Name: Muhammad Anas Baig

Enrollment No.: 01-134152-037

Section: BS(CS)-5A

ASSIGNMENT # 1

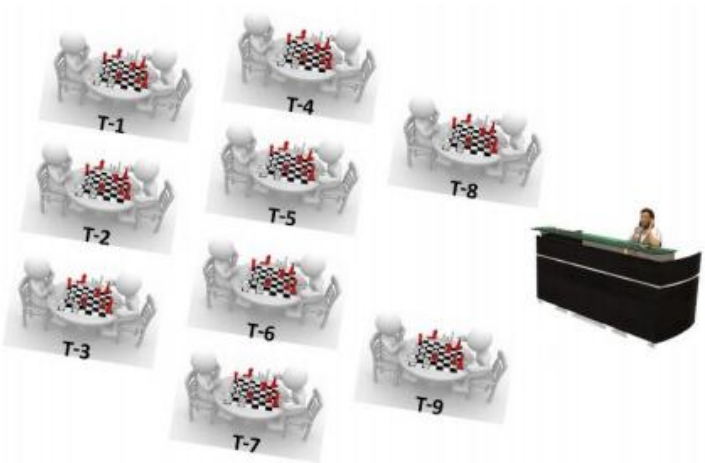
Visual Programming

BS(CS) – 5A

October 8, 2017

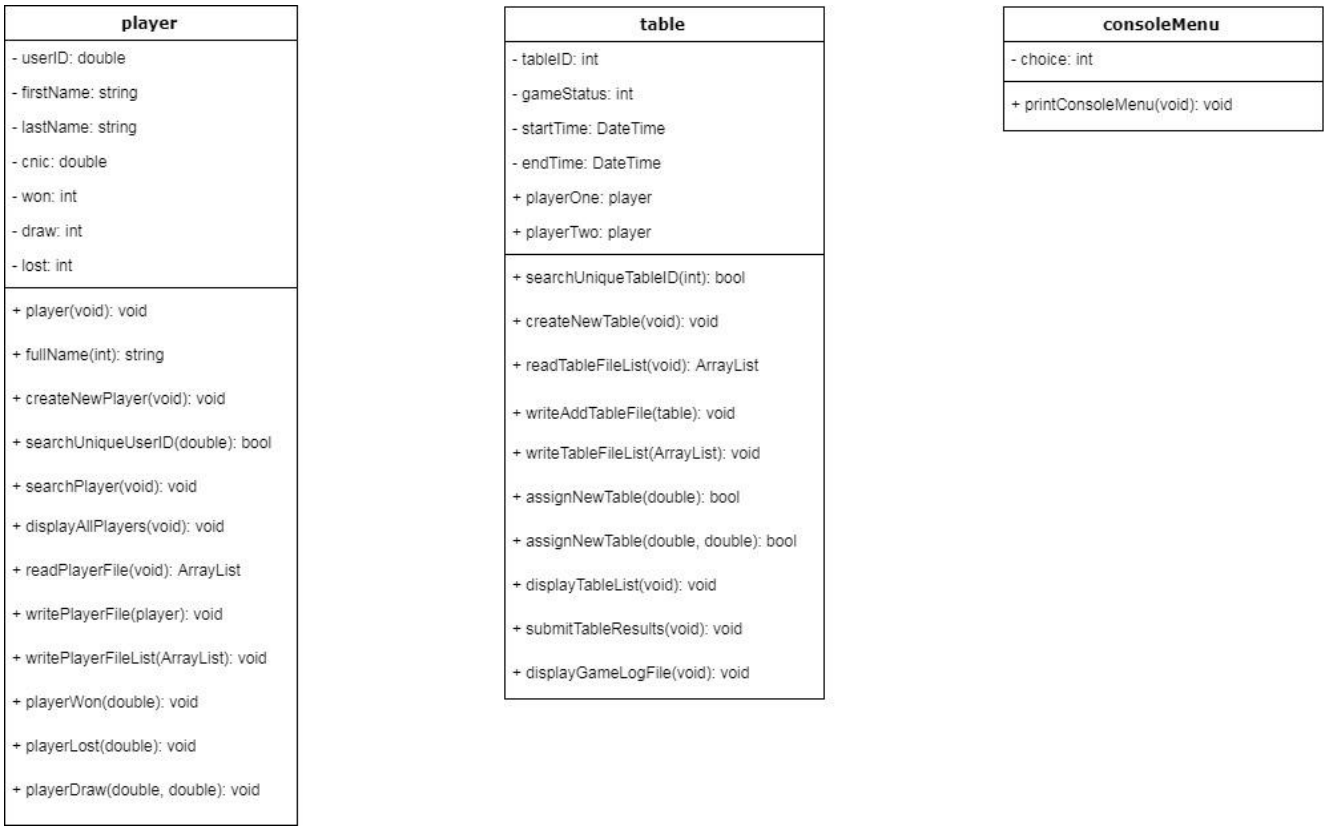
C^3 (Chess Challenge Consortium)

C Cube (C^3) is Chess Challenge Consortium where players reserve their tables for chess games with opponents or without opponents. In case of a player arrives without opponent, the manager on reception desk helps him to find the opponent. Game is played and results of the game is stored by the manager along with the information of players, table id and date time. There are total nine tables available at a time. In case all tables are busy, the player has to wait till the result of ongoing game is concluded. Your task is to design a system which will store the information of players, their reservations and results of the game they played with the opponent. Your system should have at least the following features:



1. System should allow user to register the information of player such as, Name CNIC. Only if player is not already registered.
2. In case of 2 players arrive to play, the table should be assigned immediately.
3. In case player single player is arrived, your system should reserve table for him, or should assigned already single player waiting for game.
4. In case 2 players arrive to play and table is not available then the reserved table of single player should be assigned to them for immediate start of game.
5. Result of the game should be store along with the table id, date and time. Result of game should be saved with player's statistics.
6. System should show the current status of all the tables.
7. System should be able to search a player's information by any mean along with total games he played and total wins/draw/lose.
8. System should store data in file(s).

