MATLAB 1

There are three exercises as part of this first assignment. Exercises 1 and 2 are worth 30 points and exercise 3 is worth 40 points.

Exercise 1 | Exercise 2 | Exercise 3

Creating Matrices

```
30 points [10 Points Per Part]
```

In this exercise, you create matrices manually or randomly per instructions. You may need the following commands for Part C.

- The command `rand(n)` generates an n x n matrix of random numbers between 0 and 1
- The command `rand(m,n)` generates an m x n matrix of random numbers between 0 and 1
- The command `randi([min,max],m,n)` generates an m x n matrix of random integers between `min` and `max` numbers (inclusive)
- The command `[m,n] = size(A)` returns two numbers, m, and n, where m is the number of rows of A and n is the number of columns of A

A. Create a matrix manually
   - Create the following matrices in MATLAB and display the output

$$A = \begin{bmatrix} 1 & 2 & 3 \\ -4 & -5 & -6 \\ 7 & 8 & 9 \end{bmatrix}, \; b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \; c = \begin{bmatrix} -1 & 1 & -1 \end{bmatrix},$$

$$D = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 0 \\ 2 & 3 & 4 & 0 & 1 \end{bmatrix}$$

   - Run the following commands, display the output, and briefly describe the result of each command
      - `x1 = A(2,:);`
      - `x2 = D(:,4);`
      - `x3 = [A b];`
      - `aug = [A c];`

        Note: You will get an error message for this command, and you need to make a minor fix to it so that the output is a 4x3 matrix with vector c as its 4th row

B. Create special matrices: Use the following commands, display the output, and briefly describe the result of each command
   - `x4 = eye(8);`
   - `x5 = zeros(6,3);`
   - `x6 = zeros(5);`
   - `x7 = ones(3,5);`

- `x8 = diag(c);`

C. Create random matrices: Use the commands below to generate the matrices, display the output, and briefly describe the result of each command
- `F1 = randi([-7,7],3,7);`
- `F2(:, [3 6]) = F1(:, [6 3]);`
  - Write this command on the line <u>after</u> `F1 = F2;`
- `E = [A F2];`
- `[m,n] = size(E);`
- `E1 = E(:, [3 7]);`
- `E2 = E(:, 3:7);`

Solving Ax = b

```
30 Points [10 Points Per Part]
```

In this exercise, you use the RREF and Rouché-Capelli Theorem to determine if A**x** = **b** is consistent. You may need the following commands for this exercise.

- The command `rref(A)` returns the reduced row echelon form of a matrix A
- The command `[B, pivcols]=rref(A)` returns `B` and `pivcols`, where matrix `B = rref(A)` and `pivcols` indicates pivot columns of A
- The command `rank(A)` returns the rank of a matrix A

You also need two m-file functions `rank_comp` (given) and `LS_solution` (template only). In these functions, you need to know:

if-else statement

If an expression is true, it executes statements 1 block. Otherwise, it executes statements 2 block.

```
if expression
    statements 1 block
else
    statements 2 block
end
```

if-elseif statement

If expression 1 is true, it executes statements 1 block. If not, check expression 2. If expression 2 is true, it executes statements 2 block. If not, execute statements 3 block.

```
if expression 1
    statements 1 block
elseif expression 2
```

```
    statements 2 block
else
    statements 3 block
end
```

A. Use the reduced row echelon form (RREF) to solve A**x** = **b**, where A and **b** are indicated in Exercise 1, and type your answer for the following in the Live Editor.
   - Display the reduced row echelon form and the pivot columns of the augmented matrix [A **b**].
   - Write a report to explain if there is a solution of A**x** = **b** based on rref([A **b**]).

B. Use the Rouché-Capelli Theorem by comparing the rank(A) and rank([A **b**]).
   - Call the function `rank_comp` and determine if A**x** = **b** is consistent.
   - Compare the result with Part A. You should have the same conclusion.

> **Note:** `rank_comp` gives you a new command to compare if rank(A) = rank([A **b**]). When you need to use the command, read the comments in the m-file function first and call the function by typing the name of the function, `rank_comp(A,B)`, where A and B are the two input matrices.

C. Open the template `LS_solution`. Use the Rouché-Capelli Theorem as a *guide* and code the function using an if-elseif statement (do not use nested if-else statements). When `LS_solution` is called it should return the correct system type based on whether A**x** = **b** has a solution and how many. The function should output/return/set `system_type` to:
   - `inc` if the system is inconsistent and has no solution, or
   - `con_with_one_sol` if the system is consistent and has a unique solution, or
   - `con_with_inf_sols` if the system is consistent and has infinitely many solutions.

> **Note:** You should include three input arguments (n, A, Ab) for `LS_solution`, where n is the number of variables in the system of equations.

Underdetermined and Overdetermined Systems

```
40 Points [10 Points Per Part]
```

The following system of linear equations is called underdetermined because there are more variables than equations.

Similarly, the following system is overdetermined because there are more equations than variables.

You will explore whether the number of variables and the number of equations have any bearing on the consistency of a system of linear equations.

A. For underdetermined systems:
    - Create a 2 x 1 vector **bA** with each entry between -7 and 7 using the same command `randi`
    - Create a 2 x 3 random matrix A1 with each entry between -7 and 7 using the command `randi`
    - Call `LS_solution` to find the solution `sol1` for the system A1**x** = **bA**
    - Repeat the process above for the systems A2**x** = **bA** and A3**x** = **bA** to find `sol2` and `sol3`, respectively
B. Explain why (must use the concepts learned in Linear Algebra and may use the terms pivot or free variable in your explanations) you would expect most underdetermined linear systems to have infinitely many solutions.
    - Can an underdetermined linear system have a unique solution? Why or why not? Provide a reason using Linear Algebra.
    - An underdetermined linear system can have no solution. Provide an example of an inconsistent underdetermined linear system. Do this by setting your example's matrix to `example_A1` and vector to `example_b1`.
C. For overdetermined systems -- repeat the same process
    - Create a 3 x 1 vector **bC** with each entry between -7 and 7 using the same command `randi`
    - Create a 3 x 2 random matrix A4 with each entry between -7 and 7 using the command `randi`
    - Call `LS_solution` to find the solution `sol4` for the system A4**x** = **bC**
    - Repeat the process above for the systems A5**x** = **bC** and A6**x** = **bC** to find `sol5` and `sol6`, respectively
D. Explain why (must use the concepts learned in Linear Algebra and may use the terms pivot or free variable in your explanations) you would expect most overdetermined linear systems to be inconsistent.

- An overdetermined linear system can have one solution. Provide an example of an overdetermined linear system with one solution. Do this by setting your example's matrix to `example_A2` and vector to `example_b2`.
- An overdetermined linear system can also have infinitely many solutions. Provide an example of an overdetermined linear system with infinitely many solutions. Do this by setting your example's matrix to `example_A3` and vector to `example_b3`.

> **Note:** When you provide examples for Exercise 3B & 3D, they should be non-trivial examples. It means that a matrix does not contain a zero row and does not have two or more identical rows.