```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Poker
{
    public partial class mainForm : Form
    {
        Hands hand = new Hands();

        SetTable setTable = new SetTable();
        List<Player> listOfPlayers = new List<Player>();
        String suit = "";

        int[] handNumbers = new int[] { 0, 0, 0, 0, 0, 0, 0 };

        List<int> playersList = new List<int>();
        //List<int> deck = new List<int>();
        int countFlop = 0;
        int[] flopCards = new int[] { 0, 0, 0, 0, 0 };
        int currentBettingPlayer = 0;
        int currentBettingPlayerCount = 0;
        Boolean allCardsShown = false;
        int countNumberOfPlayers = 0;
        int potAmount = 0;
        int currentBetAmount = 10;
        int currentRaiser = 0;
        Boolean played = false;

        List<int> strenghtList = new List<int> { }; Player player1 = new Player(
        Player player5 = new Player(); Player player6 = new Player(); Player pla

        int[] pairs1 = new int[2]; int[] pairs2 = new int[2]; int[] pairs3 = new
        int[] pairs6 = new int[2]; int[] pairs7 = new int[2]; int[] pairs8 = new

        String[] resultAfterHandChecked1 = new string[9]; String[] resultAfterHa
        String[] resultAfterHandChecked4 = new string[9]; String[] resultAfterHa
        String[] resultAfterHandChecked7 = new string[9]; String[] resultAfterHa

        Result result1; Result result2; Result result3; Result result4; Result r
        List<Result> listOfResults = new List<Result> { };
```

```
public mainForm()
{
    InitializeComponent();
    int flipBtnLocation = (this.panel1.Size.Width - this.flipBtn.Size.Wi
    this.flipBtn.Location = new Point(flipBtnLocation, 220); ;
}

// Event listeners to handle the player addition checkboxes
public void when1Checked(Object sender, EventArgs e)
{
   firstPlayerPanel.Visible = (firstPlayerPanel.Visible) ? false : true;
    countNumberOfPlayers = (firstPlayerPanel.Visible) ? countNumberOfPla
}

public void when2Checked(Object sender, EventArgs e)
{
    secondPlayerPanel.Visible = (secondPlayerPanel.Visible) ? false : tr
    countNumberOfPlayers = (secondPlayerPanel.Visible) ? countNumberOfPl
}

public void when3Checked(Object sender, EventArgs e)
{
    thirdPlayerPanel.Visible = (thirdPlayerPanel.Visible) ? false : true
    countNumberOfPlayers = (thirdPlayerPanel.Visible) ? countNumberOfPla
}

public void when4Checked(Object sender, EventArgs e)
{
    fourthPlayerPanel.Visible = (fourthPlayerPanel.Visible) ? false : tr
    countNumberOfPlayers = (fourthPlayerPanel.Visible) ? countNumberOfPl
}

public void when5Checked(Object sender, EventArgs e)
{
    fifthPlayerPanel.Visible = (fifthPlayerPanel.Visible) ? false : true
    countNumberOfPlayers = (fifthPlayerPanel.Visible) ? countNumberOfPla
}

public void when6Checked(Object sender, EventArgs e)
{
    sixthPlayerPanel.Visible = (sixthPlayerPanel.Visible) ? false : true
    countNumberOfPlayers = (sixthPlayerPanel.Visible) ? countNumberOfPla
}

public void when7Checked(Object sender, EventArgs e)
{
    seventhPlayerPanel.Visible = (seventhPlayerPanel.Visible) ? false :
```

```
}

public void when8Checked(Object sender, EventArgs e)
{
    eighthPlayerPanel.Visible = (eighthPlayerPanel.Visible) ? false : tr
    countNumberOfPlayers = (eighthPlayerPanel.Visible) ? countNumberOfPl

}

// start game button
private void startBtn_Click(object sender, EventArgs e)
{
    if (countNumberOfPlayers < 2)
    {
        MessageBox.Show("Please select at least two players.", "Select P
    }
    else
    {
        if(!played){
            playersList = setTable.setTable(firstPlayerPanel.Visible, se
        fourthPlayerPanel.Visible, fifthPlayerPanel.Visible, sixthPlayer
        seventhPlayerPanel.Visible, eighthPlayerPanel.Visible);
        }

        setPlayers();
        disablePlayersBoxes();

        currentBettingPlayer = setTable.listOfPlayers[0].id;
        betTurn();
        potAmountlbl.Visible = true;
        potAmountlbl.Text = "Pot Amount: " + "$" + potAmount + ".00";
        currentBetAmountLbl.Text = "Current Bet Amount: " + "$" + curren
        played = true;
    }
}

// after table is set the initial cards are set up based on the list of
public void setPlayers()
{
    if(!played){
        listOfPlayers = setTable.dealCards();
    } else{
        for (int i = 0; i < listOfPlayers.Count; i++)
        {
            listOfPlayers[i].hand = setTable.setPlayerCards();
        }
    }
```

```
{
    if (listOfPlayers[i].id == 1)
    {
        this.name1Lbl.Text = listOfPlayers[i].name;
        this.firstPmoneyLbl.Text = listOfPlayers[i].cash.ToString();
        this.firstPpicBox1.BackColor = Color.White;
        this.firstPpicBox2.BackColor = Color.White;
        this.firstPpicBox1.Load("../../images/png/" + correctedNumbe
        this.firstPpicBox2.Load("../../images/png/" + correctedNumbe
    }
    if (listOfPlayers[i].id == 2)
    {
        this.name2Lbl.Text = listOfPlayers[i].name;
        this.secondPmoneyLbl.Text = listOfPlayers[i].cash.ToString()
        this.secondPpicBox1.BackColor = Color.White;
        this.secondPpicBox2.BackColor = Color.White;
        this.secondPpicBox1.Load("../../images/png/" + correctedNumb
        this.secondPpicBox2.Load("../../images/png/" + correctedNumb
    }
    if (listOfPlayers[i].id == 3)
    {
        this.name3Lbl.Text = listOfPlayers[i].name;
        this.thirdPmoneyLbl.Text = listOfPlayers[i].cash.ToString();
        this.thirdPpicBox1.BackColor = Color.White;
        this.thirdPpicBox2.BackColor = Color.White;
        this.thirdPpicBox1.Load("../../images/png/" + correctedNumbe
        this.thirdPpicBox2.Load("../../images/png/" + correctedNumbe
    }
    if (listOfPlayers[i].id == 4)
    {
        this.name4Lbl.Text = listOfPlayers[i].name;
        this.fourthPmoneyLbl.Text = listOfPlayers[i].cash.ToString()
        this.fourthPpicBox1.BackColor = Color.White;
        this.fourthPpicBox2.BackColor = Color.White;
        this.fourthPpicBox1.Load("../../images/png/" + correctedNumb
        this.fourthPpicBox2.Load("../../images/png/" + correctedNumb
    }
    if (listOfPlayers[i].id == 5)
    {
        this.name5Lbl.Text = listOfPlayers[i].name;
        this.fifthPmoneyLbl.Text = listOfPlayers[i].cash.ToString();
        this.fifthPpicBox1.BackColor = Color.White;
        this.fifthPpicBox2.BackColor = Color.White;
        this.fifthPpicBox1.Load("../../images/png/" + correctedNumbe
        this.fifthPpicBox2.Load("../../images/png/" + correctedNumbe
    }
    if (listOfPlayers[i].id == 6)
    {
```

```
            this.thirdPmoneyLbl.Text = listOfPlayers[i].cash.ToString();
            this.sixthPpicBox1.BackColor = Color.White;
            this.sixthPpicBox2.BackColor = Color.White;
            this.sixthPpicBox1.Load("../../images/png/" + correctedNumbe
            this.sixthPpicBox2.Load("../../images/png/" + correctedNumbe
        }
        if (listOfPlayers[i].id == 7)
        {
            this.name7Lbl.Text = listOfPlayers[i].name;
            this.seventhPmoneyLbl.Text = listOfPlayers[i].cash.ToString(
            this.seventhPpicBox1.BackColor = Color.White;
            this.seventhPpicBox2.BackColor = Color.White;
            this.seventhPpicBox1.Load("../../images/png/" + correctedNum
            this.seventhPpicBox2.Load("../../images/png/" + correctedNum
        }
        if (listOfPlayers[i].id == 8)