UML CLASS DIAGRAM:



Source Code:

consoleMenu.cs File:

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

namespace Assignment1
{
    class consoleMenu
    {
        public void printConsoleMenu() //method to print console menu
        {
            int choice; //variable to store user choices
            do //loop to display after every operation
            {
                Console.Write("-------------------------------------------------------------------------------");
                Console.WriteLine( "MENU:" );
                Console.Write("-------------------------------------------------------------------------------");
                Console.WriteLine("1. Register New Player.");
                Console.WriteLine("2. Search Player.");
                Console.WriteLine("3. Display All Players Statistics.");
                Console.WriteLine("4. Assign Table to One Player.");
                Console.WriteLine("5. Assign Table to Two Players.");
                Console.WriteLine("6. Submit Table Results.");
                Console.WriteLine("7. Display All Tables Status.");
                Console.WriteLine("8. Add New Table to System.");
```

```
                Console.WriteLine("9. Display Game Log History.");
                Console.WriteLine("10. Exit." );
                Console.Write("-------------------------------------------------------------------------------");
                Console.WriteLine( "SELECT DESIRED OPERATION:" );
                Console.Write("-------------------------------------------------------------------------------");
                choice = int.Parse(Console.ReadLine());

                if (choice == 1)
                {
                    player p = new player();
                    p.createNewPlayer(); //to register new player in the system
                }
                else if (choice == 2)
                {
                    player p = new player();
                    p.searchPlayer(); //to search player in the system
                }
                else if (choice == 3)
                {
                    player p = new player();
                    p.displayAllPlayers(); //to display all players statistics
                }
                else if (choice == 4)
                {
                    table t = new table();
                    Console.Write("-------------------------------------------------------------------------------");
                    Console.WriteLine("ASSIGN TABLE TO ONE PLAYER:");
                    Console.Write("-------------------------------------------------------------------------------");
                    player p = new player();
                    int userID;
                    do //checks either the userID assigning to the table exists or not
                    {
                        Console.WriteLine("Enter Player User-ID:");
                        userID = int.Parse(Console.ReadLine());
                        if (!p.searchUniqueUserID(userID)) //checks either the user ID assigning to the table exists or not
                        {
                            Console.WriteLine("ERROR! User-ID not found, please try again.");
                        }
                    }
                    while (!p.searchUniqueUserID(userID)); //checks either the user ID assigning to the table exists or not
                    if (!t.assignNewTable(userID)) //checks that all tables are filled or not, if table is partially filled then it will be assigned to that player
                    {
                        Console.WriteLine("PLEASE WAIT! All tables are filled.");
                    }
                }
                else if ( choice == 5 )
                {
                    table t = new table();
                    Console.Write("-------------------------------------------------------------------------------");
                    Console.WriteLine("ASSIGN TABLE TO TWO PLAYERS:");
                    Console.Write("-------------------------------------------------------------------------------");
                    player p = new player();
                    int userOneID;
                    int userTwoID;
```

```
                    do //checks either the userID assigning to the table exists or not
                    {
                        Console.WriteLine("Enter Player-1 User-ID:");
                        userOneID = int.Parse(Console.ReadLine());
                        Console.WriteLine("Enter Player-2 User-ID:");
                        userTwoID = int.Parse(Console.ReadLine());
                        if (!p.searchUniqueUserID(userOneID) && !p.searchUniqueUserID(userTwoID)) //checks either the userID assigning to the table exists or not
                        {
                            Console.WriteLine("ERROR! User-ID not found, please try again.");
                        }
                    }
                    while (!p.searchUniqueUserID(userOneID) && !p.searchUniqueUserID(userTwoID)); //checks either the userID assigning to the table exists or not
                    if (!t.assignNewTable(userOneID, userTwoID)) //checks that all tables are filled or not and then assign table to players
                    {
                        Console.WriteLine("PLEASE WAIT! All tables are filled.");
                    }
                }
                else if (choice == 6)
                {
                    table t = new table();
                        t.submitTableResults(); //to submit results of the table
                    }
                    else if (choice == 7)
                    {
                        table t = new table();
                        t.displayTableList(); //to display all tables status
                    }
                    else if (choice == 8)
                    {
                        table t = new table();
                        t.createNewTable(); //to add new table to system
                    }
                    else if (choice == 9)
                    {
                        table t = new table();
                        t.displayGameLogFile(); //to display game history
                    }
                    else
                    {
                        Console.WriteLine("ERROR! Invalid Input.");
                    }
                }
                while (choice != 10); //loop to display after every operation
            }
        }
    }
```

player.cs File:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Collections;

namespace Assignment1
{
```

```
    class player
    {
        double userID;
        string firstName;
        string lastName;
        double cnic;
        int won;
        int draw;
        int lost;

        public double userIDProperty
        {
            get{    return userID;  }
            set{    userID = value; }
        }
        public string firstNameProperty
        {
            get{    return firstName;   }
            set{    firstName = value;  }
        }
        public string lastNameProperty
        {
            get{    return lastName;    }
            set{    lastName = value;   }
        }
        public double cnicProperty
        {
            get{    return cnic;    }
            set{    cnic = value;   }
        }
        public int wonProperty
        {
            get{    return won;     }
            set{    won = value;    }
        }
        public int drawProperty
        {
            get {   return draw;    }
            set {   draw = value;   }
        }
        public int lostProperty
        {
            get{    return lost;    }
            set{    lost = value;   }
        }
        public player()
        {
            won = 0;
            draw = 0;
            lost = 0;
        }
        public string fullName(double id) //function that returns concatenation of firstname and lastname
        {
            ArrayList playerList = new ArrayList(); //player list
            playerList = readPlayerFile(); //reading file to list

            for (int i = 0; i < playerList.Count; i++) //checks each user in list
            {
                if ((playerList[i] as player).userID == id) //checks for the required user
                {
                    return ((playerList[i] as player).firstName + " " + (playerList[i] as player).lastName); //returns concatenation of firstname and lastname
                }
            }
```

```
            return (""); //if user not found
        }
        public void createNewPlayer()
        {
            Console.Write("-------------------------------------------------------------------------");
            Console.WriteLine( "CREATE NEW PLAYER:" );
            Console.Write("-------------------------------------------------------------------------");
            do //loop that prompts user to enter new userID if the typed one is already taken
            {
                Console.WriteLine("Enter New User-ID:");
                this.userIDProperty = double.Parse(Console.ReadLine());
                if ( searchUniqueUserID(this.userIDProperty) )
                {
                    Console.WriteLine( "ERROR! User-ID already assigned, kindly choose another.");
                }
            }
            while (searchUniqueUserID(this.userIDProperty)); //loop that prompts user to enter new userID if the typed one is already taken
            Console.WriteLine("Enter First Name:");
            this.firstNameProperty = Console.ReadLine();
            Console.WriteLine("Enter Last Name:");
            this.lastNameProperty = Console.ReadLine();
            Console.WriteLine("Enter CNIC:");
            this.cnicProperty = double.Parse(Console.ReadLine());
            Console.WriteLine("Player Succesfully Registered");

            writePlayerFile(this); //appends the new player in player file
        }
        public bool searchUniqueUserID(double id) //before creating new user this methods checks either the userID is already taken or not
        {
            ArrayList playerList = new ArrayList(); //player list
            playerList = readPlayerFile(); //reading file to list

            for (int i = 0; i < playerList.Count; i++)  //checks each user in list
            {
                if ((playerList[i] as player).userID == id) //checks either userID is already taken or not
                {
                    return true;
                }
            }
            return false;
        }
        public void searchPlayer() //to search a specific user in the system

        {
            ArrayList playerList = new ArrayList(); //player list
            playerList = readPlayerFile(); //reading file to list

            Console.Write( "-------------------------------------------------------------------------" );
            Console.WriteLine("SEARCH PLAYER:");
            Console.Write("-------------------------------------------------------------------------");
            Console.WriteLine("Enter Desired Operation:");
            Console.WriteLine("1. Search by User-ID.");
            Console.WriteLine("2. Search by Name.");
            Console.WriteLine("3. Search by CNIC.");
            int choice = int.Parse(Console.ReadLine());

