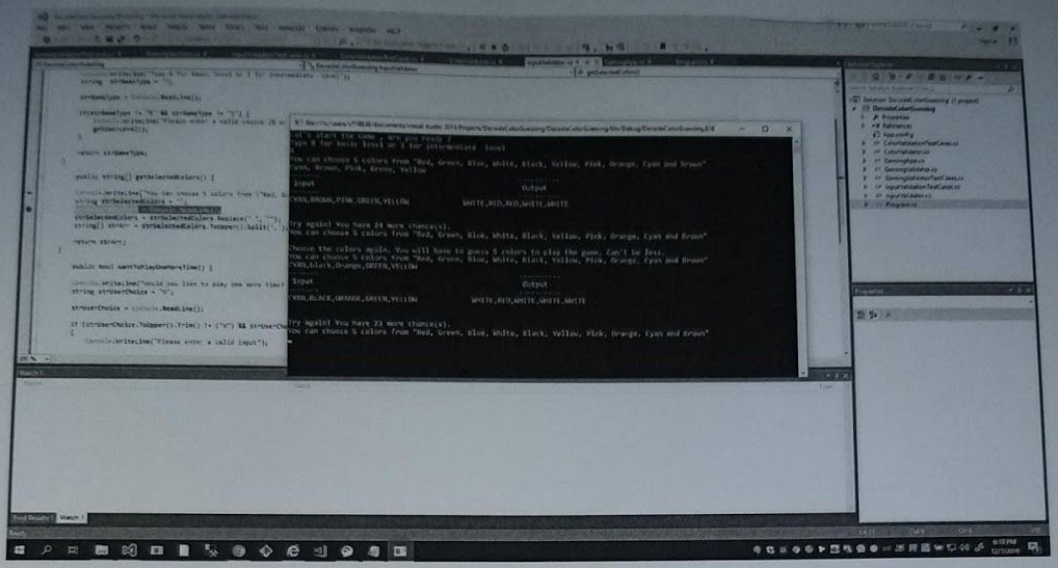

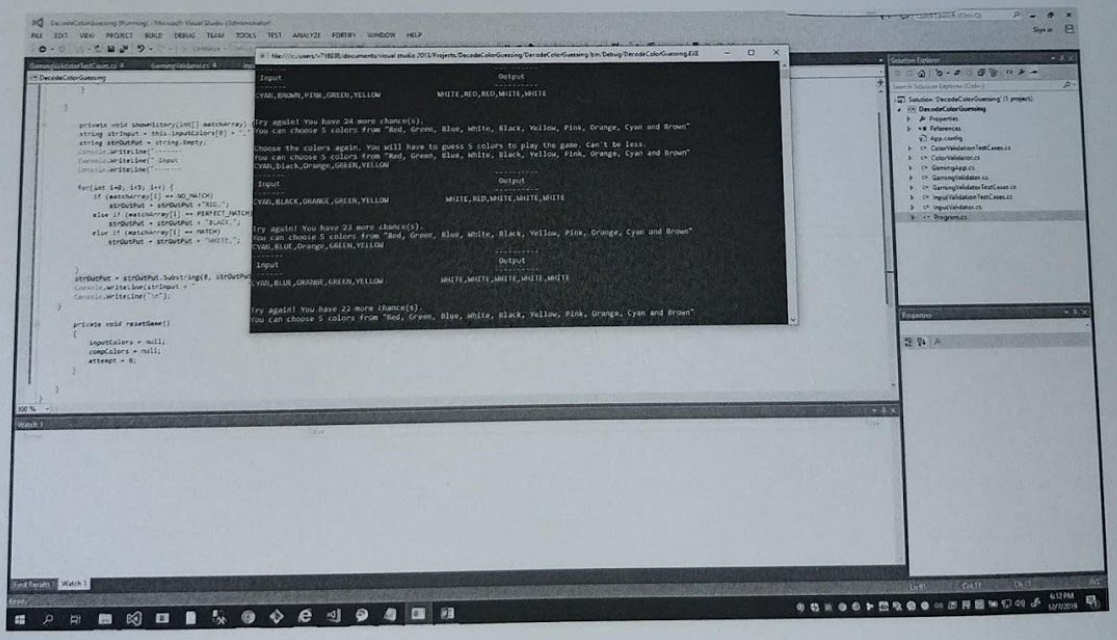

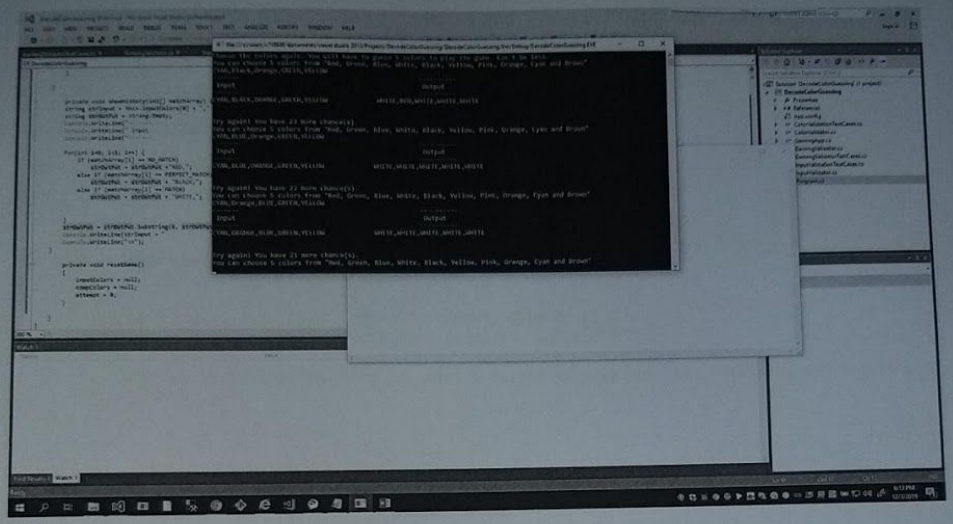

Able to color the all colors luckily but the perfect matchI am unable to.

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

namespace DecodeColorGuessing
{
    class ColorValidationTestCases
    {

        ColorValidator colorsUtil = new ColorValidator();

        void testvalidateColorslessColors()
        {

            string[] colors = { "BLUE", "GREEN", "CYAN", "BLACK" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsMoreColors()
        {

            string[] colors = { "BLUE", "GREEN", "CYAN", "BLACK", "RED", "WHITE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You  have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsNullColors()
        {

            string[] colors = null;
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You  have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsInvalidColors()
        {

            string[] colors = { "PURPLE", "GREEN", "CYAN", "BLACK", "BLUE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You will have to guess 5 colors to play the game.Please enter 5 Colors. ↙
");
            }
        }


        void testvalidateColorsUniqueColors()
        {

            string[] colors = { "BLUE", "CYAN", "CYAN", "BLACK", "WHITE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("Duplicate value (CYAN) . You opted for Intermedaite Level game .        ↙
Duplicates are not allowed.Please enter 5 Different Colors.");
```

```
            }
        }


        void testvalidateColorsBasicSuccessColors()
        {

            string[] colors = { "BLUE", "RED", "CYAN", "BLACK", "WHITE" };
            if (colorsUtil.validateColors(colors, false) == "SUCCESS")
            {
                Console.WriteLine("Input Validation Succes.");
            }
        }


        void testvalidateColorsAdvancedSuccess1Colors()
        {

            string[] colors = { "BLUE", "RED", "CYAN", "BLACK", "WHITE" };

            if (colorsUtil.validateColors(colors, false) == "SUCCESS")
            {
                Console.WriteLine("Input Validation Succes.");
            }
        }