        {
            this.name8Lbl.Text = listOfPlayers[i].name;
            this.eighthPmoneyLbl.Text = listOfPlayers[i].cash.ToString()
            this.eighthPpicBox1.BackColor = Color.White;
            this.eighthPpicBox2.BackColor = Color.White;
            this.eighthPpicBox1.Load("../../images/png/" + correctedNumb
            this.eighthPpicBox2.Load("../../images/png/" + correctedNumb
        }
    }
}

public void disablePlayersBoxes()
{
    if (startBtn.Visible)
    {
        firstPPlayingLbl.Enabled = false;
        secondPPlayingLbl.Enabled = false;
        thirdPPlayingLbl.Enabled = false;
        fourthPPlayingLbl.Enabled = false;
        fifthPPlayingLbl.Enabled = false;
        sixthPPlayingLbl.Enabled = false;
        seventhPPlayingLbl.Enabled = false;
        eighthPPlayingLbl.Enabled = false;
        startBtn.Visible = false;
    }
    else
    {
        firstPPlayingLbl.Enabled = true;
        secondPPlayingLbl.Enabled = true;
        thirdPPlayingLbl.Enabled = true;
        fourthPPlayingLbl.Enabled = true;
        fifthPPlayingLbl.Enabled = true;
```

```
            seventhPPlayingLbl.Enabled = true;
            eighthPPlayingLbl.Enabled = true;
            startBtn.Visible = true;
        }
    }

    // method to handle each player's turn
    public void betTurn()
    {
        if (currentRaiser == currentBettingPlayer)
        {
            currentBettingPlayer = 0;
            currentBettingPlayerCount = 0;
            currentRaiser = 0;
        }

        if (currentBettingPlayerCount == 0 && currentBettingPlayer == 0 && (
        {
            turnOffPlayers();
            currentBettingPlayer = 0;
            flipBtn.Visible = true;
            currentBetAmount = 0;
            currentBetAmountLbl.Text = "Current Bet: ";
            if (allCardsShown)
            {
                checkHands();
                checkWinner();
                flipBtn.Visible = false;
            }
            currentRaiser = 0;
        }
        else
        {
            if (currentRaiser != 0 && currentBettingPlayer == 0)
            {
                if (currentRaiser == 1)
                {
                    currentRaiser = 0;
                }
                currentBettingPlayer = setTable.listOfPlayers[0].id;
                currentBettingPlayerCount = 0;
            }

            switch (currentBettingPlayer)
            {
                case 1:
                    this.turnCK1.BackColor = Color.Lime;
                    this.firstPbetRB.Visible = true;
```

```
            this.firstPBetBtn.Visible = true;
            this.firstPCheckRB.Checked = true;
            this.fold1.Visible = true;
            if (currentBetAmount > 0)
            {
                firstPCheckRB.Visible = false;
                firstPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.firstPAmountTxtB.Text = "10";
            }
            else
            {
                this.firstPAmountTxtB.Text = currentBetAmount.ToStri
            }
            break;
        case 2:
            this.turnCK2.BackColor = Color.Lime;
            this.secondPbetRB.Visible = true;
            this.secondPCheckRB.Visible = true;
            this.secondPBetBtn.Visible = true;
            this.secondPCheckRB.Checked = true;
            this.fold2.Visible = true;
            if (secondPCheckRB.Checked)
            {
                secondPAmountTxtB.Text = "10";
            }
            if (currentBetAmount > 0)
            {
                secondPCheckRB.Visible = false;
                secondPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.secondPAmountTxtB.Text = "10";
            }
            else
            {
                this.secondPAmountTxtB.Text = currentBetAmount.ToStr
            }
            break;
        case 3:
            this.turnCK3.BackColor = Color.Lime;
            this.thirdPbetRB.Visible = true;
            this.thirdPCheckRB.Visible = true;
            this.thirdPBetBtn.Visible = true;
            this.thirdPCheckRB.Checked = true;
```

```
            if (currentBetAmount > 0)
            {
                thirdPCheckRB.Visible = false;
                thirdPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.thirdPAmountTxtB.Text = "10";
            }
            else
            {
                this.thirdPAmountTxtB.Text = currentBetAmount.ToStri
            }
            break;
        case 4:
            this.turnCK4.BackColor = Color.Lime;
            this.fourthPbetRB.Visible = true;
            this.fourthPCheckRB.Visible = true;
            this.fourthPBetBtn.Visible = true;
            this.fourthPCheckRB.Checked = true;
            this.fold4.Visible = true;
            if (currentBetAmount > 0)
            {
                fourthPCheckRB.Visible = false;
                fourthPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.fourthPAmountTxtB.Text = "10";
            }
            else
            {
                this.fourthPAmountTxtB.Text = currentBetAmount.ToStr
            }
            break;
        case 5:
            this.turnCK5.BackColor = Color.Lime;
            this.fifthPbetRB.Visible = true;
            this.fifthPCheckRB.Visible = true;
            this.fifthPBetBtn.Visible = true;
            this.fifthPCheckRB.Checked = true;
            this.fold5.Visible = true;
            if (currentBetAmount > 0)
            {
                fifthPCheckRB.Visible = false;
                fifthPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
```

```
                this.fifthPAmountTxtB.Text = "10";
            }
            else
            {
                this.fifthPAmountTxtB.Text = currentBetAmount.ToStri
            }
            break;
        case 6:
            this.turnCK6.BackColor = Color.Lime;
            this.sixthPbetRB.Visible = true;
            this.sixthPCheckRB.Visible = true;
            this.sixthPBetBtn.Visible = true;
            this.sixthPCheckRB.Checked = true;
            this.fold6.Visible = true;
            if (currentBetAmount > 0)
            {
                sixthPCheckRB.Visible = false;
                sixthPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.sixthPAmountTxtB.Text = "10";
            }
            else
            {
                this.sixthPAmountTxtB.Text = currentBetAmount.ToStri
            }
            break;
        case 7:
            this.turnCK7.BackColor = Color.Lime;
            this.seventhPbetRB.Visible = true;
            this.seventhPCheckRB.Visible = true;
            this.seventhPBetBtn.Visible = true;
            this.seventhPCheckRB.Checked = true;
            this.fold7.Visible = true;
            if (currentBetAmount > 0)
            {
                seventhPCheckRB.Visible = false;
                seventhPbetRB.Checked = true;
            }
            if (currentBetAmount == 0)
            {
                this.seventhPAmountTxtB.Text = "10";
            }
            else
            {
                this.seventhPAmountTxtB.Text = currentBetAmount.ToSt
            }
```

```
            case 8:
                this.turnCK8.BackColor = Color.Lime;
                this.eighthPbetRB.Visible = true;
                this.eighthPCheckRB.Visible = true;
                this.eighthPBetBtn.Visible = true;
                this.eighthPCheckRB.Checked = true;
                this.fold8.Visible = true;