```

```
                    if(choice == 1) //search by userID
                    {
                        Console.WriteLine();
                        Console.WriteLine("Enter Search Player User-ID.");
                        double ID = double.Parse(Console.ReadLine());

                        for (int i = 0; i < playerList.Count; i++) //checks each user in list
                        {
                            if ((playerList[i] as player).userID == ID) //checks for the requied userID
                            {
                                Console.WriteLine("----------------------");
                                Console.WriteLine("Player - " + (i + 1) + " Statistics:");
                                Console.WriteLine("----------------------");
                                Console.WriteLine("User ID:     " + (playerList[i] as player).userID);
                                Console.WriteLine("Name:        " + (playerList[i] as player).firstName + " " + (playerList[i] as player).lastName);
                                Console.WriteLine("CNIC:        " + (playerList[i] as player).cnic);
                                int gamesPlayed = ((playerList[i] as player).won + (playerList[i] as player).draw + (playerList[i] as player).lost); //number of games played
                                Console.WriteLine("SCORE:");
                                Console.WriteLine("------");
                                Console.WriteLine("Won:         " + (playerList[i] as player).won);
                                Console.WriteLine("Draw:        " + (playerList[i] as player).draw);
                                Console.WriteLine("Lost:        " + (playerList[i] as player).lost);
                                Console.WriteLine("----------------------");
                                Console.WriteLine("Total Games: " + gamesPlayed);
                                Console.WriteLine("----------------------");
                            }
                        }
                    }
                    else if(choice == 2) //search by name
                    {
                        Console.WriteLine("Enter Search Player Full Name:");
                        string name = Console.ReadLine();

                        for (int i = 0; i < playerList.Count; i++) //checks each user in list
                        {
                            if (((playerList[i] as player).firstName + " " + (playerList[i] as player).lastName) == name) //checks for the requied name(firstName + lastName)
                            {
                                Console.WriteLine("----------------------");
                                Console.WriteLine("Player - " + (i + 1) + " Statistics:");
                                Console.WriteLine("----------------------");
                                Console.WriteLine("User ID:     " + (playerList[i] as player).userID);
                                Console.WriteLine("Name:        " + (playerList[i] as player).firstName + " " + (playerList[i] as player).lastName);
                                Console.WriteLine("CNIC:        " + (playerList[i] as player).cnic);
                                int gamesPlayed = ((playerList[i] as player).won + (playerList[i] as player).draw + (playerList[i] as player).lost); //number of games played
                                Console.WriteLine( "SCORE:");
                                Console.WriteLine("------");
```

```
                                Console.WriteLine("Won:         " + (playerList[i] as player).won);
                                Console.WriteLine("Draw:        " + (playerList[i] as player).draw);
                                Console.WriteLine("Lost:        " + (playerList[i] as player).lost);
                                Console.WriteLine("----------------------" );
                                Console.WriteLine("Total Games: " + gamesPlayed);
                                Console.WriteLine("----------------------");
                            }
                        }
                    }
                    else if(choice == 3) //search by cnic
                    {
                        Console.WriteLine("Enter Search Player CNIC.");
                        double num = double.Parse(Console.ReadLine());

                        for (int i = 0; i < playerList.Count; i++) //checks each user in list
                        {
                            if ((playerList[i] as player).cnic == num) //check for the required cnic
                            {
                                Console.WriteLine("----------------------");
                                Console.WriteLine("Player - " + (i + 1) + " Statistics:");
                                Console.WriteLine("----------------------" );
                                Console.WriteLine("User ID:     " + (playerList[i] as player).userID);
                                Console.WriteLine("Name:        " + (playerList[i] as player).firstName + " " + (playerList[i] as player).lastName);
                                Console.WriteLine("CNIC:        " + (playerList[i] as player).cnic);
                                int gamesPlayed = (( playerList[i] as player).won + (playerList[i] as player).draw + (playerList[i] as player).lost); //number of games played
                                Console.WriteLine("SCORE:");
                                Console.WriteLine("------");
                                Console.WriteLine("Won:         " + (playerList[i] as player).won);
                                Console.WriteLine("Draw:        " + (playerList[i] as player).draw);
                                Console.WriteLine("Lost:        " + (playerList[i] as player).lost);
                                Console.WriteLine("----------------------");
                                Console.WriteLine("Total Games: " + gamesPlayed);
                                Console.WriteLine("----------------------");
                            }
                        }
                    }
                    else
                    {
                        Console.WriteLine("ERROR! Invalid Input.");
                    }
                }
                public void displayAllPlayers() //to display all players statistics

                {
                    ArrayList playerList = new ArrayList(); //player list
                    playerList = readPlayerFile(); //reading file to list
                    Console.Write("------------------------------------------------------------------------------");
                    Console.WriteLine( "DISPLAY ALL PLAYERS STATISTICS:");
                    Console.Write("------------------------------------------------------------------------------");

