Player.cs

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

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

ITeam.cs

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

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

OneDayTeam.cs

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

namespace ConAppCricketTeam
{
    public class OneDayTeam:ITeam
    {
        public static List<Player> oneDayTeam = new List<Player>();

        public OneDayTeam()
        {
            oneDayTeam.Capacity = 11;
        }
```

```
        public void Add(Player player)
        {
            if (oneDayTeam.Count < 11)
            {
                oneDayTeam.Add(player);
                Console.WriteLine($"Player {player.PlayerName} added to the team.");
            }
            else
            {
                Console.WriteLine("Team is full. Cannot add more players.");
            }
        }

        public void Remove(int playerId)
        {
            Player playerToRemove = oneDayTeam.FirstOrDefault(player =>
player.PlayerId == playerId);
            if (playerToRemove != null)
            {
                oneDayTeam.Remove(playerToRemove);
                Console.WriteLine($"Player {playerToRemove.PlayerName} removed from
the team.");
            }
            else
            {
                Console.WriteLine("Player not found.");
            }
        }

        public Player GetPlayerById(int playerId)
        {
            return oneDayTeam.FirstOrDefault(player => player.PlayerId == playerId);
        }

        public Player GetPlayerByName(string playerName)
        {
            return oneDayTeam.FirstOrDefault(player => player.PlayerName ==
playerName);
        }

        public List<Player> GetAllPlayers()
        {
            return oneDayTeam;
        }
    }
}
```

Program.cs

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

namespace ConAppCricketTeam
{
    internal class Program
    {
```

```
        static void Main(string[] args)
        {
            OneDayTeam team = new OneDayTeam();

            while (true)
            {
                Console.WriteLine("Enter 1: To Add Player 2: To Remove Player by Id
3. Get Player By Id 4. Get Player by Name 5. Get All Players:");
                int choice = Convert.ToInt32(Console.ReadLine());
                if (int.TryParse(Console.ReadLine(), out choice))
                {

                    switch (choice)
                    {
                        case 1:
                            Console.Write("Enter Player Id: ");
                            int id = Convert.ToInt32(Console.ReadLine());
                            Console.Write("Enter Player Name: ");
                            string name = Console.ReadLine();
                            Console.Write("Enter Player Age: ");
                            int age = Convert.ToInt32(Console.ReadLine());
                            Player newPlayer = new Player { PlayerId = id,
PlayerName = name, PlayerAge = age };
                            team.Add(newPlayer);
                            break;

                        case 2:
                            Console.Write("Enter Player Id to Remove: ");
                            int playerIdToRemove =
Convert.ToInt32(Console.ReadLine());
                            team.Remove(playerIdToRemove);
                            break;

                        case 3:
                            Console.Write("Enter Player Id to Get: ");
                            int playerIdToGet = Convert.ToInt32(Console.ReadLine());
                            Player playerById = team.GetPlayerById(playerIdToGet);
                            if (playerById != null)
                            {
                                Console.WriteLine($"Player Id:
{playerById.PlayerId}, Name: {playerById.PlayerName}, Age: {playerById.PlayerAge}");
                            }
                            else
                            {
                                Console.WriteLine("Player not found.");
                            }
                            break;

                        case 4:
                            Console.Write("Enter Player Name to Get: ");
                            string playerNameToGet = Console.ReadLine();
                            Player playerByName =
team.GetPlayerByName(playerNameToGet);
                            if (playerByName != null)
                            {
                                Console.WriteLine($"Player Id:
{playerByName.PlayerId}, Name: {playerByName.PlayerName}, Age:
{playerByName.PlayerAge}");
```

```
                            }
                            else
                            {
                                Console.WriteLine("Player not found.");
                            }
                            break;

                        case 5:
                            List<Player> allPlayers = team.GetAllPlayers();
                            foreach (Player player in allPlayers)
                            {
                                Console.WriteLine($"Player Id: {player.PlayerId},
Name: {player.PlayerName}, Age: {player.PlayerAge}");
                            }
                            break;

                        default:
                            Console.WriteLine("Invalid choice.");
                            break;
                    }
                }
                else
                {
                    Console.WriteLine("Invalid input. Please enter a number.");
                }
                Console.ReadKey();
            }

            }

        }
    }
```