                if (currentBetAmount > 0)
                {
                    eighthPCheckRB.Visible = false;
                    eighthPbetRB.Checked = true;
                }
                if (currentBetAmount == 0)
                {
                    this.eighthPAmountTxtB.Text = "10";
                }
                else
                {
                    this.eighthPAmountTxtB.Text = currentBetAmount.ToStr
                }
                break;
        }
    }
}

public void betTurnListener(Object sender, EventArgs e)
{
    switch (currentBettingPlayer)
    {
        case 1:
            int p1 = currentBettingPlayer - 1;
            int p1Cash = Convert.ToInt32(setTable.listOfPlayers[p1].cash
            int betAmount1 = Convert.ToInt32(firstPAmountTxtB.Text);
            Boolean checkIsOn1 = false;
            if (firstPCheckRB.Checked)
            {
                checkIsOn1 = true;
            }
            betTurnAssistant(p1, p1Cash, betAmount1, checkIsOn1);

            firstPmoneyLbl.Text = setTable.listOfPlayers[p1].cash.ToStri
            firstPAmountTxtB.Visible = false;
            fold1.Visible = false;
            firstPAmountTxtB.Text = "10";
            checkIsOn1 = false;
            break;
        case 2:
            int p2 = currentBettingPlayer - 1;
```

```
            int betAmount2 = Convert.ToInt32(secondPAmountTxtB.Text);
            Boolean checkIsOn2 = false;
            if (secondPCheckRB.Checked)
            {
                checkIsOn2 = true;
            }
            betTurnAssistant(p2, p2Cash, betAmount2, checkIsOn2);

            secondPmoneyLbl.Text = setTable.listOfPlayers[p2].cash.ToStr
            secondPAmountTxtB.Visible = false;
            fold2.Visible = false;
            secondPAmountTxtB.Text = "10";
            checkIsOn1 = false;
            break;
        case 3:
            int p3 = currentBettingPlayer - 1;
            int p3Cash = Convert.ToInt32(setTable.listOfPlayers[p3].cash
            int betAmount3 = Convert.ToInt32(thirdPAmountTxtB.Text);
            Boolean checkIsOn3 = false;
            if (thirdPCheckRB.Checked)
            {
                checkIsOn3 = true;
            }
            betTurnAssistant(p3, p3Cash, betAmount3, checkIsOn3);

            thirdPmoneyLbl.Text = setTable.listOfPlayers[p3].cash.ToStri
            thirdPAmountTxtB.Visible = false;
            fold3.Visible = false;
            thirdPAmountTxtB.Text = "10";
            checkIsOn3 = false;
            break;
        case 4:
            int p4 = currentBettingPlayer - 1;
            int p4Cash = Convert.ToInt32(setTable.listOfPlayers[p4].cash
            int betAmount4 = Convert.ToInt32(fourthPAmountTxtB.Text);
            Boolean checkIsOn4 = false;
            fold4.Visible = false;
            if (fourthPCheckRB.Checked)
            {
                checkIsOn4 = true;
            }
            betTurnAssistant(p4, p4Cash, betAmount4, checkIsOn4);

            fourthPmoneyLbl.Text = setTable.listOfPlayers[p4].cash.ToStr
            fourthPAmountTxtB.Visible = false;
            fourthPAmountTxtB.Text = "10";
            checkIsOn4 = false;
            break;
```

```
            int p5 = currentBettingPlayer - 1;
            int p5Cash = Convert.ToInt32(setTable.listOfPlayers[p5].cash
            int betAmount5 = Convert.ToInt32(fifthPAmountTxtB.Text);
            Boolean checkIsOn5 = false;
            fold5.Visible = false;
            if (fifthPCheckRB.Checked)
            {
                checkIsOn5 = true;
            }
            betTurnAssistant(p5, p5Cash, betAmount5, checkIsOn5);

            fifthPmoneyLbl.Text = setTable.listOfPlayers[p5].cash.ToStri
            fifthPAmountTxtB.Visible = false;
            fold5.Visible = false;
            fifthPAmountTxtB.Text = "10";
            checkIsOn5 = false;
            break;
        case 6:
            int p6 = currentBettingPlayer - 1;
            int p6Cash = Convert.ToInt32(setTable.listOfPlayers[p6].cash
            int betAmount6 = Convert.ToInt32(sixthPAmountTxtB.Text);
            Boolean checkIsOn6 = false;
            fold6.Visible = false;
            if (sixthPCheckRB.Checked)
            {
                checkIsOn6 = true;
            }
            betTurnAssistant(p6, p6Cash, betAmount6, checkIsOn6);

            sixthPmoneyLbl.Text = setTable.listOfPlayers[p6].cash.ToStri
            sixthPAmountTxtB.Visible = false;
            sixthPAmountTxtB.Text = "10";
            checkIsOn6 = false;
            break;
        case 7:
            int p7 = currentBettingPlayer - 1;
            int p7Cash = Convert.ToInt32(setTable.listOfPlayers[p7].cash
            int betAmount7 = Convert.ToInt32(seventhPAmountTxtB.Text);
            Boolean checkIsOn7 = false;
            if (seventhPCheckRB.Checked)
            {
                checkIsOn7 = true;
            }
            betTurnAssistant(p7, p7Cash, betAmount7, checkIsOn7);

            seventhPmoneyLbl.Text = setTable.listOfPlayers[p7].cash.ToSt
            seventhPAmountTxtB.Visible = false;
            fold7.Visible = false;
```

```
            checkIsOn7 = false;
            break;
        case 8:
            int p8 = currentBettingPlayer - 1;
            int p8Cash = Convert.ToInt32(setTable.listOfPlayers[p8].cash
            int betAmount8 = Convert.ToInt32(eighthPAmountTxtB.Text);
            Boolean checkIsOn8 = false;
            if (eighthPCheckRB.Checked)
            {
                checkIsOn8 = true;
            }
            betTurnAssistant(p8, p8Cash, betAmount8, checkIsOn8);

            eighthPmoneyLbl.Text = setTable.listOfPlayers[p8].cash.ToStr
            eighthPAmountTxtB.Visible = false;
            fold8.Visible = false;
            eighthPAmountTxtB.Text = "10";
            checkIsOn8 = false;
            break;
    }
}

public void betTurnAssistant(int playerNumber, int playerCash, int playe
{
    String currentBetText = "Current Bet: $" + playerBetAmt + ".00";
    if (playerBetAmt < currentBetAmount)
    {
        MessageBox.Show("Bet amount has to match or be higher than curre
    }
    else
    {
        if (checkIsOn)
        {
            playerBetAmt = 0;
            currentBetText = "Current Bet: Check";
        }
        if (playerBetAmt > currentBetAmount)
        {
            currentRaiser = playerNumber + 1;
        }
        potAmount = potAmount + playerBetAmt;
        potAmountlbl.Text = "Pot Amount: " + "$" + potAmount + ".00";

        setTable.listOfPlayers[playerNumber].cash = playerCash - playerB

        currentBetAmountLbl.Text = "Current Bet Amount: $" + playerBetAm

        currentBetAmount = playerBetAmt;
```

```
            checkIfLastPlayer();
            turnOffPlayers();
            betTurn();
        }
    }

    public void checkIfLastPlayer()
    {
        //check to see if the current player is the last player, if yes, res
        if (currentBettingPlayerCount < setTable.listOfPlayers.Count - 1)
        {
            currentBettingPlayerCount++;
            currentBettingPlayer = setTable.listOfPlayers[currentBettingPlay
        }
        else
        {
            currentBettingPlayer = 0;
            currentBettingPlayerCount = 0;
            currentBetAmountLbl.Visible = false;
        }
    }

    // method to gather all strenght numbers from each player
    // compare them and see who got the highest.
