```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Diagnostics;

using System.Text.RegularExpressions;

using System.Reflection;


namespace ProgramAssignment3

{

   /// <summary>

   /// Searching through a WordList

   /// </summary>

   class Program

   {



      // Creates a public array for all of the words

      public string[] lines =
System.IO.File.ReadAllLines(@"C:\Users\sani\Dropbox\School\College\4.Senior\First
Semester\ComPE361\Labs\ProgramAssignment2" + @"\WordList.txt");


      //Creates a public array for all the letters of the alphabet

      public char[] az = Enumerable.Range('a', 'z' - 'a' + 1).Select(i => (Char)i).ToArray();
```

```
        /// <summary>
        /// Goes through a word that has been morphed and computs the morphed of each of those and puts them in a list.
        /// </summary>
        /// <param name="args">command-line args</param>
        /// <returns>The list of all the morphed words for that word</returns>
        public static List<string> Morph(string startWord)
        {
            // This is here so i can access the public WordList
            Program MOTO = new Program();

            List<string> morphIndi = new List<string>();
            // Creates an interger for a sublist and then loops through this, however loops through all strings within this sublist first.
            int subList;
            for (subList = 0; subList < MOTO.lines.Length; subList++)
            {
                if (MOTO.lines[subList].Length == startWord.Length)
                {
                    // Sets a check variable to zero each time it enters this if statement
                    int check = 0;

                    // checks all strings within the sublist loop.
                    for (int list = 0; list < startWord.Length; list++)
                    {
```

```
                // If the word doesn't equal the startword
                if (MOTO.lines[subList][list] != startWord[list])
                {
                    // checks to see how many times this loop has ran, if more than once breaks
                    // Else it adds one to the check counter.
                    if (check > 1)
                        break;
                    else
                        check++;
                }
            }

            // After it breaks from the loop and it sees that check is one it will add the string to the list.
            if (check == 1)
                morphIndi.Add(MOTO.lines[subList]);
        }
    }

    // After going through all of this it sends back this MorphIndi list for the word.
    return morphIndi;
}

/// <summary>
/// Uses the Morphed words from the previous method and sees which one match the final word and stores that path into a list
/// </summary>
```

```
    /// <param name="args">command-line args</param>
    /// <returns>The Final list of the morphed words to final word</returns>
    public static List<string> MorphChain(string startWord, string endWord, List<string> morphedList, int Length)
    {
        // Creates a function for morhping all the posbilities of a certain word
        List<string> morphIndi = Morph(startWord);

        // Adds the start word to the final output list
        morphedList.Add(startWord);

        // Checks to see if the individual morped list has the final word in it if it does it aadds the final word to the final list
        if (morphIndi.Contains(endWord))
        {
            morphedList.Add(endWord);
        }

        // The length is decreesing by 1 each time this function starts. and this checks if the Length has reached zero.
        // If so it will return the final list back to main function
        if (Length == 0)
            return morphedList;

        for (int i = 0; i < morphIndi.Count; i++)
        {
            // Checks to see if any words have been repeated.
```

```
            if (!(morphedList.Contains(morphIndi[i])))
            {
                // If none have, repeates this function
                MorphChain(morphIndi[i], endWord, morphedList, Length - 1);

                // This Checks to see if the lat word in the list is the end Word. If so it gets out of the loop.
                // if it isn't it removes the lat word and starts the loop again
                if (morphedList[((morphedList).Count) - 1] == endWord)
                    break;
                else
                    morphedList.RemoveAt((morphedList.Count) - 1);
            }
        }
        // Returns the final list.
        return morphedList;
    }


    /// <summary>
    /// The purpose of this program was be able to access and parse a txt file.
    /// The Program reads in a text file, in our case each line is a different word, the it reads in the text file as an array per line
    /// With this array of strings it asks the user to enter a task that he/she wants to do.
    /// The program should display the user's choices in a menu, and prompt the user for any needed input.
    /// List of tasks:
```

```
        ///     1. list all words
        ///     2. list rhyming words(words that end in a string specified by the user)
        ///     3. list scrabble words(words that are constructed from the letters specified by the user; note that a letter may be used multiple times only if it appears that many times  in the user list)
        ///     4. list morph words(words that differ from a specified word in only one letter)
        ///     5. Morph Chains
        /// </summary>
        /// <param name="args">command-line args</param>
        static void Main(string[] args)
        {
            Program MOTO = new Program();


            //Asking the User to pick the choice he wants to do.