        void testvalidateColorsAdvancedSuccess2Colors()
        {
            string[] colors = { "RED", "RED", "CYAN", "BLACK", "RED" };
            if (colorsUtil.validateColors(colors, false) == "SUCCESS")
            {
                Console.WriteLine("Input Validation Succes.");
            }
        }
    }
}
```

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

namespace DecodeColorGuessing
{
    class ColorValidationTestCases
    {

        ColorValidator colorsUtil = new ColorValidator();

        void testvalidateColorslessColors()
        {

            string[] colors = { "BLUE", "GREEN", "CYAN", "BLACK" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsMoreColors()
        {

            string[] colors = { "BLUE", "GREEN", "CYAN", "BLACK", "RED", "WHITE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You  have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsNullColors()
        {

            string[] colors = null;
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You  have to guess 5 colors to play the game. Please enter 5 Colors.");
            }
        }


        void testvalidateColorsInvalidColors()
        {

            string[] colors = { "PURPLE", "GREEN", "CYAN", "BLACK", "BLUE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("You will have to guess 5 colors to play the game.Please enter 5 Colors. ↙
");
            }
        }


        void testvalidateColorsUniqueColors()
        {

            string[] colors = { "BLUE", "CYAN", "CYAN", "BLACK", "WHITE" };
            if (colorsUtil.validateColors(colors, true) == "SUCCESS")
            {
                Console.WriteLine("Duplicate value (CYAN) . You opted for Intermedaite Level game .        ↙
Duplicates are not allowed.Please enter 5 Different Colors.");
```

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

namespace DecodeColorGuessing
{
    class ColorValidator
    {
        public static string[] ALL_COLORS = { "RED", "GREEN", "BLUE", "WHITE", "BLACK", "YELLOW", "PINK",
    "ORANGE", "CYAN", "BROWN" };
        InputValidator inputValidator = new InputValidator();

        public string[] getValidatedUserColors(bool uniqueColors) {
        string validationResult = null;
        string[] userChoices = null;
        do {
            userChoices = inputValidator.getSelectedColors();
            validationResult = validateColors(userChoices, uniqueColors);
            if(!validationResult.Equals("SUCCESS"))
                Console.WriteLine("Choose the colors again. "+ validationResult);

        }while (!validationResult.Equals("SUCCESS"));
        return userChoices;
    }

        public string validateColors(string[] colors, bool uniqueColors) {
        if(colors == null || colors.Length < 5 || colors.Length > 5) {
            return "You will have to guess 5 colors to play the game. Can't be less.";
        }

        List<string> list = ALL_COLORS.ToList();
        for(int i=0; i<5; i++) {
            if(!list.Contains(colors[i]))
                return colors[i] + " is not a valid color.";
        }

        if(uniqueColors) {
            List<string> seenBeforeList = new List<string>();
            for(int index = 0; index < 5; index++) {
                string value = colors[index];
                if(seenBeforeList.Contains(value))
                    return value + " appears more than once. Duplicates are not allowed in basic level.";
                else
                    seenBeforeList.Add(value);

            }
        }
        return "SUCCESS";
    }
    }
}
```

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

namespace DecodeColorGuessing
{
    class GamingApp
    {
        private static int PERFECT_MATCH = 1;
        private static int NO_MATCH = 0;
        private static int MATCH = 2;

        private ColorValidator colorValidator = new ColorValidator();
        private GamingValidator gamingValidator = new GamingValidator();

        private string[] inputColors = null;
        private string[] compColors = null;

        private int attempt = 0;
        private bool basic = true;

        public void playGame(bool basicLevel)
        {
            this.basic = basicLevel;
            this.compColors = gamingValidator.getColorsForIndices(gamingValidator.generateIndices(this.
   basic));
            this.startAttempt();
        }

        private void startAttempt()
        {
            // get colors from user
            this.inputColors = colorValidator.getValidatedUserColors(this.basic);

            //increment attempt
            this.attempt++;

            //play
            this.checkGameStatus();
        }

        private void checkGameStatus() {
        int perfectMatchCount = 0;
        int[] matchArray = new int[5];

        for(int idx=0; idx<5; idx++) {
            if(inputColors[idx].Trim().Equals(compColors[idx]))
            {
                matchArray[idx] = 1;
                perfectMatchCount++;
            }
            else {
                List<string> list = compColors.ToList();
                if(list.Contains(inputColors[idx].Trim()))
                    matchArray[idx] = 2;
            }
        }

        if(perfectMatchCount == 5) {
            Console.WriteLine("Congratulations! You won the game. ");
            resetGame();
        }
        else {
            if(attempt != 25) {
```