    // If more than one player get the winning number than further compariso
    public void checkWinner(){
        int[] arrayStrenght = strenghtList.ToArray();
        List<int[]> pairs = new List<int[]> {};

        List<Result> playersWithWinningHands = new List<Result> { };
        List<int[]> handsToCompareList = new List<int[]> { };

        Array.Sort(arrayStrenght);
        Array.Sort<int>(arrayStrenght, new Comparison<int>((i1, i2) => i2.Co

        foreach (var res in listOfResults)
        {
            if (res.handStrenght == arrayStrenght[0]){
                playersWithWinningHands.Add(res);
                pairs.Add(res.pairs);
                handsToCompareList.Add(res.finalHand);
            }
        }

        if(playersWithWinningHands.Count == 1){
            winLbl.Text = playersWithWinningHands[0].player.name + " wins $"
            playersWithWinningHands[0].player.cash = playersWithWinningHands
        } else {
```

```
            List<int> winnerIndex = new List<int> { };
            winnerIndex = compareSimilarHands(handsToCompareList, arrayStren

            if(winnerIndex.Count > 1){
                int division = 0;
                division = potAmount / winnerIndex.Count;
                winLbl.Text = "Multiple Winners with a " + playersWithWinnin
                String winners = "(";
                for (int i = 0; i < playersWithWinningHands.Count; i++)
                {
                    for (int j = 0; j < winnerIndex.Count; j++)
                    {
                        if (i == winnerIndex[j])
                        {
                            if (i == playersWithWinningHands.Count - 1)
                            {
                                winners = winners + playersWithWinningHands[
                                playersWithWinningHands[i].player.cash = pla
                            }
                            else
                            {
                                winners = winners + playersWithWinningHands[
                                playersWithWinningHands[i].player.cash = pla
                            }
                        }
                    }
                }
                winnersNamesLbl.Text = winners;
                winnersNamesLbl.Visible = true;
                int winnersNamesLblLocation = (this.panel1.Size.Width - this
                this.winnersNamesLbl.Location = new Point(winnersNamesLblLoc

            } else {
                winLbl.Text = playersWithWinningHands[winnerIndex[0]].player
                playersWithWinningHands[winnerIndex[0]].player.cash = player
            }

            listOfResults[winnerIndex[0]].player.cash = listOfResults[winner
        }
        int winLblLocation = (this.panel1.Size.Width - this.winLbl.Size.Widt
        this.winLbl.Location = new Point(winLblLocation, 165);

        refreshMoney();
        winLbl.Visible = true;
        currentBetAmountLbl.Visible = false;
        potAmountlbl.Visible = false;
    }
```

```
public List<int> compareSimilarHands(List<int[]> winnersHands, int stren
    List<int> handIndex = new List<int> {};
    if (strenght == 1 || strenght == 5 || strenght == 8 || strenght == 6
        handIndex = selectHighestHandByHighestCard(winnersHands);
    } else if (strenght == 2 || strenght == 4 || strenght == 7){
        handIndex = selectPairs(winnersHands, pairs);
    } else {
        handIndex = selectPairs(winnersHands, pairs);
    }
    return handIndex;
}

public List<int> selectPairs(List<int[]> winnersHands, List<int[]> pairs
    List<int> index = new List<int> {};

    List<int> maxNumbers = new List<int> { };
    int[] emptyArray = new int[] { 0, 0, 0 };
    int[] largestArray = new int[2];
    int maxNum = 0;
    List<int> indexes = new List<int> { };

    // adding the largest number from each hand into a list and getting
    // of them
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < pairs.Count; j++)
        {
            maxNumbers.Add(pairs[j][i]);
        }
        maxNum = maxNumbers.Max();
        largestArray[i] = maxNum;

        // empting out the pairs that does not contain the largest pair
        for (int f = 0; f < pairs.Count; f++)
        {
            if (pairs[f][i] != maxNum)
            {
                pairs[f] = emptyArray;
            }
        }
        maxNumbers.Clear();
    }

    // checking to see which pairs are equal to the largest pair.
    // Then save the index of the pairs that are equal.
    int count = 0;
    for (int i = 0; i < pairs.Count; i++)
    {
```

```
            {
                count++;
                indexes.Add(i);
            }
        }
        // if only one set of pairs match the largest pair, then save the in
        if (count == 1){
            index.Add(indexes[0]);
        } else {
            // if more than one set of pairs match the largest pair, then se
            // to be checked by the selectHighestHandByHighestCard method.