            Console.WriteLine("Pick from the following:\n 1. All words\n 2.Rhyming words\n 3.Scrabble words\n 4.Morph words\n 5.Morph Chains\n 6.Quit\n");
            // This Reads and Parses the input for the choice.
            int choice = int.Parse(Console.ReadLine());
            //The next part is going to be 5 If statements for what the user picked
            //Choice one is printing all of the lines
            if (choice == 1)
            {
                Console.WriteLine("You Choose Print All Words\n");
                // For each Line in Lines print to screen
                foreach (string line in MOTO.lines)
                {
                    Console.WriteLine(line);
```

```
    }
}
//Choice two is finding the ryhming words and printing them.
if (choice == 2)
{
    //Asks the user to enter the ending they want to ryhme with and Reads their input
    Console.WriteLine("You Choose Ryhming Words\nEnter desiered ending string ");
    string ryhme = (Console.ReadLine());
    // Searchs through each line in the wordlist to see which word ends with the users input.
    foreach (string line in MOTO.lines)
    {
        // if the endings match it prints out the line.
        if (line.EndsWith(ryhme))
        {
            Console.WriteLine(line);
        }
    }
}
// This if statement is the scrabble function allowing the user to enter 7 random letters
// and for the program to find words with those letters.
if (choice == 3)
{
    //Asks the user to enter the 7 letters and Reads their input
    Console.WriteLine("You Choose Scrabble Words\nEnter the Random Letters ");
    string lettersOrg = (Console.ReadLine());
```

```
        //Giant loop which shuffles through all of the words in the list
        foreach (string line in MOTO.lines)
        {
            // I initlize bad here to show if the word has a letter that is not in the user input it will set bad to 1 which will then break the loop.
            int bad = 0;
            // I also coppied the user input into a new string so i still have the orginal when i alter the new one.
            string lettersNew = lettersOrg;
            // This goes through each letter in each word.
            foreach (char letter in line)
            {
                // Check to see if the letter is not in the user input.
                if (!(lettersNew.ToLower().Contains(letter)))
                {
                    // if it's not it breaks out of the loop and sets bad = 1;
                    bad = 1;
                    break;
                }
                // If it is in the loop it sets that charecter as NULL so it doesn't repeat.
                else
                {
                    var regex = new Regex(Regex.Escape(letter.ToString()));
                    lettersNew = regex.Replace(lettersNew, "\0", 1);
                }
            }
```

```
        // It Bad was not set to 1 it will print the word.
        if (bad == 0)
        {
            Console.WriteLine(line);
        }
    }
}
// Choice 4 is the Morph function, which finds all words that are off by only one letter.
if (choice == 4)
{
    int n;
    // This creates an array of all the letters in the alphabet.
    //char[] az = Enumerable.Range('a', 'z' - 'a' + 1).Select(i => (Char)i).ToArray();
    // Asks the user for the word he wants to morph and takes the user input
    Console.WriteLine("You Choose Morph Words\nEnter the Word you want to Morph ");
    string Morph = (Console.ReadLine());
    //This converts the word that the user inputed into a charecter array.
    char[] characters = Morph.ToCharArray();
    //Saves an orginal version of the Charecters so it does not get aleted.
    string comboOrg = new string(characters);
    // Creates another copy of the charecerters array.
    char[] charactersNew = Morph.ToCharArray();
    //Gets the length of the charecerters array for the loop.
    int length = characters.Length;
    //The Loop starts where the main portion of the function is held.
```

```
// It has the main loop which alters the word in the wordlist


foreach (string line in MOTO.lines)
{
   // This loop alters the letter in AZ array
   foreach (char let in MOTO.az)
   {
      //Resets the count to zero for each new letter.
      n = 0;
      // compares the count to the length of the charecerters array.
      while (n < length)
      {
         //changes the first letter of the array to the first letter in the az array.
         charactersNew[n] = let;
         // Converts the charecter array into a string.
         string combo = new string(charactersNew);
         //If the new word made is not the word that you started with it goes inside the loop
         if (!(combo.Equals(comboOrg)))
         {
            //If the new word matches a word in the word list it prints it out.
            if (line.Equals(combo))
            {
               Console.WriteLine(line);
            }
         }
```

```
                    // Changes the changed letter back to the original. Goes back and repeates the loop with the next letter.
                    charactersNew[n] = characters[n];
                    n++;
                }
            }
        }
    }
    if (choice == 5)
    {
        // The next few lines Simply Have the user input the start, end word and length
        Console.WriteLine("You Choose Morph Chains\nEnter start word:");
        string startWord = Console.ReadLine();
        Console.WriteLine("Enter end word:");
        string endWord = Console.ReadLine();
        Console.WriteLine("Enter Maximum Chain Length: ");
        int length = int.Parse(Console.ReadLine());

        // Subtracts 2 from the length
        length = length - 2;

        //Creates a new list for all the morphed words that lead up to the final word.
        List<string> morphedList = new List<string>();

        // Checks to see if the final word equals to the start word. if it does, it prints both of them.
        if (startWord == endWord)
```

```
    {
        Console.WriteLine(startWord);
        Console.WriteLine(endWord);
    }

    //Calls the morphcahin function that completes the main program.
    MorphChain(startWord, endWord, morphedList, length);

    // If the length of the morphed list is greater than the length specfied it does not print
    if (morphedList.Count > length + 2)
    {
        Console.WriteLine("No Solution with the given Length or Less");
    }

    //But if it is fine it prints all the strings in morphed list
    else
        {
            //Loops through all of the words
            for (int i = 0; i < morphedList.Count; i++)
            {
                Console.WriteLine(morphedList[i]);
            }
        }

}
```

```
            // If the User enters 5 It wil exit the program
            if (choice == 6)
            {
               Console.WriteLine("You Choose Quit. Bye Bye");
               System.Environment.Exit(1);
            }
            Console.ReadKey();
        }
    }
}
```