```

```
                    for ( int i = 0; i < playerList.Count; i++ ) //checks each user in list
                    {
                        Console.WriteLine();
                        Console.WriteLine("----------------------");
                        Console.WriteLine("Player - " + (i+1) + " Statistics:");
                        Console.WriteLine("----------------------" );
                        Console.WriteLine("User ID:     " + (playerList[i] as player).userID);
                        Console.WriteLine("Name:        " + (playerList[i] as player).firstName + " " + (playerList[i] as player).lastName);
                        Console.WriteLine("CNIC:        " + (playerList[i] as player).cnic);
                        int gamesPlayed = ((playerList[i] as player).won + (playerList[i] as player).draw + (playerList[i] as player).lost); //number of games played
                        Console.WriteLine("SCORE:");
                        Console.WriteLine("------");
                        Console.WriteLine("Won:         " + (playerList[i] as player).won);
                        Console.WriteLine("Draw:        " + (playerList[i] as player).draw);
                        Console.WriteLine("Lost:        " + (playerList[i] as player).lost);
                        Console.WriteLine("----------------------");
                        Console.WriteLine("Total Games: " + gamesPlayed);
                        Console.WriteLine("----------------------");
                    }
                }
                public ArrayList readPlayerFile() //reads player file and returns ArrayList which containts all players data
                {
                    ArrayList playerList = new ArrayList(); //to display all players statistics
                    StreamReader readPlayerFile = new StreamReader("Players.txt"); //reading file to list
                    player p;

                    while (!readPlayerFile.EndOfStream) //reading file till end
                    {
                        p = new player();
                        p.userID = double.Parse(readPlayerFile.ReadLine());
                        p.firstName = readPlayerFile.ReadLine();
                        p.lastName = readPlayerFile.ReadLine();
                        p.cnic = double.Parse(readPlayerFile.ReadLine());
                        p.won = int.Parse(readPlayerFile.ReadLine());
                        p.draw = int.Parse(readPlayerFile.ReadLine());
                        p.lost = int.Parse(readPlayerFile.ReadLine());
                        playerList.Add( p );
                    }
                    readPlayerFile.Close();
                    return playerList; //returning ArrayList which contains all players data
                }
                public void writePlayerFile(player p) //to add new player to the file
                {
                    StreamWriter writePlayerFile = new StreamWriter("Players.txt", true); //appending the player file

                    writePlayerFile.WriteLine( p.userID );
                    writePlayerFile.WriteLine( p.firstName );
                    writePlayerFile.WriteLine( p.lastName );
                    writePlayerFile.WriteLine( p.cnic );
                    writePlayerFile.WriteLine( p.won );
                    writePlayerFile.WriteLine( p.draw );
                    writePlayerFile.WriteLine( p.lost );
```

```
275.
276.                    writePlayerFile.Close();
277.                }
278.                public void writePlayerFileList(ArrayList playerList) //to write mod
    ified/updated data to file -> modify/update -> game Win/Loss
279.                {
280.                    StreamWriter writePlayerFile = new StreamWriter("Players.txt");
    //not opened in appended mode because all modified/updated data is to write to file
     -> modify/update -> game Win/Loss
281.
282.                    for (int i = 0; i < playerList.Count; i++) //checks each user in
     list
283.                    {
284.                        writePlayerFile.WriteLine((playerList[i] as player ).userIDP
    roperty);
285.                        writePlayerFile.WriteLine((playerList[i] as player).firstNam
    eProperty);
286.                        writePlayerFile.WriteLine((playerList[i] as player).lastName
    Property);
287.                        writePlayerFile.WriteLine((playerList[i] as player).cnicProp
    erty);
288.                        writePlayerFile.WriteLine((playerList[i] as player).wonPrope
    rty);
289.                        writePlayerFile.WriteLine((playerList[i] as player).drawProp
    erty);
290.                        writePlayerFile.WriteLine((playerList[i] as player).lostProp
    erty);
291.                    }
292.                    writePlayerFile.Close();
293.                }
294.                public void playerWon(double id) //takes userID and updates user's w
    on games
295.                {
296.                    ArrayList playerList = new ArrayList(); //player list
297.                    playerList = readPlayerFile(); //reading file to list
298.
299.                    for (int i = 0; i < playerList.Count; i++) //checks each user in
     list
300.                    {
301.                        if ((playerList[i] as player).userIDProperty == id)
302.                        {
303.                            (playerList[i] as player).wonProperty = 1 + (playerList[
    i] as player).wonProperty; //increments in user's won games
304.                        }
305.                    }
306.                    writePlayerFileList(playerList);
307.                }
308.                public void playerLost(double id) //takes userID and updates user's
    won games
309.                {
310.                    ArrayList playerList = new ArrayList(); //player list
311.                    playerList = readPlayerFile(); //reading file to list
312.
313.                    for (int i = 0; i < playerList.Count; i++) //checks each user in
     list
314.                    {
315.                        if ((playerList[i] as player).userIDProperty == id) //checks
     if required user is found
316.                        {
317.                            (playerList[i] as player).lostProperty = 1 + (playerList
    [i] as player).lostProperty; //increments in user's won games
318.                        }
319.                    }
320.                    writePlayerFileList(playerList);
321.                }
```

```
322.                public void playerDraw(double id1, double id2) //takes userIDs of pl
    ayer1 and player2 and updates both user's draw games
323.                {
324.                    ArrayList playerList = new ArrayList(); //player list
325.                    playerList = readPlayerFile(); //reading file to list
326.
327.                    for (int i = 0; i < playerList.Count; i++) //checks each user in
     list
328.                    {
329.                        if ((playerList[i] as player).userIDProperty == id1)
330.                        {
331.                            (playerList[i] as player).drawProperty = 1 + (playerList
    [i] as player).drawProperty; //increments player's draw games
332.                        }
333.                        if ((playerList[i] as player).userIDProperty == id2)
334.                        {
335.                            (playerList[i] as player).drawProperty = 1 + (playerList
    [i] as player).drawProperty; //increments player's draw games
336.                        }
337.                    }
338.                    writePlayerFileList( playerList );
339.                }
340.            }
341.        }
```