            for (int i = 0; i < winnersHands.Count; i++)
            {
                Boolean checkTruth = false;
                for (int j = 0; j < indexes.Count; j++)
                {
                    if (i == indexes[j])
                    {
                        checkTruth = true;
                    }
                }
                if (!checkTruth)
                {
                    winnersHands[i] = emptyArray;
                }
            }
            index = selectHighestHandByHighestCard(winnersHands);
        }

        return index;
    }

    public List<int> selectHighestHandByHighestCard(List<int[]> hands){
        // gathering largesr card from each player

        List<int> index = new List<int> {};
        int[] zeroArray = new int[5] { 0, 0, 0, 0, 0 };

        for (int i = 0; i < 5; i++)
        {
            List<int> largest = new List<int> { };
            // adding highest card from each player into a list
            for (int j = 0; j < hands.Count; j++)
            {
                largest.Add(hands[j][i]);
            }

            // getting largest card among all players
```

```
            int maxIndex = largest.IndexOf(maxNumber);
            int count = 0;

            // checking to see if there's more than one player with the high
            index.Clear();
            for (int j = 0; j < hands.Count; j++)
            {
                if (maxNumber == hands[j][i])
                {
                    count++;
                    index.Add(j);
                }
                else
                {
                    // exclude all players that don't have the highest card
                    hands[j] = zeroArray;
                }
            }

            // if only one player has the highest card than break and return
            if (count == 1)
            {
                index.Clear();
                index.Add(maxIndex);
                break;
            }
        }
        return index;
    }

    public void refreshMoney()
    {

        for (int i = 0; i < setTable.listOfPlayers.Count; i++)
        {
            switch (listOfPlayers[i].id)
            {
                case 1:
                    firstPmoneyLbl.Text = (listOfResults[i].player.cash).ToS
                    break;
                case 2:
                    secondPmoneyLbl.Text = (listOfResults[i].player.cash).To
                    break;
                case 3:
                    thirdPmoneyLbl.Text = (listOfResults[i].player.cash).ToS
                    break;
                case 4:
                    fourthPmoneyLbl.Text = (listOfResults[i].player.cash).To
```

```
                case 5:
                    fifthPmoneyLbl.Text = (listOfResults[i].player.cash).ToS
                    break;
                case 6:
                    sixthPmoneyLbl.Text = (listOfResults[i].player.cash).ToS
                    break;
                case 7:
                    seventhPmoneyLbl.Text = (listOfResults[i].player.cash).T
                    break;
                case 8:
                    eighthPmoneyLbl.Text = (listOfResults[i].player.cash).To
                    break;
            }
        }
    }



    public void firstRB(Object sender, EventArgs e)
    {
        if (firstPbetRB.Checked)
        {
            firstPAmountTxtB.Visible = true;
            firstPBetBtn.Text = "Bet!";
        }
        else
        {
            firstPAmountTxtB.Visible = false;
            firstPBetBtn.Text = "Check!";
        }
    }

    public void secondRB(Object sender, EventArgs e)
    {
        if (secondPbetRB.Checked)
        {
            secondPAmountTxtB.Visible = true;
            secondPBetBtn.Text = "Bet!";
        }
        else
        {
            secondPAmountTxtB.Visible = false;
            secondPBetBtn.Text = "Check!";
        }
    }

    public void thirdRB(Object sender, EventArgs e)
    {
```

```
    {
        thirdPAmountTxtB.Visible = true;
        thirdPBetBtn.Text = "Bet!";
    }
    else
    {
        thirdPAmountTxtB.Visible = false;
        thirdPBetBtn.Text = "Check!";
    }
}

public void fourthRB(Object sender, EventArgs e)
{
    if (fourthPbetRB.Checked)
    {
        fourthPAmountTxtB.Visible = true;
        fourthPBetBtn.Text = "Bet!";
    }
    else
    {
        fourthPAmountTxtB.Visible = false;
        fourthPBetBtn.Text = "Check!";
    }
}

public void fifthRB(Object sender, EventArgs e)
{
    if (fifthPbetRB.Checked)
    {
        fifthPAmountTxtB.Visible = true;
        fifthPBetBtn.Text = "Bet!";
    }
    else
    {
        fifthPAmountTxtB.Visible = false;
        fifthPBetBtn.Text = "Check!";
    }
}

public void sixthRB(Object sender, EventArgs e)
{
    if (sixthPbetRB.Checked)
    {
        sixthPAmountTxtB.Visible = true;
        sixthPBetBtn.Text = "Bet!";
    }
    else
    {
```

```
            sixthPBetBtn.Text = "Check!";
        }
    }

    public void seventhRB(Object sender, EventArgs e)
    {
        if (seventhPbetRB.Checked)
        {
            seventhPAmountTxtB.Visible = true;
            seventhPBetBtn.Text = "Bet!";
        }
        else
        {
            seventhPAmountTxtB.Visible = false;
            seventhPBetBtn.Text = "Check!";
        }
    }

    public void eighthRB(Object sender, EventArgs e)
    {
        if (eighthPbetRB.Checked)
        {
            eighthPAmountTxtB.Visible = true;
            eighthPBetBtn.Text = "Bet!";
        }
        else
        {
            eighthPAmountTxtB.Visible = false;
            eighthPBetBtn.Text = "Check!";
        }
    }

    public void fold(object sender, EventArgs e)
    {

        if(turnCK1.BackColor == Color.Lime){
            playersList.Remove(1);
            listOfPlayers.RemoveAt(0);
            countNumberOfPlayers--;
            fold1.Visible = false;
        }

        checkIfLastPlayer();
        turnOffPlayers();
        betTurn();

    }
```

```
        int[] intHand = new int[5];

        intHand[0] = Convert.ToInt32(strHand[2]);
        intHand[1] = Convert.ToInt32(strHand[3]);
        intHand[2] = Convert.ToInt32(strHand[4]);
        intHand[3] = Convert.ToInt32(strHand[5]);
        intHand[4] = Convert.ToInt32(strHand[6]);

        return intHand;
    }

    public void checkHands()
    {
        for (int i = 0; i < playersList.Count; i++)
        {
            if (playersList[i] == 1)
            {
                player1 = setTable.listOfPlayers[i];

                resultAfterHandChecked1 = hand.checkHand(setTable.listOfPlay
                pairs1[0] = Convert.ToInt32(resultAfterHandChecked1[7]); pai

                result1 = new Result(player1, player1.hand, resultAfterHandC

                listOfResults.Add(result1);
                strenghtList.Add(result1.handStrenght);

                hand1Lbl.Text = result1.handType;
                int location = (this.p1WinPanel.Size.Width - this.hand1Lbl.S
                int location2 = (this.p1WinPanel.Size.Height - this.hand1Lbl
                this.hand1Lbl.Location = new Point(location, location2);
                p1WinPanel.Visible = true;
            }
            if (playersList[i] == 2)
            {
                player2 = setTable.listOfPlayers[i];

                resultAfterHandChecked2 = hand.checkHand(setTable.listOfPlay
                pairs2[0] = Convert.ToInt32(resultAfterHandChecked2[7]); pai
                result2 = new Result(player2, player2.hand, resultAfterHandC

                listOfResults.Add(result2);
                strenghtList.Add(result2.handStrenght);

                hand2Lbl.Text = result2.handType;
                int location = (this.p2WinPanel.Size.Width - this.hand2Lbl.S
                int location2 = (this.p2WinPanel.Size.Height - this.hand2Lbl
