**Yahtzee**

**Description:**

Write a program to develop a multi-player game of Yahtzee. The program should allow a player to roll 5 dice, see the results of the roll, and select which dice, if any, to re-roll. Players are allowed to re-roll any number of dice up to two times. The player should then be prompted to choose the category in which he/she will score his/her roll. The program should determine the appropriate score based on the player’s roll and fill in the score sheet. Review the rules of Yahtzee.

**Challenges:**

Check for Player Cheating – if the player chooses to score in a particular area but doesn’t have a roll that allows that score.

**Graphics Option:**

Save the Dice.mat file to your current MATLAB folder (the folder in which you will write your game program. Try out the following commands in the command window to understand how they work. Then incorporate them into your game program.

>> load Dice; % Loads cell array (1x6) of dice images

>> roll = randi([1 6] ,[1 5]) % Roll 5 dice – only allow values from 1 to 6

roll =

3 1 6 1 5

imshow( [ Dice{roll} ] ); % Displays the roll graphically for user



**Displaying Scores (A small example that must be extended and can certainly be modified)**

>> score = {'ones'; 'twos';'threes';'fours'}

% Categories – Yahtzee has more than this

score =

'ones'

'twos'

'threes'

'fours'

>> for k = 1:4, for p = 2:4, score{k,p}=NaN; end; end % Adds 3 Players to Score Sheet

>> score

score =

'ones' [NaN] [NaN] [NaN]

'twos' [NaN] [NaN] [NaN]

'threes' [NaN] [NaN] [NaN]

'fours' [NaN] [NaN] [NaN]

>> score{1,2} = 5; score{2,2}=7; score{3,2}=11; score{4,2}=0;

% Throw in some scores for Player 1 (Column 2)

>> score{1,3} = 2; score{3,3}=12;

% Throw in some scores for Player 2 (Column 3)

>> disp(score) % Display updated score sheet

score =

'ones' [ 5] [ 2] [NaN]

'twos' [ 7] [NaN] [NaN]

'threes' [11] [ 12] [NaN]

'fours' [ 0] [NaN] [NaN]

>> player1 = [score{: , 2}]

% Accessing player1 scores (if desired) – creates a vector of doubles

player1 =

5 7 11 0

>> player2 = [score{: , 3}]

% Accessing player2 scores (if desired) – creates a vector of doubles

player2 =

2 NaN 12 NaN

>> isnan(player2) % isnan is a useful command

ans =

0 1 0 1

>> find(isnan(player2))

ans =

2 4