table.cs File:

```
1.  using System;
2.  using System.Collections.Generic;
3.  using System.Linq;
4.  using System.Text;
5.  using System.IO;
6.  using System.Collections;
7.
8.  namespace Assignment1
9.  {
10.     class table
11.     {
12.         int tableID; //stores tableID
13.         int gameStatus; //stores game status i.e. 0->Empty Table, 1-
    >One Player Assigned, 2->Two Players Assigned
14.         DateTime startTime;
15.         DateTime endTime;
16.         public player playerOne = new player(); //player1 on table
17.         public player playerTwo = new player(); //player2 on table
18.
19.         public int tableIDProperty
20.         {
21.             get { return tableID; }
22.             set { tableID = value; }
23.         }
24.         public int gameStatusProperty
25.         {
26.             get { return gameStatus; }
27.             set { gameStatus = value; }
28.         }
29.         public double playerOneProperty
30.         {
31.             get { return playerOne.userIDProperty; }
32.             set { playerOne.userIDProperty = value; }
33.         }
34.         public double playerTwoProperty
35.         {
36.             get { return playerTwo.userIDProperty; }
```

```
            set { playerTwo.userIDProperty = value; }
        }
        public DateTime startTimeProperty
        {
            get { return startTime; }
            set { startTime = value; }
        }
        public DateTime endTimeProperty
        {
            get { return endTime; }
            set { endTime = value; }
        }
        public bool searchUniqueTableID(int id) //while creating new table checks either the tableID in already assigned or not
        {
            ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
            tableList = readTableFileList(); //reads tables from file to list

            for (int i = 0; i < tableList.Count; i++) //checks each table
            {
                if ((tableList[i] as table).tableID == id) //checks for the required tableID
                {
                    return true;
                }
            }
            return false;
        }
        public void createNewTable() //to add new table to system
        {
            Console.Write("-------------------------------------------------------------------------");
            Console.WriteLine("ADD NEW TABLE TO SYSTEM:");
            Console.Write("-------------------------------------------------------------------------");

            do
            {
                Console.WriteLine("Enter New Table-ID:");
                this.tableID = int.Parse(Console.ReadLine());
                if (searchUniqueTableID(this.tableID)) //while creating new table checks either the tableID in already assigned or not
                {
                    Console.WriteLine("ERROR! Table-ID already assigned, kindly choose another.");
                }
            }
            while (searchUniqueTableID(this.tableID));

            gameStatusProperty = 0; //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
            playerOneProperty = 0;
            playerTwoProperty = 0;
            startTimeProperty = DateTime.Now;
            endTimeProperty = DateTime.Now;
            Console.WriteLine("Table Successfully Created.");

            writeAddTableFile(this); //appends new to table to table file
        }
        public ArrayList readTableFileList() //reads table file to list and then returns list
        {
            ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
```

```
            StreamReader readTableFile = new StreamReader("Tables.txt"); //read file
            table t;

            while (!readTableFile.EndOfStream) //reads table file till end
            {
                t = new table();
                t.tableIDProperty = int.Parse(readTableFile.ReadLine());
                t.gameStatusProperty = int.Parse(readTableFile.ReadLine());
                        t.startTime = DateTime.Parse(readTableFile.ReadLine());
                        t.endTime = DateTime.Parse(readTableFile.ReadLine());
                        t.playerOneProperty = double.Parse(readTableFile.ReadLine());
                        t.playerTwoProperty = double.Parse(readTableFile.ReadLine());
                        tableList.Add(t);
                    }
                    readTableFile.Close();
                    return tableList;
                }
                public void writeAddTableFile(table t) //to add new table to the system by appending
                {
                    StreamWriter writeTableFile = new StreamWriter("Tables.txt", true); //appending table file

                    writeTableFile.WriteLine(t.tableIDProperty);
                    writeTableFile.WriteLine(t.gameStatusProperty);
                    writeTableFile.WriteLine(t.startTimeProperty);
                    writeTableFile.WriteLine(t.endTimeProperty);
                    writeTableFile.WriteLine(t.playerOneProperty);
                    writeTableFile.WriteLine(t.playerTwoProperty);

                    writeTableFile.Close();
                }
                public void writeTableFileList(ArrayList tableList) //to write modified/updated data to file -> modify/update -> table status
                {
                    StreamWriter writeTableFile = new StreamWriter("Tables.txt"); //not opened in appended mode because all modified/updated data is to write to file -> modify/update -> table status

                    for (int i = 0; i < tableList.Count; i++)
                    {
                        writeTableFile.WriteLine((tableList[i] as table).tableIDProperty);
                        writeTableFile.WriteLine((tableList[i] as table).gameStatusProperty);
                        writeTableFile.WriteLine((tableList[i] as table).startTimeProperty);
                        writeTableFile.WriteLine((tableList[i] as table).endTimeProperty);
                        writeTableFile.WriteLine((tableList[i] as table).playerOneProperty);
                        writeTableFile.WriteLine((tableList[i] as table).playerTwoProperty);
                    }
                    writeTableFile.Close();
                }
                public bool assignNewTable(double playerOneUserID) //to assign new table if one player comes
                {
                    ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
                    tableList = readTableFileList(); //reads tables from file to list
```

```
                    for (int i = 0; i < tableList.Count; i++) //checks each table
                    {
                        if ((tableList[i] as table).gameStatus == 0) //if table is empty
                        {
                            (tableList[i] as table).gameStatusProperty = 1; //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                            (tableList[i] as table).startTimeProperty = DateTime.Now;
                            (tableList[i] as table).endTimeProperty = DateTime.Now;
                            (tableList[i] as table).playerOneProperty = playerOneUserID;
                            (tableList[i] as table).playerTwoProperty = 0;
                            writeTableFileList(tableList); //write again to file
                            Console.WriteLine("Table Successfully Assigned.");
                            return true;
                        }
                        else if ((tableList[i] as table).gameStatus == 1) //if table has 1 playyer then assign the new player to this table
                        {
                            (tableList[i] as table).gameStatusProperty = 2; //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                            (tableList[i] as table).startTimeProperty = DateTime.Now;
                            (tableList[i] as table).endTimeProperty = DateTime.Now;
                            (tableList[i] as table).playerTwoProperty = playerOneUserID;
                            writeTableFileList(tableList); //write again to file
                            Console.WriteLine("Table Successfully Assigned.");
                            return true;
                        }
                    }
                    return false;
                }
                public bool assignNewTable(double playerOneUserID, double playerTwoUserID) //to assign new table if two players come
                {
                    ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
                    tableList = readTableFileList(); //reads tables from file to list

                    for (int i = 0; i < tableList.Count; i++) //checks each table
                    {
                        if ((tableList[i] as table).gameStatus == 0) //if table is empty then assign to them
                        {
                            (tableList[i] as table).gameStatusProperty = 2; //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                            (tableList[i] as table).startTimeProperty = DateTime.Now;
                            (tableList[i] as table).endTimeProperty = DateTime.Now;
                            (tableList[i] as table).playerOneProperty = playerOneUserID;
                            (tableList[i] as table).playerTwoProperty = playerTwoUserID;
                            writeTableFileList(tableList); //write again to file
                            Console.WriteLine("Table Successfully Assigned.");
                            return true;
                        }
                    }
                    for (int i = 0; i < tableList.Count; i++) //checks each table
```

```
                    {
                        if ((tableList[i] as table).gameStatus == 1) //as no full table is empty so now it will check table where one player is assigned so that they can start game immediately
                        {
                            (tableList[i] as table).gameStatusProperty = 2; //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                            (tableList[i] as table).startTimeProperty = DateTime.Now;
                            (tableList[i] as table).endTimeProperty = DateTime.Now;

                            (tableList[i] as table).playerOneProperty = playerOneUserID;
                            (tableList[i] as table).playerTwoProperty = playerTwoUserID;
                            writeTableFileList(tableList);
                            Console.WriteLine("Table Successfully Assigned.");
                            return true;
                        }
                    }
                    return false;
                }
                public void displayTableList() //to display all tables status
                {
                    ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
                    tableList = readTableFileList(); //reads tables from file to list

                    Console.Write("------------------------------------------------------------------------------");
                    Console.WriteLine("DISPLAY ALL TABLES STATUS:");
                    Console.Write("------------------------------------------------------------------------------");

                    for (int i = 0; i < tableList.Count; i++) //checks each table
                    {
                        Console.WriteLine();
                        Console.WriteLine("----------------------");
                        Console.WriteLine("Table-ID: " + (tableList[i] as table).tableID);
                        Console.WriteLine("----------------------");
                        if ((tableList[i] as table).gameStatus == 0) //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                        {
                            Console.WriteLine("0 Players Assigned.");
                        }
                        else if ((tableList[i] as table).gameStatus == 1) //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                        {
                            player p = new player();
                            Console.WriteLine("1 Player Assigned.");

                            string playerOneName = p.fullName((tableList[i] as table).playerOneProperty);
                            Console.WriteLine("1. Player-1 (ID-" + (tableList[i] as table).playerOneProperty + ") " + playerOneName);
                            Console.WriteLine("Start Time:" + (tableList[i] as table).startTimeProperty);
                        }
                        if ((tableList[i] as table).gameStatus == 2) //game status i.e. 0->Empty Table, 1->One Player Assigned, 2->Two Players Assigned
                        {
                            player p = new player();
                            Console.WriteLine("2 Players Assigned.");