                this.hand2Lbl.Location = new Point(location, location2);
```

```
}
if (playersList[i] == 3)
{
    player3 = setTable.listOfPlayers[i];

    resultAfterHandChecked3 = hand.checkHand(setTable.listOfPlay
    pairs3[0] = Convert.ToInt32(resultAfterHandChecked3[7]); pai

    result3 = new Result(player3, player3.hand, resultAfterHandC

    listOfResults.Add(result3);
    strenghtList.Add(result3.handStrenght);

    hand3Lbl.Text = result3.handType;
    int location = (this.p3WinPanel.Size.Width - this.hand3Lbl.S
    int location2 = (this.p3WinPanel.Size.Height - this.hand3Lbl
    this.hand3Lbl.Location = new Point(location, location2);
    p3WinPanel.Visible = true;
}
if (playersList[i] == 4)
{
    player4 = setTable.listOfPlayers[i];

    resultAfterHandChecked4 = hand.checkHand(setTable.listOfPlay
    pairs4[0] = Convert.ToInt32(resultAfterHandChecked4[7]); pai
    result4 = new Result(player4, player4.hand, resultAfterHandC
    listOfResults.Add(result4);
    strenghtList.Add(result4.handStrenght);

    hand4Lbl.Text = result4.handType;
    int location = (this.p4WinPanel.Size.Width - this.hand4Lbl.S
    int location2 = (this.p4WinPanel.Size.Height - this.hand4Lbl
    this.hand4Lbl.Location = new Point(location, location2);
    p4WinPanel.Visible = true;
}
if (playersList[i] == 5)
{
    player5 = setTable.listOfPlayers[i];

    resultAfterHandChecked5 = hand.checkHand(setTable.listOfPlay
    pairs5[0] = Convert.ToInt32(resultAfterHandChecked5[7]); pai
    result5 = new Result(player5, player5.hand, resultAfterHandC
    listOfResults.Add(result5);
    strenghtList.Add(result5.handStrenght);

    hand5Lbl.Text = result5.handType;
    int location = (this.p5WinPanel.Size.Width - this.hand5Lbl.S
    int location2 = (this.p5WinPanel.Size.Height - this.hand5Lbl
```

```
            p5WinPanel.Visible = true;
        }
        if (playersList[i] == 6)
        {
            player6 = setTable.listOfPlayers[i];

            resultAfterHandChecked6 = hand.checkHand(setTable.listOfPlay
            pairs6[0] = Convert.ToInt32(resultAfterHandChecked6[7]); pai
            result6 = new Result(player6, player6.hand, resultAfterHandC
            listOfResults.Add(result6);
            strenghtList.Add(result6.handStrenght);

            hand6Lbl.Text = result6.handType;
            int location = (this.p6WinPanel.Size.Width - this.hand6Lbl.S
            int location2 = (this.p6WinPanel.Size.Height - this.hand6Lbl
            this.hand6Lbl.Location = new Point(location, location2);
            p6WinPanel.Visible = true;
        }
        if (playersList[i] == 7)
        {
            player7 = setTable.listOfPlayers[i];

            resultAfterHandChecked7 = hand.checkHand(setTable.listOfPlay
            pairs7[0] = Convert.ToInt32(resultAfterHandChecked7[7]); pai
            result7 = new Result(player7, player7.hand, resultAfterHandC
            listOfResults.Add(result7);
            strenghtList.Add(result7.handStrenght);

            hand7Lbl.Text = result7.handType;
            int location = (this.p7WinPanel.Size.Width - this.hand7Lbl.S
            int location2 = (this.p7WinPanel.Size.Height - this.hand7Lbl
            this.hand7Lbl.Location = new Point(location, location2);
            p7WinPanel.Visible = true;
        }
        if (playersList[i] == 8)
        {
            player8 = setTable.listOfPlayers[i];

            resultAfterHandChecked8 = hand.checkHand(setTable.listOfPlay
            pairs8[0] = Convert.ToInt32(resultAfterHandChecked8[7]); pai
            result8 = new Result(player8, player8.hand, resultAfterHandC
            listOfResults.Add(result8);
            strenghtList.Add(result8.handStrenght);

            hand8Lbl.Text = result8.handType;
            int location = (this.p8WinPanel.Size.Width - this.hand8Lbl.S
            int location2 = (this.p8WinPanel.Size.Height - this.hand8Lbl
            this.hand8Lbl.Location = new Point(location, location2);
```

```
        }
    }
}

private int correctedNumbers(int cardNumber)
{
    if (cardNumber < 14)
    {
        suit = "spades";
    }
    else
    if (cardNumber > 13 && cardNumber < 27)
    {
        suit = "clubs";
        cardNumber = cardNumber - 13;
    }
    else if (cardNumber > 26 && cardNumber < 40)
    {
        suit = "diamonds";
        cardNumber = cardNumber - 26;
    }
    else if (cardNumber > 39)
    {
        suit = "hearts";
        cardNumber = cardNumber - 39;
    }
    return cardNumber;
}

private void resetBtn_Click(object sender, EventArgs e)
{
    suit = "";
    int[] handNumbersClean = new int[] { 0, 0, 0, 0, 0, 0, 0 };
    handNumbers = handNumbersClean;

    int[] flopCardsClean = new int[] { 0, 0, 0, 0, 0 };
    flopCards = flopCardsClean;

    potAmount = 0;
    currentBetAmount = 10;

    hand.resetHand();
    cleanLabels();
    disablePlayersBoxes();
    strenghtList.Clear();
    setTable.createDeck();

    countFlop = 0;
```

```
        currentBettingPlayerCount = 0;
        currentRaiser = 0;

        winnersNamesLbl.Visible = false;
        flipBtn.Visible = false;
        allCardsShown = false;
        startBtn.Visible = true;
        winLbl.Visible = false;

    }

    public void cleanLabels()
    {
        hand1Lbl.Text = "";
        hand2Lbl.Text = "";
        hand3Lbl.Text = "";
        hand4Lbl.Text = "";
        hand5Lbl.Text = "";
        hand6Lbl.Text = "";
        hand7Lbl.Text = "";
        hand8Lbl.Text = "";

        this.firstPpicBox1.Image = null;
        this.firstPpicBox1.BackColor = Color.Transparent;
        this.firstPpicBox2.Image = null;
        this.firstPpicBox2.BackColor = Color.Transparent;
        this.secondPpicBox1.Image = null;
        this.secondPpicBox1.BackColor = Color.Transparent;
        this.secondPpicBox2.Image = null;
        this.secondPpicBox2.BackColor = Color.Transparent;
        this.thirdPpicBox1.Image = null;
        this.thirdPpicBox1.BackColor = Color.Transparent;
        this.thirdPpicBox2.Image = null;
        this.thirdPpicBox2.BackColor = Color.Transparent;
        this.fourthPpicBox1.Image = null;
        this.fourthPpicBox1.BackColor = Color.Transparent;
        this.fourthPpicBox2.Image = null;
        this.fourthPpicBox2.BackColor = Color.Transparent;
        this.fifthPpicBox1.Image = null;
        this.fifthPpicBox1.BackColor = Color.Transparent;
        this.fifthPpicBox2.Image = null;
        this.fifthPpicBox2.BackColor = Color.Transparent;
        this.sixthPpicBox1.Image = null;
        this.sixthPpicBox1.BackColor = Color.Transparent;
        this.sixthPpicBox2.Image = null;
        this.sixthPpicBox2.BackColor = Color.Transparent;
        this.seventhPpicBox1.Image = null;
        this.seventhPpicBox1.BackColor = Color.Transparent;
```

```
        this.seventhPpicBox2.BackColor = Color.Transparent;
        this.eighthPpicBox1.Image = null;
        this.eighthPpicBox1.BackColor = Color.Transparent;
        this.eighthPpicBox2.Image = null;
        this.eighthPpicBox2.BackColor = Color.Transparent;
        this.firstFlopPB.Image = null;
        this.firstFlopPB.BackColor = Color.Transparent;
        this.secondFlopPB.Image = null;
        this.secondFlopPB.BackColor = Color.Transparent;
        this.thirdFlopPB.Image = null;
        this.thirdFlopPB.BackColor = Color.Transparent;
        this.turnPB.Image = null;
        this.riverPB.BackColor = Color.Transparent;
        this.riverPB.Image = null;
        this.turnPB.BackColor = Color.Transparent;
        this.p1WinPanel.Visible = false;
        this.p2WinPanel.Visible = false;
        this.p3WinPanel.Visible = false;
        this.p4WinPanel.Visible = false;
        this.p5WinPanel.Visible = false;
        this.p6WinPanel.Visible = false;
        this.p7WinPanel.Visible = false;
        this.p8WinPanel.Visible = false;