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

namespace DecodeColorGuessing
{
    class GamingApp
    {
        private static int PERFECT_MATCH = 1;
        private static int NO_MATCH = 0;
        private static int MATCH = 2;

        private ColorValidator colorValidator = new ColorValidator();
        private GamingValidator gamingValidator = new GamingValidator();

        private string[] inputColors = null;
        private string[] compColors = null;

        private int attempt = 0;
        private bool basic = true;

        public void playGame(bool basicLevel)
        {
            this.basic = basicLevel;
            this.compColors = gamingValidator.getColorsForIndices(gamingValidator.generateIndices(this.
   basic));
            this.startAttempt();
        }

        private void startAttempt()
        {
            // get colors from user
            this.inputColors = colorValidator.getValidatedUserColors(this.basic);

            //increment attempt
            this.attempt++;

            //play
            this.checkGameStatus();
        }

        private void checkGameStatus() {
        int perfectMatchCount = 0;
        int[] matchArray = new int[5];

        for(int idx=0; idx<5; idx++) {
            if(inputColors[idx].Trim().Equals(compColors[idx]))
            {
                matchArray[idx] = 1;
                perfectMatchCount++;
            }
            else {
                List<string> list = compColors.ToList();
                if(list.Contains(inputColors[idx].Trim()))
                    matchArray[idx] = 2;
            }
        }

        if(perfectMatchCount == 5) {
            Console.WriteLine("Congratulations! You won the game. ");
            resetGame();
        }
        else {
            if(attempt != 25) {
```

```
            showHistory(matchArray);
            Console.WriteLine("Try again! You have "+(25-attempt) + " more chance(s).");
            startAttempt();
        }
        else {
            Console.WriteLine("Game Over! Better luck next time.");
            resetGame();
        }
    }

}

    private void showHistory(int[] matchArray) {
    string strInput = this.inputColors[0] + "," + this.inputColors[1] + "," + this.inputColors[2] + "," ↙
 + this.inputColors[3] + "," + this.inputColors[4];
    string strOutPut = string.Empty;                                                    ----------      ↙
    Console.WriteLine("-------
                      ");                                                               Output          ↙
    Console.WriteLine(" Input
                       ");                                                              ----------      ↙
    Console.WriteLine("-------
                     ");

    for(int i=0; i<5; i++) {
        if (matchArray[i] == NO_MATCH)
            strOutPut = strOutPut +"RED,";
        else if (matchArray[i] == PERFECT_MATCH)
            strOutPut = strOutPut + "BLACK,";
        else if (matchArray[i] == MATCH)
            strOutPut = strOutPut + "WHITE,";


    }
    strOutPut = strOutPut.Substring(0, strOutPut.Length - 1);
    Console.WriteLine(strInput + "              " + strOutPut + "     ");
    Console.WriteLine("\n");
}

    private void resetGame()
    {
        inputColors = null;
        compColors = null;
        attempt = 0;
    }

