Task 1

- Write up plan/subtasks for each task

Task 2

- Task 2a: main()
    - Write main() function with commented out sections outlining how the code will rn Write read_data function (parse in each line from csv using loop, and append to a list of lists EXCLUDE HEADERS)
- Task 2b: read_data()
    - Save path to file as string variable for ease of access, open file in read mode, read each row using loop, and append to a list of lists
    - Convert data to correct types in the loop (if number, int; if empty, 0; if string, keep as string)
    - Keep line to read headers outside of loop so that it doesn't parse it in
    - have read_data() return list_of_lists
- Task 2c
    - use prewritten code to check dimensions in main(), and make sure that no empty cells exist, as well as that all types of each data are correct (use BD class)
- Task 2d: get_menu_choice()
    - design list
    - create loop that keeps prompting the user with the menu list until they select an appropriate option (check output using print statements if the relevant code hasn't been implemented yet)
    - Menu:
```
1: Line raph of games played for one player over time
2: Line raph of games played for one player over time
separated by team
3: Histogram of runs scored for all players in one year
(no cutoff)
4: Histogram of runs scored for all players in one year
(cutoff = 100)
5: Graph of team presence over time
6: Homeruns over time (percentiles)
7: [FILL ME IN]
8: [FILL ME IN]
0: Exit
```
- Task 2e
    - get_matching_rows(data, BD.columnindex, value):
        - in an accumulator list collect al of the rows for which the value matches within the index (category)
    - get_column_values(data, BD.columnindex):
        - take all daa and collect all the values for the given column as a new list

    - get_unique_values(list):
        - Parses through each item in a list; if the item is present in an accumulator list, skip over it
    - get_unique_column_values(data, BD.columnindex):
        - Parses through column from get_column_values(data, BD.columnindex) as list and uses get_unique_values(list) to get the unique values of the target column from all the rows

Task 3

- Task 3a:
    - Line graph of number off games played per year for a particular player (time series off number of games)
    - Give user option to pick between
      a) entering the player ID themselves
      b) generating a graph for a randomly selected player
    - get_matching_rows for a playerid, then get_column_values to get the number of games played
- Task 3b
    - Line graph of games played for one player over time, separated based on team (time series of multiple players, with key)
    - Get all of the data for one player, and separate the list into lists based on what team they were playing for (get_unique_column_values with data, and BD.team_name) then use get_matching_rows and create lists for a time series of each team.
    - Give the player options of
        a) entering the player id themselves OR
        b) generating a graph for a randomly selected player
- Task 3c
    - histogram of total runs scored for all players for a specific year
    - Hisogram has 100 bins, sum the runs for all rows belonging to that player to calculate their total runs
    - use get_matching_rows and get_unique_column_values to get data to pass into plt.hist()
    - Same function as 3d, but with cutoff = 0
- Task 3d
    - histogram of runs scored for all players in a specific year (with a minimum cutoff)
    - check through the data from 3c and make a list of all players whose total runs value is higher than the cutoff

Task 4

- Task 4a
    - scatterplot of years, and whether or not team was active.

    - team is active is any player on that team has statistics from that year
- Task 4b
    - time series of the maximum number of home runs scoredd each year, the 99th percentile, and the 50th percentile (median home runs scoredd each year)
- Task 4c
    - come up with a new statistic to investigate

Extra credit

- Make plan for extra credit assignments