    }

    private void flipBtn_Click(object sender, EventArgs e)
    {
        flipCards();
        currentBettingPlayer = setTable.listOfPlayers[0].id;
        currentBettingPlayerCount = 0;
        betTurn();
        currentBetAmount = 0;
    }

    public void flipCards()
    {
        flopCards = setTable.setFlop();
        setFlopCards(flopCards);
        addToHands(flopCards);
        countFlop++;
    }

    public void setFlopCards(int[] flopCards)
    {
        if (countFlop == 0)
        {
            this.firstFlopPB.Load(("../../images/png/" + correctedNumbers(fl
            this.secondFlopPB.Load(("../../images/png/" + correctedNumbers(f
```

```
            this.firstFlopPB.BackColor = Color.White;
            this.secondFlopPB.BackColor = Color.White;
            this.thirdFlopPB.BackColor = Color.White;
            this.flipBtn.Text = "Deal Turn";
            this.flipBtn.Visible = false;
        }
        else if (countFlop == 1)
        {
            this.turnPB.Load(("../../images/png/" + correctedNumbers(flopCar
            this.turnPB.BackColor = Color.White;
            this.flipBtn.Text = "Deal River";
            this.flipBtn.Visible = false;
        }
        else if (countFlop == 2)
        {
            this.riverPB.Load(("../../images/png/" + correctedNumbers(flopCa
            this.riverPB.BackColor = Color.White;
            this.turnPB.BackColor = Color.White;
            this.flipBtn.Text = "Deal Flop";
            this.flipBtn.Visible = false;
            allCardsShown = true;
        }
        else if (countFlop > 2)
        {
            countFlop = 0;
        }
    }

    public void addToHands(int[] flopCards)
    {
        if (countFlop == 0)
        {
            for (int i = 0; i < playersList.Count; i++)
            {
                if (playersList[i] == 1)
                {
                    setTable.listOfPlayers[i].hand[2] = flopCards[0];
                    setTable.listOfPlayers[i].hand[3] = flopCards[1];
                    setTable.listOfPlayers[i].hand[4] = flopCards[2];
                }
                if (playersList[i] == 2)
                {
                    setTable.listOfPlayers[i].hand[2] = flopCards[0];
                    setTable.listOfPlayers[i].hand[3] = flopCards[1];
                    setTable.listOfPlayers[i].hand[4] = flopCards[2];
                }
                if (playersList[i] == 3)
                {
```

```
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
            if (playersList[i] == 4)
            {
                setTable.listOfPlayers[i].hand[2] = flopCards[0];
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
            if (playersList[i] == 5)
            {
                setTable.listOfPlayers[i].hand[2] = flopCards[0];
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
            if (playersList[i] == 6)
            {
                setTable.listOfPlayers[i].hand[2] = flopCards[0];
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
            if (playersList[i] == 7)
            {
                setTable.listOfPlayers[i].hand[2] = flopCards[0];
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
            if (playersList[i] == 8)
            {
                setTable.listOfPlayers[i].hand[2] = flopCards[0];
                setTable.listOfPlayers[i].hand[3] = flopCards[1];
                setTable.listOfPlayers[i].hand[4] = flopCards[2];
            }
        }
    }
    else if (countFlop == 1)
    {
        for (int i = 0; i < playersList.Count; i++)
        {
            if (playersList[i] == 1)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 2)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 3)
```

```
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 4)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 5)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 6)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 7)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
            if (playersList[i] == 8)
            {
                setTable.listOfPlayers[i].hand[5] = flopCards[3];
            }
        }

    }
    else if (countFlop == 2)
    {
        for (int i = 0; i < playersList.Count; i++)
        {
            if (playersList[i] == 1)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 2)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 3)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 4)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 5)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
```

```
            if (playersList[i] == 6)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 7)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
            if (playersList[i] == 8)
            {
                setTable.listOfPlayers[i].hand[6] = flopCards[4];
            }
        }
    }
}

public void turnOffPlayers()
{
    this.turnCK1.BackColor = Color.Green;
    this.firstPbetRB.Visible = false;
    this.firstPCheckRB.Visible = false;
    this.firstPBetBtn.Visible = false;

    this.turnCK2.BackColor = Color.Green;
    this.secondPbetRB.Visible = false;
    this.secondPCheckRB.Visible = false;
    this.secondPBetBtn.Visible = false;

    this.turnCK3.BackColor = Color.Green;
    this.thirdPbetRB.Visible = false;
    this.thirdPCheckRB.Visible = false;
    this.thirdPBetBtn.Visible = false;

    this.turnCK4.BackColor = Color.Green;
    this.fourthPbetRB.Visible = false;
    this.fourthPCheckRB.Visible = false;
    this.fourthPBetBtn.Visible = false;

    this.turnCK5.BackColor = Color.Green;
    this.fifthPbetRB.Visible = false;
    this.fifthPCheckRB.Visible = false;
    this.fifthPBetBtn.Visible = false;

    this.turnCK6.BackColor = Color.Green;
    this.sixthPbetRB.Visible = false;
    this.sixthPCheckRB.Visible = false;
    this.sixthPBetBtn.Visible = false;
```

```
        this.seventhPbetRB.Visible = false;
        this.seventhPCheckRB.Visible = false;
        this.seventhPBetBtn.Visible = false;

        this.turnCK8.BackColor = Color.Green;
        this.eighthPbetRB.Visible = false;
        this.eighthPCheckRB.Visible = false;
        this.eighthPBetBtn.Visible = false;
    }

    private void guessTxt_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
        {
            e.Handled = true;
        }
    }

    private void onLeave(object sender, EventArgs e)
    {
        if (firstPAmountTxtB.Text == "" || Convert.ToInt32(firstPAmountTxtB.
        {
            if (!firstPCheckRB.Checked)
            {
                MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
                firstPAmountTxtB.Text = "10";
            }
        }
        if (secondPAmountTxtB.Text == "" || Convert.ToInt32(secondPAmountTxt
        {
            if (!secondPCheckRB.Checked)
            {
                MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
                secondPAmountTxtB.Text = "10";
            }
        }
        if (thirdPAmountTxtB.Text == "" || Convert.ToInt32(thirdPAmountTxtB.
        {
            if (!thirdPCheckRB.Checked)
            {
                MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
                thirdPAmountTxtB.Text = "10";
            }
        }
        if (fourthPAmountTxtB.Text == "" || Convert.ToInt32(fourthPAmountTxt
        {
            if (!fourthPCheckRB.Checked)
            {
```

```
            fourthPAmountTxtB.Text = "10";
        }
    }
    if (fifthPAmountTxtB.Text == "" || Convert.ToInt32(fifthPAmountTxtB.
    {
        if (!fifthPCheckRB.Checked)
        {
            MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
            fifthPAmountTxtB.Text = "10";
        }
    }
    if (sixthPAmountTxtB.Text == "" || Convert.ToInt32(sixthPAmountTxtB.
