Project Description:
Name: 112 Bowl

I want to create a football video game, where there is both a multiplayer and a single player option. You will have the options to play as or play against different teams, and you will be able to choose from different plays. The game will keep time and score, and scroll as you move.

Similar projects:
My project was heavily inspired by the mobile game Retro Bowl, but I do have some changes to make. I expect that I will make the graphics similar to Retro Bowl, which has a classic or retro arcade design. However, some mechanics that I want to change are the fact that in Retro Bowl, the game just gives you a play to run, but in my game I plan to have the user choose between around 6 different plays. Another difference is that this game has a multiplayer gamemode, where the player has the ability to control the defense in the game, but there is only the ability to control the offense in Retro Bowl.

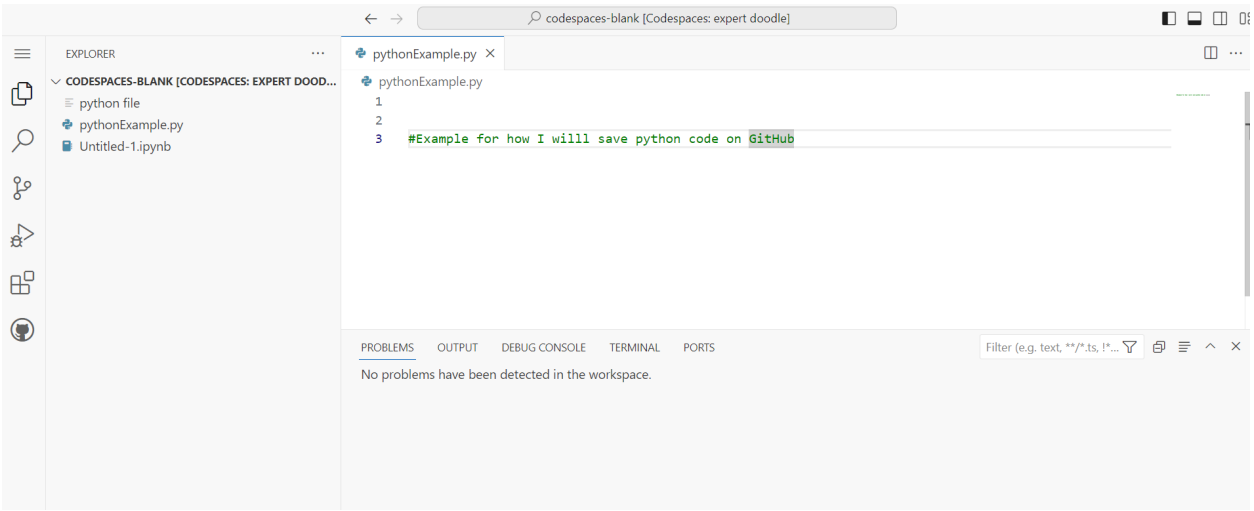
Structural Plan:
For the different teams, I plan to have a text file with a very long list of dictionaries, and each of these dictionaries will contain information about the team, such as sprites for the different uniforms, and different stats for the different players. For the players, I expect to use classes, and these classes will be different positions, as for example there will be a wide receiver class. Each instance of this class would correspond to one player on the field, and this class would contain different variables, such as their position and velocity. I will also extract information from the team dictionary to give each player different stats such as max speed, acceleration, which represents how quickly they can reach their top speed, and agility, which represents how quickly they can decelerate and change direction. The instances will also contain the player's statistics throughout the game, which will be displayed on the pause screen. I expect the pass plays to pass information through a dictionary, with keys corresponding to the receivers, and values corresponding to where they are going on the field. For the actual throwing of the ball, I will first use a function that determines where the ball will land, and how to make the parabola when throwing the ball. I will also organize the defense into different functions for defending the pass, defending the run, and defending after the receiver has caught the ball. For running the play, there will be a runPlay(app, play) which takes in a play in the form of a dictionary, and all of the instances of the various players of the class will read instructions contain in the dictionary to move to the right places in the play, and perform the correct actions. There will also be functions for pauseGame(app) and resetGame(app)