}
}
```

```
            return randomIndexes;
        }
    }
}
```

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

namespace DecodeColorGuessing
{
    class GamingValidator
    {
        public int[] generateIndices(bool unique)
        {
            int[] indices = null;
           indices = RandomColorsIndexes(unique);
            return indices;
        }

        public string[] getColorsForIndices(int[] indices)
        {
            string[] colors = new string[5];
            for (int i = 0; i < indices.Length; i++)
            {
                colors[i] = ColorValidator.ALL_COLORS[indices[i]];
            }
            return colors;
        }

        public static int[] RandomColorsIndexes( bool blnUnique)
        {
            Random random = new Random();
            int num = 0;
            int[] randomIndexes = new int[5];
            for (int iCounter = 0; iCounter < 5; iCounter++)
            {
                num = random.Next(0, 9);
                if(blnUnique == true)
                {

                    if (randomIndexes.Contains(num) == false)
                        {
                            randomIndexes[iCounter] = num;
                        }
                        else
                        {
                            num = random.Next(0, 9);
                            if (randomIndexes.Contains(num) == false)
                            {
                                randomIndexes[iCounter] = num;
                            }
                            else
                            {
                                num = random.Next(0, 9);
                                if (randomIndexes.Contains(num) == false)
                                {
                                    randomIndexes[iCounter] = num;
                                }
                            }
                        }

                }
                else
                {
                    randomIndexes[iCounter] = num;
                }

            }
```

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

namespace DecodeColorGuessing
{
    class GamingValidatorTestCases
    {
        void testGenerateIndices() {
        Console.WriteLine("Not yet implemented");
    }


    void testGetColorsForIndices() {
        Console.WriteLine("Not yet implemented");
    }

    }
}
```



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

10.As a player, I want to Play the game, so the attempt gets incremented

Acceptance Criteria:1

Given I Play the Game for the 1st time

When I make wrong guess

Then Attempt Count should be incremented to 2

Acceptance Criteria:2

Given I Play the Game for the 7th time

When I make wrong guess

Then Attempt Count should be incremented to 8

11.As a player, I want to the Guess the Color and position correctly for all five color, So that system displays "Black,Black,Black,Black,Black"

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And All the Color and order matches with the system guess

Then System should displays "Black,Black,Black,Black,Black"

12. As a player, I want to the Guess the Color and position wrongly for all five color, So that system displays "Red,Red,Red,Red,Red"

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And all the Color and order doesn't matches with the system guess

Then System should displays "Red,Red,Red,Red,Red"

13. As a player, I want to the Guess the correct Color and position wrongly for all five color, So that system displays "White ,White,White,White,White"

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And all the Color matches with the system guess

And The position does not match with the system guess

Then System should displays "White,White,White,White"

14.As a player, I want to the Guess the Color and position correctly for any one all five color, So that system displays "Black" in the respective position

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And any one Color and order matches with the system guess

Then System should displays "Black" in the respective position

15. As a player, I want to the Guess the Color and position wrongly for any one of five color, So that system displays "Red" in the respective position

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And any one Color and order doesn't match with the system guess

Then System should display "Red" in the respective position

16. As a player, I want to the Guess the correct Color and position wrongly for any one of the five color, so that system displays "White" in the respective position

Acceptance Criteria:1

Given I Play the Game

When I enter five color from the list

And one of the Color matches with the system guess

And The position does not match with the system guess

Then System should display "White" in the respective position

17.As a player, i should play the game and male a wrong guess on 25 attempts, so that system display the message "Game Over! Better Luck Next time

Acceptance Criteria:1

Given I play the game

When I enter the guess wrong for the"25th time

Then System should display "Game Over! Better Luck Next time"

18.As a Player, I want to game to be over when I make the correct Guess

Acceptance Criteria:1

Given I play the game

When I enter the color

And the Guess matches with the system choice

And System should display "You Won"

And I try to play the game

Then System should not allow the user to enter the guess

19.As a player, I want to win the game, so the system displays "You Won"

Acceptance Criteria:1

Given I play the game

When I enter the color

And the Guess matches with the system choice

Then System should display "You Won"

20.As advanced Level Player, I want to enter duplicate color, so I can play the game

Acceptance Criteria:1

Given I play the game

When I enter the advanced level

And I enter duplicate color

Then system should allow the user to enter duplicate color