```

```
                            string playerOneName = p.fullName((tableList[i] as table).playerOneProperty);
                            string playerTwoName = p.fullName((tableList[i] as table).playerTwoProperty);
                            Console.WriteLine("1. Player-1 (ID-" + (tableList[i] as table).playerOneProperty + ") " + playerOneName);
                            Console.WriteLine("2. Player-2 (ID-" + (tableList[i] as table).playerTwoProperty + ") " + playerTwoName);
                            Console.WriteLine("Start Time:" + (tableList[i] as table).startTimeProperty);
                        }
                    }
                }
                public void submitTableResults() //to submit game results and clear table status
                {
                    ArrayList tableList = new ArrayList(); //ArrayList to store list of tables
                    tableList = readTableFileList(); //reads tables from file to list

                    Console.Write("------------------------------------------------------------------------------");
                    Console.WriteLine("SUBMIT TABLE RESULTS:");
                    Console.Write("------------------------------------------------------------------------------");

                    Console.WriteLine("Enter Table-ID:");
                    int id = int.Parse(Console.ReadLine());

                    for (int i = 0; i < tableList.Count; i++) //checks each table
                    {
                        if ((tableList[i] as table).tableID == id) //checks for the required table
                        {
                            player p = new player();
                            Console.WriteLine("Select Won User:"); //asks for the won user
                            string playerOneName = p.fullName((tableList[i] as table).playerOneProperty);
                            string playerTwoName = p.fullName((tableList[i] as table).playerTwoProperty);
                            Console.WriteLine("1. Player-1 (ID-" + (tableList[i] as table).playerOneProperty + ") " + playerOneName);
                            Console.WriteLine("2. Player-2 (ID-" + (tableList[i] as table).playerTwoProperty + ") " + playerTwoName);
                            Console.WriteLine("3. Game Draw.");
                            int choice = int.Parse(Console.ReadLine());
                            if (choice == 1)
                            {
                                p.playerWon((tableList[i] as table).playerOneProperty); //updates won status of player1
                                p.playerLost((tableList[i] as table).playerTwoProperty); //updates lost status of player2
                            }
                            else if (choice == 2)
                            {
                                p.playerWon((tableList[i] as table).playerTwoProperty); //updates won status of player2
                                p.playerLost((tableList[i] as table).playerOneProperty); //updates lost status of player1
                            }
                            else if (choice == 3)
                            {
```

```
                                p.playerDraw((tableList[i] as table).playerOneProperty, (tableList[i] as table).playerTwoProperty); //updates draw status of both users
                            }
                            else
                            {
                                Console.WriteLine("ERROR!!! Invalid Input.");
                            }
                            (tableList[i] as table).endTimeProperty = DateTime.Now;

                            StreamWriter writeGameLogFile = new StreamWriter("GameLog.txt", true);
                            writeGameLogFile.WriteLine((tableList[i] as table).tableIDProperty);
                            writeGameLogFile.WriteLine((tableList[i] as table).playerOneProperty);
                            writeGameLogFile.WriteLine((tableList[i] as table).playerTwoProperty);
                            writeGameLogFile.WriteLine((tableList[i] as table).startTimeProperty);
                            writeGameLogFile.WriteLine((tableList[i] as table).endTimeProperty);
                            writeGameLogFile.Close();

                            (tableList[i] as table).gameStatus = 0; //clears table status to empty
                            (tableList[i] as table).playerOneProperty = 0; //clears table player1 to empty
                            (tableList[i] as table).playerTwoProperty = 0; //clears table player2 to empty
                            writeTableFileList(tableList);
                            return;
                        }
                    }
                }
                public void displayGameLogFile() //to display game history
                {
                    Console.Write("------------------------------------------------------------------------------");
                    Console.WriteLine("DISPLAY GAME LOG HISTORY:");
                    Console.Write("------------------------------------------------------------------------------");

                    double id;
                    double player1ID;
                    double player2ID;
                    DateTime start;
                    DateTime end;

                    StreamReader readGameLogFile = new StreamReader("GameLog.txt");

                    while (!readGameLogFile.EndOfStream) //reads game log file till end
                    {
                        id = double.Parse(readGameLogFile.ReadLine());
                        player1ID = double.Parse(readGameLogFile.ReadLine());
                        player2ID = double.Parse(readGameLogFile.ReadLine());
                        start = DateTime.Parse(readGameLogFile.ReadLine());
                        end = DateTime.Parse(readGameLogFile.ReadLine());

                        Console.WriteLine("T-ID:" + id + " | P1-ID:" + player1ID + " | P2-ID:" + player2ID + " | START:" + start + " | END:" + end);
                    }
                }
```

```
            }
        }
```

