Problem 1.

In the problem I faced no major issues, I created an abstract class StaffMember so that I knew what my model employee would look like and also so that I wouldn't forget to implement any of the given features.

The only main challenge I faced was forgetting the implicit rule in Java. At first my compiler failed because the implicit construct was inherited, but I hadn't defined an implicit constructor for my abstract class. This meant for each class I had to define the same constructor to fix this issue.

Problem 2.

The biggest challenge for the GUI for the guessing game was setting up the code logic so that the program would compile and that after that the only method that needed to be called was the "activated" method for when I entered in a new value in the text box.

The approach I set up for this was to have the text field activated method filter the response, check the response, and record whether or not it was correct and then increment the appropriate counters. All this was done inside the activated method which meant that I did not have to write a while loop to keep the programming running indefinitely.

Problem 3.

For this problem I decided to create a class with several functions that would all layer so that I could have one public function that would hold the tournament and record the statistics. The first method, the most basic, was to hold a match, and create the random scores and return a string of the match summary. This method also removed the team from teams arrayList if they lost the match. In this manner, I would hold a round that would run a series of matches, and thus the next method; "hold_tournament" would run a series of rounds. Finally, I would have a method "fav_is_winner" that would hold a tournament until the desired team won.

The main challenge here was recording the goals statistics. I had to get creative here and added lines to my other methods so that the statistics where being gathered throughout the whole process and not just at the end.