***Github link:*** <https://github.com/sundershyam04/Phase2-End-Project.git>

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

/\*

FastPace Cricket Academy has decided to create a solution to maintain information about the teams’ players for one day game

\*/

namespace Player\_and\_Team\_Requirements

{

internal class Program

{

static void Main(string[] args)

{

// menu

OneDayTeam oneDay = new OneDayTeam();

string need;

do

{

Console.WriteLine("Enter\n1:To Add Player\n2:To Remove Player by Id\n3.Get Player By Id\n4.Get Player by Name\n5.Get All Players:");

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

switch (choice)

{

case 1:

Player p = new Player();

Console.Write("Enter Player Id : ");

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

Console.Write("Enter Player Name : ");

p.PlayerName = Console.ReadLine();

Console.Write("Enter Player Age : ");

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

oneDay.Add(p);

break;

case 2:

Console.Write("Enter Player Id to Remove : ");

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

oneDay.Remove(playerId);

break;

case 3:

Console.Write("Enter Player Id : ");

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

Player found1=oneDay.GetPlayerById(pId);

Console.WriteLine(found1.PlayerId + " " + found1.PlayerName + " " + found1.PlayerAge);

break;

case 4:

Console.Write("Enter Player Name : ");

string pName = Console.ReadLine();

Player found2 = oneDay.GetPlayerByName(pName);

Console.WriteLine(found2.PlayerId + " " + found2.PlayerName + " " + found2.PlayerAge);

break;

case 5:

List<Player> teamlist=oneDay.GetAllPlayers();

foreach (var item in teamlist)

{

Console.WriteLine(item.PlayerId + " " + item.PlayerName + " " + item.PlayerAge);

}

break;

default:

Console.WriteLine("Enter choice from 1 to 5");

break;

}

Console.WriteLine("===================================================");

Console.Write("Do you want to continue( yes / no ) : ");

need = Console.ReadLine();

Console.WriteLine("===================================================");

} while (need.Equals("yes"));

Console.Read();

}

}

}

**Onedayteam.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Player\_and\_Team\_Requirements

{

public class OneDayTeam : ITeam

{

public static List<Player> oneDayTeam = new List<Player>();

public OneDayTeam() //ctor to set capacity

{

oneDayTeam.Capacity = 11;

}

public void Add(Player player)

{

oneDayTeam.Add(player);

Console.WriteLine("Player is added successfully");

}

public void Remove(int playerId)

{

Player found=oneDayTeam.Find ( player => player.PlayerId.Equals(playerId));

if(found != null)

{

oneDayTeam.Remove(found);

Console.WriteLine("Player is removed successfully");

}

else

{

Console.WriteLine("Player not found in onedayteam");

}

}

public Player GetPlayerById(int playerId)

{

Player found = oneDayTeam.Find(player => player.PlayerId.Equals(playerId));

//Console.WriteLine(found.PlayerId + " " + found.PlayerName + " " + found.PlayerAge);

if (found != null)

{

return found;

}

else

{

Console.WriteLine("Player not found in onedayteam");

return default;

}

}

public Player GetPlayerByName(string playerName)

{

Player found = oneDayTeam.Find(player => player.PlayerName.Equals(playerName));

if (found != null)

return found;

else

{

Console.WriteLine("Player not found in onedayteam");

return default;

}

}

public List<Player> GetAllPlayers()

{

return oneDayTeam;

}

}

}

**Player.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Player\_and\_Team\_Requirements

{

public class Player

{

// ID , NAME, AGE

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 Player\_and\_Team\_Requirements

{

interface ITeam

{

void Add(Player player);

void Remove(int playerId);

Player GetPlayerById(int playerId);

Player GetPlayerByName(string playerName);

List<Player> GetAllPlayers();

}}