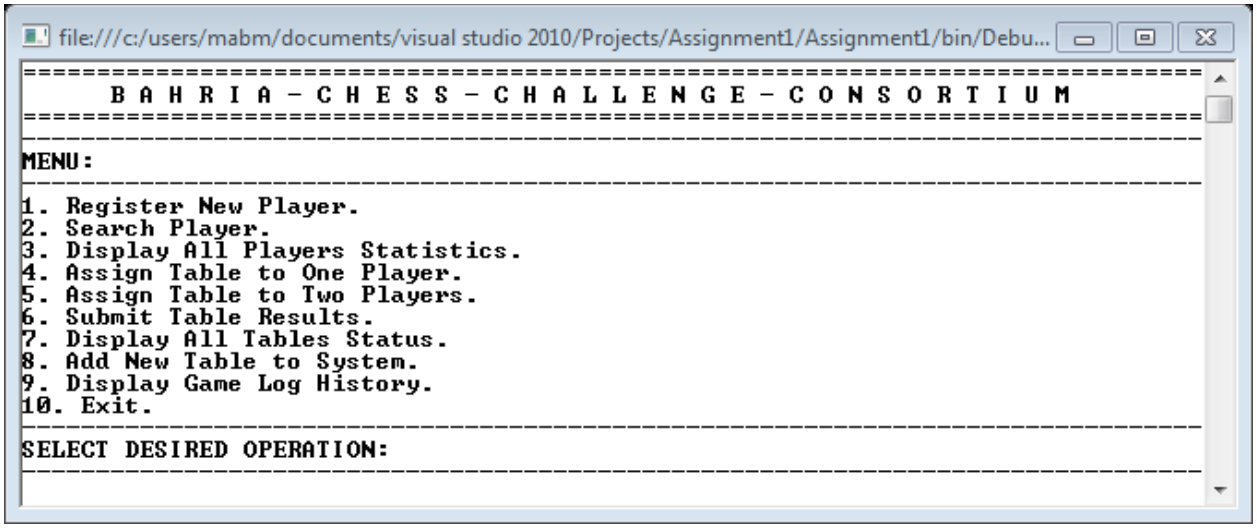
Program.cs File:

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

namespace Assignment1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.BackgroundColor = ConsoleColor.White;
            Console.Clear();
            Console.ForegroundColor = ConsoleColor.Black;
            Console.Write("==========================================================================");
            Console.WriteLine("      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M");
            Console.Write("==========================================================================");

            consoleMenu c = new consoleMenu();
            c.printConsoleMenu();
        }
    }
}
```

Console Display Output:



file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Deb...

