**ENED 1090: Engineering Models I**

**Homework Assignment #8**

**Due: Week of November 9th at the beginning of your Recitation Section**

**Problem 1:**  Work through the following MATLAB code by hand and fill in the table. Then check your results using MATLAB.

M = [-1 2 5; 6 2 -4; 7 0 5];

[nrows, ncols] = size(M);

S = zeros(1,nrows);

for row = 1:nrows

S(row) = M(row,1);

for col = 2:ncols

if M(row,col) < S(row)

S(row) = M(row,col);

end

end

end

***Note: The dash in the col column indicates just prior to entering the inner loop***

|  |  |  |
| --- | --- | --- |
|  | | **S** |
| **Before Loops** | | [ 0 0 0] |
| **row** | **col** |  |
| 1 | - |  |
| 1 | 2 |  |
| 1 | 3 |  |
| 2 | - |  |
| 2 | 2 |  |
| 2 | 3 |  |
| 3 | - |  |
| 3 | 2 |  |
| 3 | 3 |  |

**Problem 2:**  Work through the following MATLAB code by hand and fill in the table. Then check your results using MATLAB.

M = [3 1 5; 4 -4 5; 2 -3 -5; 0 1 -1];

[nrows, ncols] = size(M);

Add = zeros(1,ncols);

Total = 0;

for col = 1:ncols

for row = 1:nrows

Add(col) = Add(col) + M(row,col);

end

Total = Total + Add(col);

end

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Add** | **Total** |
| **Before Loops** | | [ 0 0 0] | 0 |
| **row** | **col** |  |  |
| 1 | 1 |  |  |
| 2 | 1 |  |  |
| 3 | 1 |  |  |
| 4 | 1 |  |  |
| 1 | 2 |  |  |
| 2 | 2 |  |  |
| 3 | 2 |  |  |
| 4 | 2 |  |  |
| 1 | 3 |  |  |
| 2 | 3 |  |  |
| 3 | 3 |  |  |
| 4 | 3 |  |  |

**Problem 3:**  For this problem, you will need to download the HW8.mat file. Save it in your current MATLAB folder. At the MATLAB command prompt, enter the following:

>> load HW8

You should see two 4x6 matrices, A and B in your workspace. Create a script file and include the command: load HW8 in your script. Use nested ***for loops*** to accomplish the following:

* Create a matrix C that has entries with the maximum of A and B (i.e., if A(r,c) exceeds B(r,c) then C(r,c) equals A(r,c) otherwise C(r,c) = B(r,c)). ***Display matrix C after the for loops.***
* Count how many entries in Matrix A are greater than the corresponding entries in B. ***Display the count after the for loops.***
* Count how many entries in Matrix B are greater than the corresponding entries in A. ***Display the count after the for loops.***
* Count how many entries in Matrix A are equal to the corresponding entries in B. ***Display the count after the for loops.***

***(Note: it is certainly possible to do this problem without loops, but the purpose of this problem is to practice nested loops so use them in your script)***

Run your code and paste your script and the resulting output in the spaces indicated below.

**Output to Command Window:**

**Script:**

**Problem 4:**  Rock, Paper, Scissors, Lizard, Spock is an expansion of the Rock, Paper, Scissors game. The table below shows how a winner is chosen for each possible combination of choices.

|  |  |  |  |
| --- | --- | --- | --- |
| **Choice 1** | **Choice 2** | **Winner** | **Reason** |
| **Rock** | **Paper** | **Paper** | Paper covers Rock |
| **Rock** | **Scissors** | **Rock** | Rock crushes Scissors |
| **Rock** | **Lizard** | **Rock** | Rock crushes Lizard |
| **Rock** | **Spock** | **Spock** | Spock vaporizes the Rock |
| **Paper** | **Scissors** | **Scissors** | Scissors cuts Paper |
| **Paper** | **Lizard** | **Lizard** | Lizard eats the Paper |
| **Paper** | **Spock** | **Paper** | Paper disproves Spock |
| **Scissors** | **Lizard** | **Scissors** | Scissors decapitates Lizard |
| **Scissors** | **Spock** | **Spock** | Spock smashes the Scissors |
| **Lizard** | **Spock** | **Lizard** | Lizard poisons Spock |

***Note: if both players pick the same thing, it is a tie.***

In this problem, you will write a script to play Rock, Paper, Scissors, Lizard, Spock against the computer.

1. In order to avoid writing a bunch of conditional statements, your script will instead create a Win-Lose-Tie matrix. This 5x5 matrix should have exactly 3 numerical values: one value to indicate a tie, one value to indicate a player win, and one value to indicate a player loss. Decide on your numerical values then fill in the table below to determine what your Win-Lose-Tie matrix will look like.

Value for a Player Win: \_\_\_\_\_\_ Value for a Player Loss: \_\_\_\_\_\_\_\_ Value for a Tie: \_\_\_\_\_\_\_

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | **Player Choice** | | | | |
| **Rock** | **Paper** | **Scissors** | **Lizard** | **Spock** |
| **Computer Choice** | **Rock** |  |  |  |  |  |
| **Paper** |  |  |  |  |  |
| **Scissors** |  |  |  |  |  |
| **Lizard** |  |  |  |  |  |
| **Spock** |  |  |  |  |  |

1. Write a script to play Rock, Paper, Scissors, Lizard, Spock against the computer. Your script should include a while loop that allows the player to continue playing as long as he/she wants along with the following features:

Prior to the while loop:

* Enter your Win-Lose-Tie matrix
* Use the rng(‘shuffle’) command to randomly seed the random function
* Create the following cell array (use curly braces { }, not parenthesis)

Picks = {‘Rock’,’Paper’,’Scissors’,’Lizard’,’Spock’};

Within the while loop:

* Use a menu statement asking the player to pick Rock, Paper, Scissors, Lizard, or Spock
* Use the randi command to generate the computer’s choice
* Use your Win-Lose-Tie matrix and a short conditional statement (if … elseif … else … end) to determine the whether the player won, lost, or tied.
* Output statement(s) to show the Player’s choice, the Computer’s choice, and indicate whether the player won, lost, or tied.
* Keep a count of how many times the player wins, how many times the player loses, and the number of ties.
* Ask the user if he/she wants to play again.

After the while loop, display the counts of wins, losses, and ties.

Run your code (play a few rounds) and paste your script and the resulting output in the spaces indicated below.

**Output to Command Window:**

**Script:**