Source Code:

Program.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PlayerandTeam

{

internal class Program

{

static void Main(string[] args)

{

OneDayTeam odTeam = new OneDayTeam();

string Cont;

try

{

do

{

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");

/\* Console.WriteLine("2: To Remove Player by Id");

Console.WriteLine("3: Get Player By Id");

Console.WriteLine("4: Get Player by Name");

Console.WriteLine("5: Get All Players\n");\*/

Console.Write("Enter your choice: ");

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

switch (choice)

{

case 1:

Player player = new Player();

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

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

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

player.PlayerName = Console.ReadLine();

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

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

odTeam.Add(player);

break;

case 2:

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

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

odTeam.Remove(playerId);

break;

case 3:

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

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

Player retrievedPlayer = odTeam.GetPlayerById(playerId);

if (retrievedPlayer != null)

{

Console.WriteLine($"Player Id: {retrievedPlayer.PlayerId}, Name: {retrievedPlayer.PlayerName}, Age: {retrievedPlayer.PlayerAge}");

}

else

{

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

}

break;

case 4:

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

string playerName = Console.ReadLine();

retrievedPlayer = odTeam.GetPlayerByName(playerName);

if (retrievedPlayer != null)

{

Console.WriteLine($"Player Id: {retrievedPlayer.PlayerId}, Name: {retrievedPlayer.PlayerName}, Age: {retrievedPlayer.PlayerAge}");

}

else

{

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

}

break;

case 5:

List<Player> players = odTeam.GetAllPlayers();

foreach (Player p in players)

{

Console.WriteLine($"Player Id: {p.PlayerId}, Name: {p.PlayerName}, Age: {p.PlayerAge}");

}

break;

default:

Console.WriteLine("Invalid choice. Please try again.");

break;

}

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

Cont = Console.ReadLine().ToLower();

if (Cont == "no")

{

return;

}

} while (Cont == "yes");

}

catch (Exception ex)

{

Console.WriteLine("Error!!!!! " + ex.Message);

}

finally

{

Console.ReadKey();

}

}

}

}

Player.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PlayerandTeam

{

internal 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 PlayerandTeam

{

internal interface ITeam

{

void Add(Player player);

void Remove(int playerId);

Player GetPlayerById(int playerId);

Player GetPlayerByName(string playerName);

List<Player> GetAllPlayers();

}

}

OndayTeam.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PlayerandTeam

{

internal class OneDayTeam:ITeam

{

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

public OneDayTeam()

{

oneDayTeam.Capacity = 11;

}

public void Add(Player player)

{

if (oneDayTeam.Count < oneDayTeam.Capacity)

{

oneDayTeam.Add(player);

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

}

else

{

Console.WriteLine("Cannot add ....Team is full!!!!!!");

}

}

public void Remove(int playerId)

{

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

if (playerToRemove != null)

{

oneDayTeam.Remove(playerToRemove);

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

}

else

{

Console.WriteLine("Player not found!!!!");

}

}

public Player GetPlayerById(int playerId)

{

return oneDayTeam.Find(player => player.PlayerId == playerId);

}

public Player GetPlayerByName(string playerName)

{

return oneDayTeam.Find(player => player.PlayerName == playerName);

}

public List<Player> GetAllPlayers()

{

return oneDayTeam;

}

}

}