Final Project Assignment

In this project, you will implement a board game. Your board will consist of a path from the upper left corner of a figure (START) to one of its lower corners (FINISH). The path is constructed from a sequence of spaces with rectangular or irregular shapes. This game is based on luck only (i.e., it is not a strategy game). There can be 1-4 players. Each player is represented with a token of a certain color and/or shape that moves from space to space based on an outcome of a dice roll.

Each space on the board can be either empty or it can correspond to a certain *action* that the player must complete. Below is a list of actions that you should implement and their codes (identifying numbers):

1. Jump ahead by 2 spaces.
2. Jump ahead by 4 spaces.
3. Jump to FINISH.
4. Go back 2 spaces.
5. Go back 4 spaces.
6. Go back to START.
7. Lose 1 turn.
8. Roll again.
9. Go back 3 spaces. Lose a turn.
10. Jump ahead by 3 spaces. Roll again.

Your main function should accept three inputs (arguments):

1. The number of players (1-4).
2. The name of a text file with the board description (optional).
3. The name of a text file with the sequence of dice roll results for each player (optional).

Your function should read the text file and convert its contents into a figure with a drawing of a game board. Each line in the text file corresponds to one rectangle and contains five numbers: x y w h a, where
x – x-coordinate of the rectangle
y – y-coordinate of the rectangle
w – width of the rectangle
h – height of the rectangle
a – action corresponding to the rectangle (1-10, -1 for no action, 0 for START, 100 for FINISH).

For example, a line of 20 20 25 10 4 in the text file would be eventually implemented as

```
rectangle('Position', [20 20 25 10], 'FaceColor', [0.7 0.7 0.7], 'EdgeColor', [0.3 0.9 0.3],
'LineWidth', 2);
text(23, 25, char(Jump ahead by', '5 spaces'), 'FontName', 'Verdana')
```

Note that the color of the rectangle was chosen randomly.

Dice roll results are saved in a text file as a sequence of integers between 1 and 6. The end of a sequence corresponding to one player is marked with number 0. For example, a sequence of 1 2 5 3 6 6 5 0 1 5 1 3 3 4 4 0 means that the first player rolled 1 2 5 3 6 6 5 and the second player rolled 1 5 1 3 3 4 4.

The reason why the text files are optional arguments to your main function is because your program will be operating in two modes. In Mode 1, you will build the game board and play your game according to the two input text files. Note that you will be given sample files to test your program but they will be different from the files that will be used during the final exam period.

In Mode 2, you will create your own board and generate your dice roll results (using the randi function in MATLAB). You can use the first function argument to provide a description of your own board to the program but you do not have to. You should be using no less than 15 spaces in your game.

The goal of Mode 1 is for me to evaluate if your program works correctly. The purpose of Mode 2 is for you to get creative and have fun coming up with your own board drawing and your own actions (in addition to the 10 listed above).

Below is a rough structure of your program to get you started.

1. If the second argument is present,
   read the text file and draw the game board based on the contents of the file,
   otherwise,
   draw the game board based on the directions saved in the program (if that is your choice).

2. If the third argument is present,
   read the text file and store the results of dice rolls in a variable.

3. Place tokens corresponding to the players on the START space.

4. For each player in the game,

   If the token is not skipping a turn (because of actions 7 or 9), roll a dice (or read the corresponding dice roll result from a variable) and move the token to its new position (i.e. show the token in a new space on your figure). Implement whatever action needs to be taken.

   If the action places the token at or beyond the FINISH space, display a message on the screen and remove the token from the game. If there are no more players left, quit the game; otherwise, go to the next player or to the next turn.

   Iterate this step until there are no players left in the game.

5. To finish the game, display how many turns it took for each player to complete the game (i.e. who the winner is).

Functions that you may find useful:

- nargin – to know how many arguments a user entered while calling a function
- varargin – search for it in the Help window to learn how to define a function with variable number of arguments (inputs)
- pause – to delay the next turn so the viewer of the game can see what is happening (otherwise, the whole game will take a couple of seconds).
- randi – to simulate a dice roll
- FOR loop – to loop through the players
- WHILE loop – to implement game turns

- IF or SWITCH statements – to execute an action that corresponds to the space were a token is placed

Grading

You will be graded on how well your program works, on its implementation (Did you create a modular code with functions? Did you comment your code?), and on the project report. I will be testing your program with a specific board description file and a sequence of dice rolls (different from the examples provided to you). You will also be graded on the implementation of your own board and game so creativity plays a role here. This part is very open-ended; you can consider using sound and visual effects, colors, different shapes of spaces, additional actions, and a theme for your game. You can create more than one board and select them randomly each time the game plays. You can also decide to write a function to generate a random game board.

Deliverables

The deliverables for this project include:
1. Complete and commented code.
2. Demonstration during final exam period.
2. Final project report.

The complete and commented code of your final project and any supplementary files need to be submitted by 1:30pm on Wednesday March 18 to our Canvas class page. We will be demonstrating the programs during our final exam period on Wednesday, March 18 2:00-3:50pm. Because we have 17 teams, each team will get 6 minutes to demonstrate their project. During the demonstration, your team will run your program on the test board and dice files as well as show us how your program runs with your own inputs.

After the demonstration, you will have 24 hours to correct your code and resubmit it.

Your final project report will be due on Thursday, March 19, at 4:30pm. Please, print out a copy and deliver it to Bannan 209 to my mailbox. In addition, submit your electronic report file on Canvas.

You report should include description of your code including a block diagram of the whole program and of individual functions (when appropriate). You may find it helpful to work on your block diagram(s) before you start coding. You should also include figures with your board design and any other supporting information that will draw my attention to special features in your program.