Algorithmic Plan:
One of the hardest things about this project will be creating a computer-controlled defense that is both difficult and realistic. As I kind of pointed to earlier, I think there should be two modes for the defense. One where the defense is defending the pass, and one where the defense is trying to tackle the ball-carrier. For the former, I will have the defensive player guarding the receiver mirror the velocity of the receiver, but when there is a change of direction, I will use the time module to delay the defenders mirroring, to make the gameplay more realistic. There will also

have to be a safety who is playing zone defense. The algorithm I will generally follow is that it will look at the coordinates of all the offensive players who are 15 yards past the line of scrimmage, and it will try and find the point where the sum of the distances from the offensive player is minimized, and it will start moving towards this point. Then, when the ball is thrown, it will move towards where the ball is going to land. When the defenders are trying to tackle the ball-carrier, I will use an algorithm that will use the acceleration, position, and the velocity of the offensive player to calculate what position the offensive player is expected to move to, and it will update the acceleration of the defender to move to this direction. The next most difficult aspect will probably be getting the offense to run the plays. For the receiver instance, I want to have a variable, which is a list of tuples, which is the list of direction it will run for the route, and for example [(0,10),(10,0)] would have the receiver move 10 to the right, and then 10 forwards. As I mentioned earlier, each player would be a dictionary, where the keys would be the different receivers, and the values would be these lists that correspond to routes. Another algorithm is going to be for the physics of the players, and as I mentioned before, the main variables for the physics will be acceleration and velocity, and the main statistics that will determine this in the player class will be the acceleration state, max speed, and agility stat. The max speed stat will ensure that the magnitude of the velocity never goes past a certain point, meaning that if you are at this max speed, you cannot accelerate in the direction you are going. The acceleration stat will represent how quickly you accelerate in the direction you are going, and the agility stat will represent how quickly you accelerate in the opposite direction of where you are going. Also, the arrow keys will represent velocity to a certain x or y direction, and pressing none of the arrow keys will represent acceleration to the <0,0> velocity.

Timeline:
My goal for TP1 is to get the game to run an offense, where the user can choose the play, and can run the play, but there will be no defense.
Then between TP1 and TP2 is to implement the defense, and have it so you can play a full game. I will create both a singleplayer and a multiplayer way to play the game.
Between TP2 and TP3, I want to incorporate the ability to play as different teams and I want to incorporate a main menu screen, stat-tracking, and pause screen. Also I will refine the gameplay to enhance the user experience to the fullest extent I can, and if I have time, I will add the ability to run different defensive players also.

Version History: I will use the GitHub cloud to save files of my code and back them up.




Module List: I will not any any additional modules, except for basic builtin ones such as math and random.

TP1 Update:
None, except I am now using Google Docs to backup my code instead of GitHub

TP2 Update:
No new updates.

TP3 Updates:
I have decided to not include a one-player mode due to limited time, but for the most part my final project stayed true to the original design.

- Updated Defense

Now, the defense can cause interceptions and incomplete passes

- Sprites

Improved sprite animation, sprites now face right or left, and have two frames for each direction

- AI

I have worked on an AI for a player that that not guard a player, but I use an algorithm that determines the most optimal spot to defend. This spot is found by drawing a grid with spots 50 pixels away from each other, and for each spot, I calculated the sum of the distances between each offensive player, and weighted these shares based on how open each player is.

- Rush Clock/Additional Defensive Player

I added this to make the game more difficult and more fair, but at the start of each play, there is a 5 second time and before this timer runs out, the qb can neither go past the line of scrimmage and run the ball, and the defense cannot go past the line of scrimmage to tackle the qb. However, I also added a new defensive player, who's role it is to follow the qb's horizontal movements when this timer is running, and run after the qb when this timer runs out.

- Game Records

There is a csv file in my project folder, and there is a button on the start splash screen, which when clicked takes you to a screen in the single-game records for several stats, which tells you the record, the player the record was achieved with, and the date that it was achieved (still working at this). During the game, the app stores stats for all of the players, and at the end, it reads through these stats, and rewrites the csv file to reflect any records broken.