```
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
1
----------------------------------------------------------------------------------
CREATE NEW PLAYER:
----------------------------------------------------------------------------------
Enter New User-ID:
1111
Enter First Name:
Anas
Enter Last Name:
Baig
Enter CNIC:
6110132323
Player Succesfully Registered
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
1
----------------------------------------------------------------------------------
CREATE NEW PLAYER:
----------------------------------------------------------------------------------
Enter New User-ID:
2222
Enter First Name:
Ali
Enter Last Name:
Mirza
```

```
file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
2
----------------------------------------------------------------------------------
SEARCH PLAYER:
----------------------------------------------------------------------------------
Enter Desired Operation:
1. Search by User-ID.
2. Search by Name.
3. Search by CNIC.
1

Enter Search Player User-ID.
2222
----------------------
Player - 2 Statistics:
----------------------
User ID:      2222
Name:         Ali Mirza
CNIC:         6110143599
SCORE:
------
Won:          0
Draw:         0
Lost:         0
----------------------
Total Games: 0
----------------------
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
```

file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...

```
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
3
----------------------------------------------------------------------------------
DISPLAY ALL PLAYERS STATISTICS:
----------------------------------------------------------------------------------

-----------------------
Player - 1 Statistics:
-----------------------
User ID:      1111
Name:         Anas Baig
CNIC:         6110132323
SCORE:
------
Won:          0
Draw:         0
Lost:         0
-----------------------
Total Games: 0
-----------------------

-----------------------
Player - 2 Statistics:
-----------------------
User ID:      2222
Name:         Ali Mirza
CNIC:         6110143599
SCORE:
------
Won:          0
Draw:         0
Lost:         0
-----------------------
Total Games: 0
-----------------------

-----------------------
Player - 3 Statistics:
-----------------------
User ID:      3333
```

file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...

```
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
8
----------------------------------------------------------------------------------
ADD NEW TABLE TO SYSTEM:
----------------------------------------------------------------------------------
Enter New Table-ID:
1
Table Successfully Created.
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
8
----------------------------------------------------------------------------------
ADD NEW TABLE TO SYSTEM:
----------------------------------------------------------------------------------
Enter New Table-ID:
2
Table Successfully Created.
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
```

```
file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
4
----------------------------------------------------------------------------------
ASSIGN TABLE TO ONE PLAYER:
----------------------------------------------------------------------------------
Enter Player User-ID:
1111
Table Successfully Assigned.
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
```

```
file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
5
----------------------------------------------------------------------------------
ASSIGN TABLE TO TWO PLAYERS:
----------------------------------------------------------------------------------
Enter Player-1 User-ID:
2222
Enter Player-2 User-ID:
3333
Table Successfully Assigned.
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
```

```
file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
6
----------------------------------------------------------------------------------
SUBMIT TABLE RESULTS:
----------------------------------------------------------------------------------
Enter Table-ID:
2
Select Won User:
1. Player-1 (ID-2222) Ali Mirza
2. Player-2 (ID-3333) Arif Baig
3. Game Draw.
1
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
```

```
file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...
==============================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==============================================================================
------------------------------------------------------------------------------
MENU:
------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
------------------------------------------------------------------------------
7
------------------------------------------------------------------------------
DISPLAY ALL TABLES STATUS:
------------------------------------------------------------------------------

----------------------
Table-ID: 1
----------------------
1 Player Assigned.
1. Player-1 (ID-1111) Anas Baig
Start Time:08-Oct-17 10:42:52 PM

----------------------
Table-ID: 2
----------------------
0 Players Assigned.

----------------------
Table-ID: 3
----------------------
0 Players Assigned.

----------------------
Table-ID: 4
----------------------
0 Players Assigned.

----------------------
Table-ID: 5
----------------------
0 Players Assigned.

----------------------
Table-ID: 6
----------------------
0 Players Assigned.

----------------------
Table-ID: 7
```

file:///c:/users/mabm/documents/visual studio 2010/Projects/Assignment1/Assignment1/bin/Debu...

```
==================================================================================
      B A H R I A - C H E S S - C H A L L E N G E - C O N S O R T I U M
==================================================================================
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
9
----------------------------------------------------------------------------------
DISPLAY GAME LOG HISTORY:
----------------------------------------------------------------------------------
T-ID:2 ¦ P1-ID:2222 ¦ P2-ID:3333 ¦ START:08-Oct-17 10:43:35 PM ¦ END:08-Oct-17 1
0:44:37 PM
----------------------------------------------------------------------------------
MENU:
----------------------------------------------------------------------------------
1. Register New Player.
2. Search Player.
3. Display All Players Statistics.
4. Assign Table to One Player.
5. Assign Table to Two Players.
6. Submit Table Results.
7. Display All Tables Status.
8. Add New Table to System.
9. Display Game Log History.
10. Exit.
----------------------------------------------------------------------------------
SELECT DESIRED OPERATION:
----------------------------------------------------------------------------------
```

Text File Output:

Players.txt File:

Players - Notepad

File Edit Format View Help

```
1111
Anas
Baig
6110132323
0
0
0
2222
Ali
Mirza
6110143599
1
0
0
3333
Arif
Baig
6110234243
0
0
1
```

GameLog.txt File:

GameLog - Notepad

File Edit Format View Help

```
2
2222
3333
08-Oct-17 10:43:35 PM
08-Oct-17 10:44:37 PM
```

Tables.txt File:

Tables - Notepad

File Edit Format View Help

```
1
1
08-Oct-17 10:42:52 PM
08-Oct-17 10:42:52 PM
1111
0
2
0
08-Oct-17 10:43:35 PM
08-Oct-17 10:44:37 PM
0
0
3
0
08-Oct-17 10:41:51 PM
08-Oct-17 10:41:51 PM
0
0
4
0
08-Oct-17 10:41:54 PM
08-Oct-17 10:41:54 PM
0
0
5
0
08-Oct-17 10:41:56 PM
08-Oct-17 10:41:56 PM
0
0
6
0
```