NFL Fantasy Football Mock Draft Simulator

Jayden Slotnick and Payton Glynn

# Introduction

This report will outline the methodology of the Fantasy Football Mock Draft Simulation code. The goal of this project was to utilize existing rankings to create a mock draft simulation environment for the user. Fantasy football is a widely played game by millions of Americans. The game of fantasy football includes drafting real football players onto a fake fantasy team where they score points for you. Typically, there are fantasy football leagues of about 8-12 people, and they consist of major football positions such as quarterback, running back, wide receiver, and tight end. The league settings of every fantasy football league are slightly different, and this code provides the user flexibility to adjust key parameters such as league members, roster limits, and positional limits. In addition, this code will give the user the option to control a team and draft players themselves or have an automated process do it for them. This tool gives the user the ability to simulate three draft scenarios which are a standard league, a PPR (points per reception) league, and a half-PPR (half a point per reception) league. In summary, this code gives a unique user experience for mock drafting and is a potential tool to use before the football season to give a player practice drafting a team.

# The “ranking” Class

The purpose of the ranking class is to take a user input on a particular draft a user would like to run (standard, PPR, half-PPR). Once the user is prompted for the input of the draft type, the draft type is inputted into the ranking class. Then an object is created for each league type. The class will then output the rankings of the players for the desired league type. The main function contained in this class is the originalOperate function. This function operates by opening the CSV of the correct league type using ifstream. The rankings are then copied over into a vector container. This process is completed for each league type with 3 if statements so it will only run the chosen league type. Lastly, operator overloading is used to help print the output of the container. The format of the output is… Player: XXX, Position XXX. The XXX in each would be replaced with real NFL players and their respective positions. The concept of function overloading is also used by the functions operate() and operate(int val). Therefore, if there is no input, the code will run the standard case which is the league case. If there is an input of a league type, the code will run the case specified. At the conclusion of the class, each of the CSV files are closed and the player rankings for the chosen league type is outputted.