    {
        if (!sixthPCheckRB.Checked)
        {
            MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
            sixthPAmountTxtB.Text = "10";
        }
    }
    if (seventhPAmountTxtB.Text == "" || Convert.ToInt32(seventhPAmountT
    {
        if (!seventhPCheckRB.Checked)
        {
            MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
            seventhPAmountTxtB.Text = "10";
        }
    }
    if (eighthPAmountTxtB.Text == "" || Convert.ToInt32(eighthPAmountTxt
    {
        if (!eighthPCheckRB.Checked)
        {
            MessageBox.Show("Bet amount cannot be less than 10", "Bet Hi
            eighthPAmountTxtB.Text = "10";
        }
    }
}


//public void createDeck()
//{
//    for (int i = 1; i < 53; i++)
//    {
//        deck.Add(i);
//    }
//}

public void click8times(){
    firstPBetBtn.PerformClick();
```

```
        thirdPBetBtn.PerformClick();
        fourthPBetBtn.PerformClick();
        fifthPBetBtn.PerformClick();
        sixthPBetBtn.PerformClick();
        seventhPBetBtn.PerformClick();
        eighthPBetBtn.PerformClick();
    }

    private void test()
    {
        // first
        firstPPlayingLbl.Checked = true;
        secondPPlayingLbl.Checked = true;
        thirdPPlayingLbl.Checked = true;
        fourthPPlayingLbl.Checked = true;
        fifthPPlayingLbl.Checked = true;
        sixthPPlayingLbl.Checked = true;
        seventhPPlayingLbl.Checked = true;
        eighthPPlayingLbl.Checked = true;

        // second
        startBtn.PerformClick();

        // third
        click8times();

        // fourth
        flipBtn.PerformClick();

        // fifth
        click8times();

        // sixth
        flipBtn.PerformClick();

        // seventh
        click8times();

        // eighth
        flipBtn.PerformClick();

        // nineth
        click8times();



        //int[] ar1 = new int[] { 9, 7, 5 };
        //int[] ar2 = new int[] { 8, 3, 1 };
```

```
//int[] ar4 = new int[] { 9, 7, 4 };

//int[][] theArray = new int[4][];
//theArray[0] = ar1;
//theArray[1] = ar2;
//theArray[2] = ar3;
//theArray[3] = ar4;


//List<int> maxNumbers = new List<int> { };
//int[] emptyArray = new int[] { 0, 0, 0 };
//int[] largestArray = new int[3];
//int maxNum = 0;
//List<int> indexes = new List<int> { };

//for (int i = 0; i < 2; i++)
//{
//    for (int j = 0; j < theArray.Length; j++)
//    {
//        maxNumbers.Add(theArray[j][i]);
//    }
//    maxNum = maxNumbers.Max();
//    largestArray[i] = maxNum;
//    for (int f = 0; f < theArray.Length; f++)
//    {
//        if(theArray[f][i] != maxNum){
//            theArray[f] = emptyArray;
//        }
//    }
//    maxNumbers.Clear();
//}

//int count = 0;
//for (int i = 0; i < theArray.Length; i++)
//{
//    if (theArray[i][0] == largestArray[0] && theArray[i][1] == lar
//    {
//        count++;
//        indexes.Add(i);
//    }
//}
//Console.WriteLine(count);
//foreach (var item in indexes)
//{
//    Console.Write(item + ", ");
//}
```

```
    //int[] stringArrayInitializer = new int[5];
    //Player p1 = new Player(1, "Mark", 20000, ar1, "");
    //Player p2 = new Player(2, "Paul", 20000, ar2, "");
    //Player p3 = new Player(3, "Susan", 20000, ar3, "");
    //Player p4 = new Player(4, "Lucas", 20000, ar4, "");

    //List<Player> group = new List<Player> { p1, p2, p3, p4 };
    //String index = checkWinner(group, 4);

   //Console.WriteLine(index);

    //    int count = 0;
//    string handResult = "";
//    do
//    {

//        int num1 = 0;
//        int num2 = 0;
//        int num3 = 0;
//        int num4 = 0;
//        int num5 = 0;
//        int num6 = 0;
//        int num7 = 0;
//        int[] testHandNums = new int[7];
//        int[] copytestHandNums = new int[7];

//        List<int> testDeck = new List<int>();

//        for (int i = 1; i < 53; i++)
//        {
//            testDeck.Add(i);
//        }

//        String[] handInfo = new string[12];

//        int randomNumber1;
//        randomNumber1 = random.Next(1, testDeck.Count + 1) - 1;
//        num1 = testDeck[randomNumber1];
//        testHandNums[0] = num1;
//        copytestHandNums[0] = num1;
//        testDeck.Remove(num1);

//        int randomNumber2;
//        randomNumber2 = random.Next(1, testDeck.Count + 1) - 1;
//        num2 = testDeck[randomNumber2];
//        testHandNums[1] = num2;
//        copytestHandNums[1] = num2;
```

```
//          int randomNumber3;
//          randomNumber3 = random.Next(1, testDeck.Count + 1) - 1;
//          num3 = testDeck[randomNumber3];
//          testHandNums[2] = num3;
//          copytestHandNums[2] = num3;
//          testDeck.Remove(num3);

//          int randomNumber4;
//          randomNumber4 = random.Next(1, testDeck.Count + 1) - 1;
//          num4 = testDeck[randomNumber4];
//          testHandNums[3] = num4;
//          copytestHandNums[3] = num4;
//          testDeck.Remove(num4);

//          int randomNumber5;
//          randomNumber5 = random.Next(1, testDeck.Count + 1) - 1;
//          num5 = testDeck[randomNumber5];
//          testHandNums[4] = num5;
//          copytestHandNums[4] = num5;
//          testDeck.Remove(num5);

//          int randomNumber6;
//          randomNumber6 = random.Next(1, testDeck.Count + 1) - 1;
//          num6 = testDeck[randomNumber6];
//          testHandNums[5] = num6;
//          copytestHandNums[5] = num6;
//          testDeck.Remove(num6);

//          int randomNumber7;
//          randomNumber7 = random.Next(1, testDeck.Count + 1) - 1;
//          num7 = testDeck[randomNumber7];
//          testHandNums[6] = num7;
//          copytestHandNums[6] = num7;
//          testDeck.Remove(num7);

//          handInfo = hand.checkHand(testHandNums);

//          handResult = handInfo[0];


//          Console.WriteLine("----------------");
//          Console.WriteLine("HandInfo " + handInfo[0]);
//          Console.WriteLine("Hand strenght is: " + handInfo[1]);
//          Console.WriteLine("Hand: " + handInfo[2] + " " + handInfo[3] +
//          //Console.WriteLine("Suits: " + handInfo[7] + " " + handInfo[8

//          //Console.WriteLine(testHandNums[0].ToString());
```

```
        //        //Console.WriteLine(testHandNums[2].ToString());
        //        //Console.WriteLine(testHandNums[3].ToString());
        //        //Console.WriteLine(testHandNums[4].ToString());
        //        //Console.WriteLine(testHandNums[5].ToString());
        //        //Console.WriteLine(testHandNums[6].ToString());
        //        //Console.WriteLine("----------------");
        //        //Console.WriteLine("----------------");
        //        //Console.WriteLine("----------------");

        //        count++;
        //        eighthPPlayingLbl.Text = count.ToString();

        //    } while (handResult != "");
        }

        private void testBtn_Click(object sender, EventArgs e)
        {
            test();
        }
    }
}
```