using System;

using System.Collections.Generic;

using System.IO;

class Player

{

public int ID { get; set; }

public string Name { get; set; }

public int Age { get; set; }

public string Position { get; set; }

}

class FileHandler

{

private const string FilePath = "players.txt";

public static List<Player> ReadPlayersFromFile()

{

List<Player> players = new List<Player>();

if (File.Exists(FilePath))

{

string[] lines = File.ReadAllLines(FilePath);

foreach (string line in lines)

{

string[] data = line.Split(',');

Player player = new Player

{

ID = int.Parse(data[0]),

Name = data[1],

Age = int.Parse(data[2]),

Position = data[3]

};

players.Add(player);

}

}

return players;

}

public static void WritePlayersToFile(List<Player> players)

{

using (StreamWriter writer = new StreamWriter(FilePath))

{

foreach (Player player in players)

{

writer.WriteLine($"{player.ID},{player.Name},{player.Age},{player.Position}");

}

}

}

}

class PlayerManager

{

private static List<Player> players;

static PlayerManager()

{

players = FileHandler.ReadPlayersFromFile();

}

public static void AddPlayer(Player player)

{

players.Add(player);

FileHandler.WritePlayersToFile(players);

}

public static void UpdatePlayer(Player updatedPlayer)

{

Player existingPlayer = players.Find(p => p.ID == updatedPlayer.ID);

if (existingPlayer != null)

{

existingPlayer.Name = updatedPlayer.Name;

existingPlayer.Age = updatedPlayer.Age;

existingPlayer.Position = updatedPlayer.Position;

FileHandler.WritePlayersToFile(players);

}

else

{

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

}

}

public static void DeletePlayer(int playerID)

{

Player playerToDelete = players.Find(p => p.ID == playerID);

if (playerToDelete != null)

{

players.Remove(playerToDelete);

FileHandler.WritePlayersToFile(players);

}

else

{

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

}

}

public static void DisplayPlayers()

{

foreach (Player player in players)

{

Console.WriteLine($"ID: {player.ID}, Name: {player.Name}, Age: {player.Age}, Position: {player.Position}");

}

}

}

class Program

{

static void Main()

{

int choice;

do

{

Console.WriteLine("1. Add Player");

Console.WriteLine("2. Update Player");

Console.WriteLine("3. Delete Player");

Console.WriteLine("4. Display Players");

Console.WriteLine("5. Exit");

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

int.TryParse(Console.ReadLine(), out choice);

switch (choice)

{

case 1:

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

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

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

string name = Console.ReadLine();

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

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

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

string position = Console.ReadLine();

PlayerManager.AddPlayer(new Player { ID = id, Name = name, Age = age, Position = position });

break;

case 2:

Console.Write("Enter Player ID to update: ");

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

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

string updateName = Console.ReadLine();

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

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

Console.Write("Enter Updated Player Position: ");

string updatePosition = Console.ReadLine();

PlayerManager.UpdatePlayer(new Player { ID = updateID, Name = updateName, Age = updateAge, Position = updatePosition });

break;

case 3:

Console.Write("Enter Player ID to delete: ");

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

PlayerManager.DeletePlayer(deleteID);

break;

case 4:

PlayerManager.DisplayPlayers();

break;

case 5:

Console.WriteLine("Exiting program.");

break;

default:

Console.WriteLine("Invalid choice. Try again.");

break;

}

} while (choice != 5);

}

}