```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConAppPlayerTeamDataSheet
{
    public class Player
    {
        public int PlayerId { get; set; }
        public string PlayerName { get; set; }
        public int PlayerAge { get; set; }
    }
}
```

```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConAppPlayerTeamDataSheet
{
    public interface ITeam
    {
        void Add(Player player);
        void Remove(int playerId);
        Player GetPlayerById(int playerId);
        Player GetPlayerByName(string playerName);
        List<Player> GetAllPlayers();
    }
}
```

```cs
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConAppPlayerTeamDataSheet
{
    public class OneDayTeam : ITeam
    {
        public static List<Player> oneDayTeam = new List<Player>();
        public OneDayTeam()
        {
            oneDayTeam = new List<Player>();
        }
        public void Add(Player player)
        {
            if (oneDayTeam.Count < 11)
            {
                oneDayTeam.Add(player);
                Console.WriteLine("Player is successfully added");
```

```
            }
            else
            {
                Console.WriteLine("Cannot add more than 11 players");
            }
        }
        public void Remove(int playerId)
        {
            Player playerToRemove = oneDayTeam.FirstOrDefault(p=>p.PlayerId ==
playerId);
            if (playerToRemove != null)
            {
                oneDayTeam.Remove(playerToRemove);
                Console.WriteLine("Player is successfully removed");
            }
            else
            {
                Console.WriteLine("Player not found");
            }
        }
        public Player GetPlayerById(int playerId)
        {
            return oneDayTeam.FirstOrDefault(p=> p.PlayerId == playerId);
        }
        public Player GetPlayerByName(string playerName)
        {
            return oneDayTeam.FirstOrDefault(p => p.PlayerName == playerName);
        }
        public List<Player> GetAllPlayers()
        {
            return oneDayTeam;
        }
    }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConAppPlayerTeamDataSheet
{
    internal class Program
    {
        static void Main(string[] args)
        {
            OneDayTeam team = new OneDayTeam();
            string continueOp;
            do
            {
                try
                {
                    Console.WriteLine("Enter 1: To Add Player\t2: To Remove Player
by Id\t3: Get Player By Id\t4: Get Player by Name\t5: Get All Players:");
                    int choice = int.Parse(Console.ReadLine());
                    switch (choice)
```

```
                    {
                        case 1:
                            Console.Write("Enter Player Id: ");
                            int playerId = int.Parse(Console.ReadLine());
                            Console.Write("Enter Player Name: ");
                            string playerName = Console.ReadLine();
                            Console.Write("Enter Player Age: ");
                            int playerAge = int.Parse(Console.ReadLine());

                            Player newPlayer = new Player { PlayerId = playerId,
PlayerName = playerName, PlayerAge = playerAge };
                            team.Add(newPlayer);
                            break;
                        case 2:
                            Console.Write("Enter Player Id to Remove: ");
                            int removePlayerId = int.Parse(Console.ReadLine());
                            team.Remove(removePlayerId);
                            break;
                        case 3:
                            Console.Write("Enter Player Id: ");
                            int searchPlayerId = int.Parse(Console.ReadLine());
                            Player foundPlayerById =
team.GetPlayerById(searchPlayerId);
                            if (foundPlayerById != null)
                            {

Console.WriteLine($"{foundPlayerById.PlayerId}\t{foundPlayerById.PlayerName}\t{found
PlayerById.PlayerAge}");
                            }
                            else
                            {
                                Console.WriteLine("Player not found");
                            }
                            break;
                        case 4:
                            Console.Write("Enter Player Name: ");
                            string searchPlayerName = Console.ReadLine();
                            Player foundPlayerByName =
team.GetPlayerByName(searchPlayerName);
                            if (foundPlayerByName != null)
                            {

Console.WriteLine($"{foundPlayerByName.PlayerId}\t{foundPlayerByName.PlayerName}\t{f
oundPlayerByName.PlayerAge}");
                            }
                            else
                            {
                                Console.WriteLine("Player not found");
                            }
                            break;
                        case 5:
                            List<Player> allPlayers = team.GetAllPlayers();
                            foreach (var player in allPlayers)
                            {

Console.WriteLine($"{player.PlayerId}\t{player.PlayerName}\t{player.PlayerAge}");
                            }
                            break;
```

```
                        default:
                            Console.WriteLine("Invalid Choice");
                            break;
                    }
                }
                catch (Exception ex)
                {
                    Console.WriteLine("An error occured: " + ex.Message);
                }
                Console.Write("Do you want to continue? (yes/no): ");
                continueOp = Console.ReadLine();
            } while (continueOp.ToLower() == "yes");
        }
    }
}
```