of all elements (in all three matrices) that are less than 5.0.
    3. The standard deviation of all elements.
    4. The average of all elements (in all three matrices) which are greater than 1.5.

    The variable T should store the answers to the above. For example T(1) should have the average, T(2) should have the sum, and etc.

5.

a) Write a function called ‘**my_randg.m**’ to create and return a matrix ‘R’ of size MxN (M,N are input arguments) of random numbers generated according to the following simple distribution:
    For each element (i,j) of R,
    - R(i,j) = 1 with probability 1/5
    - R(i,j) = 2 with probability 1/5.
    - R(i,j) = -1 with probability 3/10.
    - R(i,j) = -2 with probability 3/10.

Your zip folder should include the following files:

- game1.m
- game2.m
- descent.m
- edges.m
- my_stats.m
- my_randg.m