Scala Assignment Peer Review

Q1.

Dhruv Singla Approach:-

His code prompts the user to enter an input array of floating-point numbers, splits the input by whitespace, and then converts each element into a Float value. If the user inputs invalid values, the code handles the NumberFormatException and other exceptions by printing an error message and returning None.
Then maps each value in the input array to a range of two Float values using the mapFloatToRange function. This function multiplies the input float by 10, rounds it to the nearest integer, and then calculates two Float values that represent the lower and upper bounds of the range based on whether the rounded integer is closer to the previous or next integer.
Finally, the program prints out each input value and its corresponding range in the specified format, where each value is printed with two decimal places, and the lower and upper bounds of the range are printed with three decimal places. The getInput function returns an Option type to handle errors in a more functional way, and the mapFloatToRange function is defined as a private method to encapsulate its logic and prevent code duplication.

Amit Shukla Approach:

His major approach resides in getRange Function.
The getRange method first fetches the last two digits of the input value by taking the remainder of the product of the value and 1000. It then calculates the left and right limits of the bucket range based on whether the last two digits are greater than or equal to 50 or not. If the last two digits are greater than or equal to 50, the left limit is calculated by subtracting the last two digits divided by 1000 and adding 0.050, while the right limit is calculated by adding 99 minus the last two digits divided by 1000. If the last two digits are less than 50, the left and right limits are calculated by subtracting and adding, respectively, the last two digits divided by 1000 and 49 minus the last two digits divided by 1000.

Q2.

Dhruv Singla Approach:

His code defines a class player which represents a cricket player and has attributes like Year, PlayerName, Country, Matches, Runs and Wickets. The apply method in the companion object player is used to create a new player object by taking an array of strings as input and splitting it into the relevant attributes. The printer method is used to print the player details.

The q2 object is used to perform some operations on a dataset of players, represented by an array of strings rawData. The array is refined by converting each string into a player object using the map method and the apply method of the player companion object.

The code then answers four questions related to the player data:

1. finding the player with the highest run score =>used maxBy function to find maximum runs
2. finding the top 5 players by run score=> used sortWith and give comparator logic to get in descending order
3. finding the top 5 players by wicket taken=>used sortWith and his own comparator logic to get in descending order
4. ranking players based on their overall performance, which is calculated by giving a weight of 5x to wickets taken and 5/100x to runs scored.=>again used sortWith and given his own logic for comparator on function which is returning overall performance of any player.

Amit Shukla Approach:

This code defines a case class Player to represent player details like the year, player name, country, matches, runs, and wickets.

Used different functions to find all 4 queries.

1. finding the player with the highest run score =>The highest_run_scored function takes an array of player data and returns the name of the player who scored the highest runs.
2. finding the top 5 players by run score=> used sortBy and give two arguments to sortBy one is runs and the other is *ordering.reverse* which reverse the default order of sorting.
3. finding the top 5 players by wicket taken=>used sortBy and give two arguments to sortBy one is wickets and the other is *ordering.reverse* which reverse the default order of sorting.

4. ranking players based on their overall performance, which is calculated by giving a weight of 5x to wickets taken and 5/100x to runs scored.=>again used sortWith and given his own logic for comparator on statement which is returning overall performance of any player in descending order