**CSE 590-54/59 Homework Assignment 1 (Due 2/4/22)**

**(American) Football Simulation**

For this programming assignment, you will be producing a heavily boiled-down (American) football simulator and play visualization using Python. The process should give you experience with built-in and custom functions with the language, process simulation, and very simple depiction of data. There are four interrelated problems with a cumulative value of 100%, and the optional components for problems 1-3 can be completed for 2-3 extra credit points each (with the max grade not exceeding 106%). **For full credit, you must include brief documentation for your code at least, including a very simple README (which should indicate, for example, optional components you completed and any changes from the norm in your code) and a short (line or two) comment for each function you implement, indicating what it does.**

1) **(20 pts)** In football terminology, a **down** is the period in which a single play (successful or failed) is executed. You are to write the function **down(successprct, yardrange)**, where *successprct* is a number between 0-100 indicating probability of offensive success (i.e. because passes can be incomplete or plays otherwise unsuccessful), and *yardrange* is a tuple with two values representing the minimum and maximum number of yards gained. Your function should return a number of yards according to the following rules: i. a random number between 1 and 100 is generated -- if it exceeds successprct then the play "fails" and 0 is returned, otherwise ii. the number of yards returned is equal to a random number between the min and max according to yardrange.

Optional: if you like, you can include "sacks" or "penalties" in your code, presumably which have an attached percentage chance of happening and incorporate them into the down function.

2) **(30 pts)** A **drive** represents a series of downs that result in either a touchdown or turnover (assume no punting or field goal is possible). You are to write the function **drive**(**yards\_to\_TD, successprct, yardrange) where** *yards\_to\_TD* is the number of yards a team must move the ball to achieve a touchdown, and *successprct* and *yardrange* are identical in form to their down counterparts. The drive function must do the following: up to four downs will be executed in sequence using the *down* function above with *successprct* and *yardrange* as arguments, with the following details: i. the number of yards generated by the down function will be subtracted from yards\_to\_TD on each call ii. if yards\_to\_TD ever reaches zero or below, the team scores (see below) iii. If four sequential plays take place and the team *doesn’t* score, the ball is turned over to the other team with zero points scored (see below).  
The output of *drive* will be a tuple with two elements representing i. points scored and ii. field position for the other team, respectively. If the team scores (yards\_to\_TD reaches zero or below), then the first element will be 7 (90% of an extra point attempt to succeed) or 6 (10% of the attempt to fail) – otherwise the first value will be 0. If the team scores then the second element will be 80 (reflecting a kick-off/touchback position for the other team) – otherwise the second element will be calculated as  
100-yards\_to\_TD (reflecting the other team moving the ball in the opposite direction.)

Optional: if you like, you can include the notion of “first downs” in your code: in this case, the down counter “resets” to down 1 (first down) whenever ten or more positive yards are gained cumulatively within the four sequential downs.

3) **(30 pts)** Here you are to create a simple visual depiction of a drive. Function **drive\_depicted**(**yards\_to\_TD, successprct, yardrange)** is identical in form and return to the drive function, with the exception that it must *also* provide visual details of every down within the drive, including the progress of the side on offense (i.e. how many yards from their own end zone) and the yards remaining to touch-down (i.e. how many yards from the opponent’s end zone). The nature of the visualization is flexible, but the simplest approach is an ascii-like depiction as in the examples below:

*Ex 1: Successful drive*O----|---->----|----|----|----|----|----|----|----X 1st Down, 80 Yds to Go

O----|----|----|----|--->|----|----|----|----|----X 2nd Down, 51 Yds to Go

O----|----|----|----|----|----|----|>---|----|----X 3rd Down, 28 Yds to Go

O----|----|----|----|----|----|----|----|----|----T TD Scored! Xtra Pt Made

Ex 2: Failed drive  
O----|---->----|----|----|----|----|----|----|----X 1st Down, 80 Yds to Go

O----|----|----|--->|----|----|----|----|----|----X 2nd Down, 61 Yds to Go

O----|----|----|--->|----|----|----|----|----|----X 3rd Down, 61 Yds to Go

O----|----|----|----|----|----|>---|----|----|----X 4th Down, 38 Yds to Go  
O----|----|----|----|----|----|----|---Q|----|----X Turnover, 21 Yds to Go

Note that in the above examples yards-to-go is being rounded to the closest even yard for depiction purposes – this is not essential, but may make visualization a bit less clunky in practice.

Optional: If you like, you can use any of the Python figure/image/animation libraries to make a more sophisticated visualization of drives. But be aware that this can become a very complicated and difficult effort if you don’t have previous experience!

4) **(20 pts)** Your last step is to create a football game simulation. While normal football is based on quarters and time limits, you can assume a game has a fixed number of alternating drives between the two teams, and each of the teams has a different success-rate/yard-gain as input to the drive function. More specifically, you are to write function **simulategame(num\_drives, prctT1, yrangeT1, prctT2, yrangeT2),** where *num\_drives* is the total number of drives played in the game by each team, *prctT1* and *yrangeT1* correspond to the successprct and yardrange for the first team (T1), and *prctT2* and *yrangeT2* correspond to the successprct and yardrange for the second team (T2). The function will do the following: i. initialize scores for both teams at 0 and yards\_to\_TD at 80, ii. call the drive function with yard\_to\_TD and team 1’s successprct/yardrange values, iii. increment T1’s score according to drive’s return and adjust yards\_to\_TD according to the second return (see problem 2 above), iv. call the drive function for team 2 using its parameters and v. adjust T2’s score if relevant, and vi. repeat steps ii-v *num\_drives* time in total. Your return should be a 2-element tuple, with the first element being T1’s final score, and the second being T2’s final score.

It is highly advised that you test your functions work correctly by calling them from an outside script.

**Your submission to Blackboard should be a single .zip file with the name <LN>\_<FN>\_1.zip, where <LN> and <FN> are your last name and first name respectively. As outlined above, the file should include your jupyter notebook and/or python file(s), plus a README file for documentation and to guide execution.**