//Interface

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

namespace AssessmentOne
{
   public interface ITeam
   {
      void Add(Player player);
      void Remove(int id);
      Player GetPlayerById(int id);
      Player GetPlayerByName(string name);
      List<Player> GetAllPlayer();

   }
}
```

//Player Class

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

namespace AssessmentOne
{
   public class Player
   {
      public int PlayerId { get; set; }
      public string PlayerName { get; set; }
      public int PlayerAge { get; set; }
      public Player() { }
      public Player(int id,string name,int age)
      {
         PlayerId = id;
         PlayerName = name;
         PlayerAge = age;
      }
   }
}
```

//Team Class which implements Interface

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


namespace AssessmentOne
{
   public class OneDayTeam : ITeam
   {
      public static List<Player> oneDayTeam = new List<Player>();
      public OneDayTeam()
      {
         if (oneDayTeam.Count <= 11)
         {
            oneDayTeam = new List<Player>();
         }
         else
         {
            Console.WriteLine("The team can only have 11 Members");
         }
      }
      public void Add(Player player)
      {
```

```
            oneDayTeam.Add(player);
            Console.WriteLine("Player Added Successfully");
        }
        public void Remove(int id)
        {
            Player playerRemoval = oneDayTeam.Find(oneDayTeam => oneDayTeam.PlayerId == id);
            if (playerRemoval != null)
            {
                oneDayTeam.Remove(playerRemoval);
                Console.WriteLine("Player Removed Successfully");
            }
            else
                Console.WriteLine("There is no such member in the Team");
        }
        public Player GetPlayerById(int id)
        {
            Player playerById = oneDayTeam.Find(oneDayTeam => oneDayTeam.PlayerId == id);
            return playerById;
        }
        public Player GetPlayerByName(string name)
        {
            Player playerByName = oneDayTeam.Find(oneDayTeam =>
oneDayTeam.PlayerName.Equals(name));
            return playerByName;
        }
        public List<Player> GetAllPlayer()
```

```
        {
            return oneDayTeam;
        }
    }
}
```

//Main Class

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

namespace AssessmentOne
{
    internal class Program
    {
        static void Main(string[] args)
        {
            try
            {
                OneDayTeam team = new OneDayTeam();
                again:
```

```
        Console.WriteLine("The Available Functions to Perform\n\t1. Add Player\n\t2. Remove Player by Id\n\t3. Get Player By Id\n\t4. Get Player By Name\n\t5. Get All Players");

        Console.Write("Enter the Number : ");

        switch (int.Parse(Console.ReadLine()))

        {

          case 1:

            {

              Console.WriteLine("\nAdding Player to the Team....");

              Player player = new Player();

              Console.WriteLine("Enter Player Id");

              player.PlayerId = int.Parse(Console.ReadLine());

              Console.WriteLine("Enter Player Name");

              player.PlayerName = Console.ReadLine();

              Console.WriteLine("Enter Player Age");

              player.PlayerAge = int.Parse(Console.ReadLine());

              team.Add(player);

              break;

            }

          case 2:

            {

              Console.WriteLine("\nEnter Player Id to Remove");

              int id = int.Parse(Console.ReadLine());

              team.Remove(id);

              break;

            }

          case 3:
```

```
    {
        Console.WriteLine("\nEnter Player Id to Display");
        int id = int.Parse(Console.ReadLine());
        Player player = team.GetPlayerById(id);
        Console.WriteLine("\nPlayer Id\tPlayer Name\t Player Age");
        Console.WriteLine($"{player.PlayerId}\t\t{player.PlayerName}\t\t{player.PlayerAge}");
        break;
    }
case 4:
    {
        Console.WriteLine("\nEnter Player Name to Display");
        string name = Console.ReadLine();
        Player player = team.GetPlayerByName(name);
        Console.WriteLine("\nPlayer Id\tPlayer Name\t Player Age");
        Console.WriteLine($"{player.PlayerId}\t\t{player.PlayerName}\t\t{player.PlayerAge}");
        break;
    }
case 5:
    {
        Console.WriteLine("\nList Of All Players....");
        List<Player> allPlayers = team.GetAllPlayer();
        Console.WriteLine("\nPlayer Id\tPlayer Name\t Player Age");
        foreach (Player player in allPlayers)
        {
            Console.WriteLine($"{player.PlayerId}\t\t{player.PlayerName}\t\t{player.PlayerAge}");
```

```
                }
                break;
            }
        default:
            {
                Console.WriteLine("\nYou have Entered the Wrong Number!!!\n Choose the Right
One\n");
                goto again;
            }
        }
        Console.WriteLine("\nPress 1 to Continue....And Others to Exit.");
        if (int.Parse(Console.ReadLine()) == 1)
            goto again;
    }
    catch(Exception ec)
    {
        Console.WriteLine("\nNow The Error has Occured\n");
        Console.WriteLine(ec.Message);
    }
    finally
    {
        Console.ReadKey();
    }
  }
 }
}
```