Structure Definition:

Player:

name (string): Name of the player.

matchesPlayed (integer): Number of matches played.

runsScored (integer): Total runs scored by the player.

wicketsTaken (integer): Total wickets taken by the player.

battingAverage (float): Runs scored per match (calculated field).

Enhanced Features:

Input and Output:

Allow dynamic memory allocation for storing N players using malloc or realloc.

Input the details of each player, including their name, matches played, runs scored, and wickets taken.

Calculate and display the batting average (runsScored / matchesPlayed) for each player.

Player Analysis:

Identify and display the player with the highest batting average.

Identify the player with the most wickets taken.

Sorting:

Sort players by batting average in descending order.

Sort players by wickets taken in descending order.

Search:

Allow searching for a player by name and display their details.

Team Statistics:

Calculate and display the team's total runs, total wickets, and average runs scored across all players.

Menu-Driven Interface:

Implement a user-friendly menu to perform the above operations.

**********/

```
#include <stdio.h>
#include <stdlib.h>
```

```
#include <string.h>

typedef struct players {
   char name[50];
   int matchesPlayed;
   int runsScored;
   int wicketsTaken;
   float battingAverage;
} Player;

void add_player(Player *player, int player_count);
void display_details(Player *player, int player_count);
void highest_avg(Player *player, int player_count);
void most_wickets(Player *player, int player_count);
void sort_by_batting_avg(Player *player, int player_count);
void sort_by_wickets(Player *player, int player_count);
void search_player(Player *player, int player_count);
void display_team_statistics(Player *player, int player_count);

int main() {
   int n;
   printf("Enter the number of players: ");
   scanf("%d", &n);

   Player *player = (Player *)malloc(n * sizeof(Player));

   if (!player) {
      printf("Memory allocation failed.\n");
      return 1;
   }
```

```
int player_count = 0;
int op;

do {
    printf("\nMenu:\n");
    printf("1. Add player details\n");
    printf("2. Display player details\n");
    printf("3. Find player with highest batting average\n");
    printf("4. Find player with most wickets taken\n");
    printf("5. Sort players by batting average\n");
    printf("6. Sort players by wickets taken\n");
    printf("7. Search player\n");
    printf("8. Display team statistics\n");
    printf("9. Exit\n");
    printf("Choose an option: ");
    scanf(" %d", &op);

    switch (op) {
        case 1:
            if (player_count < n) {
                add_player(player, player_count);
                player_count++;
                printf("Player details added successfully!!\n");
            } else {
                printf("Player limit reached. Cannot add more players.\n");
            }
            break;
        case 2:
            display_details(player, player_count);
            break;
        case 3:
```

```
            highest_avg(player, player_count);
            break;
        case 4:
            most_wickets(player, player_count);
            break;
        case 5:
            sort_by_batting_avg(player, player_count);
            break;
        case 6:
            sort_by_wickets(player, player_count);
            break;
        case 7:
            search_player(player, player_count);
            break;
        case 8:
            display_team_statistics(player, player_count);
            break;
        case 9:
            printf("Exiting.\n");
            break;
        default:
            printf("Invalid choice. Try again.\n");
      }
   } while (op != 9);

   free(player);
   return 0;
}

void add_player(Player *player, int player_count) {
   printf("Enter the player name: ");
```

```
    scanf(" %[^\n]", player[player_count].name);
    printf("Enter the matches played: ");
    scanf("%d", &player[player_count].matchesPlayed);
    printf("Enter the runs scored: ");
    scanf("%d", &player[player_count].runsScored);
    printf("Enter the number of wickets taken: ");
    scanf("%d", &player[player_count].wicketsTaken);
    player[player_count].battingAverage = (float)player[player_count].runsScored /
player[player_count].matchesPlayed;
}

void display_details(Player *player, int player_count) {
    for (int i = 0; i < player_count; i++) {
        printf("\nPlayer %d:\n", i + 1);
        printf("Name: %s\n", player[i].name);
        printf("Matches Played: %d\n", player[i].matchesPlayed);
        printf("Runs Scored: %d\n", player[i].runsScored);
        printf("Wickets Taken: %d\n", player[i].wicketsTaken);
        printf("Batting Average: %.2f\n", player[i].battingAverage);
    }
}

void highest_avg(Player *player, int player_count) {
    if (player_count == 0) {
        printf("No players to evaluate.\n");
        return;
    }
    int index = 0;
    for (int i = 1; i < player_count; i++) {
        if (player[i].battingAverage > player[index].battingAverage) {
            index = i;
```

```
        }
    }
    printf("Player with highest batting average is %s (%.2f)\n",
           player[index].name, player[index].battingAverage);
}

void most_wickets(Player *player, int player_count) {
    if (player_count == 0) {
        printf("No players to evaluate.\n");
        return;
    }
    int index = 0;
    for (int i = 1; i < player_count; i++) {
        if (player[i].wicketsTaken > player[index].wicketsTaken) {
            index = i;
        }
    }
    printf("Player with most wickets taken is %s (%d wickets)\n",
           player[index].name, player[index].wicketsTaken);
}

void sort_by_batting_avg(Player *player, int player_count) {
    for (int i = 0; i < player_count - 1; i++) {
        for (int j = i + 1; j < player_count; j++) {
            if (player[i].battingAverage < player[j].battingAverage) {
                Player temp = player[i];
                player[i] = player[j];
                player[j] = temp;
            }
        }
    }
```

```
    printf("Players sorted by batting average in descending order.\n");
}

void sort_by_wickets(Player *player, int player_count) {
    for (int i = 0; i < player_count - 1; i++) {
        for (int j = i + 1; j < player_count; j++) {
            if (player[i].wicketsTaken < player[j].wicketsTaken) {
                Player temp = player[i];
                player[i] = player[j];
                player[j] = temp;
            }
        }
    }
    printf("Players sorted by wickets taken in descending order.\n");
}

void search_player(Player *player, int player_count) {
    char name[50];
    printf("Enter the player name to search: ");
    scanf(" %[^\n]", name);
    for (int i = 0; i < player_count; i++) {
        if (strcmp(player[i].name, name) == 0) {
            printf("Player found: %s, Matches: %d, Runs: %d, Wickets: %d, Average: %.2f\n",
                   player[i].name, player[i].matchesPlayed, player[i].runsScored,
                   player[i].wicketsTaken, player[i].battingAverage);
            return;
        }
    }
    printf("Player not found.\n");
}
```

```
void display_team_statistics(Player *player, int player_count) {
    int totalRuns = 0, totalWickets = 0;
    for (int i = 0; i < player_count; i++) {
        totalRuns += player[i].runsScored;
        totalWickets += player[i].wicketsTaken;
    }
    printf("Team Statistics:\n");
    printf("Total Runs: %d\n", totalRuns);
    printf("Total Wickets: %d\n", totalWickets);
    printf("Total Players: %d\n", player_count);
}
```