**CSCE 155N Final Project Report**

Matthew Dohmen

May 3, 2021

In my free time, I am a competitive bowler. I travel throughout Nebraska bowling in tournaments in my free time, and I have even bowled in national youth tournaments in the past. For this reason, I wanted to use MATLAB to create a GUI which could keep score in bowling, just like the computers do at the bowling alley. My GUI does not have nearly as many features as the scoring system at the bowling alley, but it does allow for a single bowler to keep track of their score. This project was approved by Trevor Thomazin.

In thinking of how I wanted to design my GUI, I knew I wanted a pretty basic layout. The ten frames should be displayed at the top, with a number pad centered at the bottom in order to input the scores. There would also need to be a reset button in the bottom left in order to start over if that’s what you wanted to do. I also knew that I wanted the formatting to be similar to what is traditionally used so, for example, even though a strike is thrown on the first ball of a frame, the ‘X’ appears in the second half of the frame. Additionally, the score is updated under each frame to show what your score is currently (and what it was at every point throughout the game). However, these scores traditionally don’t appear until the score in that frame cannot be changed. For example, when you throw a strike in bowling, it gives you ten points, but it also adds the value of your next two shots to that frame as well. So, if you started off the game with a strike and then got a nine followed by a spare in the second frame, the first frame score would not appear under the first frame until those two shots have been thrown, at which point it would be updated to display 20, and thus that 20 would never have to be updated for the rest of the game. These were the rules I had for myself when starting on this project, and I successfully implemented all of them into my GUI.

Before writing my code, I had to think of how I could design the input so my code could not be broken. I came up with a plan so that if you tried to say you got a spare on the first shot, it would give you an error message. Likewise, if you tried to input a strike on the second shot of any given frame (besides the case in the tenth frame where you could throw two strikes in a row), I needed the code to ignore that input as well and give an error message. The final error message would be if you tried to input 10 or more points in the frame without using the ‘X’ or ‘/’ buttons. In addition to this, I had to think through how I wanted the score keeping system to work. As mentioned above, a strike counts as ten plus the value of your next two shots. Likewise, a spare counts as ten plus the value of the very next shot. I needed to design a function which could look at the previous shot(s) to see if there was a spare or 1 or 2 strikes. This is because if, for example, you throw a strike in the first and second frames and then get 8 on your first ball in the 3rd frame, the score for the first frame needs to be displayed as 28. In addition to planning out how this would function, I also needed to think about the tenth frame, because it functions much differently than the rest. If you get a strike or a spare in the tenth frame, you get to throw a total of three shots in the tenth instead of two. Because of these additional shots, the strikes and spares in the tenth simply just count as 10 points. This meant I had to come up with different parameters for the tenth frame in order to count the score correctly. By having a good understanding of all the scoring cases possible, it made it easier to write the code.

While writing the code, I encountered many problems. The first problem was that the checkMark function needed to be able to look back at previous frames to check for a spare or strike(s). However, if it’s the first frame there are no previous frame to look back at, resulting in an error. Likewise, if it is the second frame, there is only one frame to look back at, so I needed to make sure the code did not try to check for 2 strikes in a row leading up to that frame. These were simple fixes, as I was able to just add additional parameters to my if statements and create additional if statements for those specific cases. Within this same function, I had problems when trying to calculate the scores for the 10th frame. This is because a strike in the tenth frame gets put in the first box, so I needed the code to only look one, and possibly three, boxes back to check for a strike or spare instead of two (and possibly four). This was also fairly easy to solve, but it meant I had to rework the entire scoring system independently for the tenth frame. I also had many problems working through the logic of the score keeping system. Even though I had thought it through and had a good understanding of what needed to be displayed, it was difficult to add all the parameters to correctly display the score at each individual frame and reveal that score at the proper time. This required many iterations, but I finally came up with a system that was slightly less complex than I was originally trying to make it, although likely far from the most efficient system possible. The other major problem I had showed up at the very end of the project. When you finish a game, a box pops up asking if you would like to start a new game. This line of code is at the end of the checkMark function, but the problem is that I had my code set up to run the scoreKeep and subsequent checkMark function, and then going back to add 1 to the frameCounter variable to take you to the next shot. This meant that when you would start a new game, it would reset the board, then go back and add 1 to frameCounter, starting you off on the second ball of the first frame. I had to use the debugging feature in MATLAB in order to figure out what was going on and add an extra global variable to stop that from happening immediately after the reset function is called.

I really enjoyed making this GUI, because it created a system that is more user-friendly than simply creating a function to do a similar problem. In the future, I could make GUIs in order to solve more complex problems, like graphing functions with the option to intuitively customize how that function appears on the graph. I could also make GUIs to function as complete games like chess or checkers. I honestly do not know a lot about engineering at this point so it is hard to say how this could be applied to my career, but I am glad I know how to create these GUIs for when the time comes when something like this is useful.