Ejercicio 1: Juego del ahorcado

Este ejercicio consiste en hacer un programa con las siguientes opciones:

- Opción Configuración que permita crear el fichero de texto sino existe e incrementar el repertorio de palabras (las palabras estarán una en cada línea)
- Opción jugar que permita jugar al clásico juego del ahorcado, que utilizará el archivo de texto de palabras.

1. **Opción Configuración**: Permitirá crear el fichero de texto con las palabras y añadir nuevas palabras.

Las palabras serán registradas en un archivo de texto utilizando un algoritmo de encriptación (Se adjunta en este documento al final). Utilizar la clase fichero añadiéndole los métodos de cifrado y descifrado. Cada vez que se escriba una línea en el fichero se deberá llamar previamente a la función de cifrar para obtener la palabra cifrada y después escribirla en el fichero

2. **Opción juego ahorcado**: tendrá tres opciones (Cargar Palabras, Jugar, Salir)

El programa cargará las palabras del fichero en un array. Teniendo en cuenta que las palabras están cifradas deberá utilizarse la clase fichero para leer cada palabra y descífrala antes de asignarla al array.

- Opción 1: Se debe seleccionar de forma aleatoria la palabra, para lo cual primero debe contar cuantas palabras hay y generar un número aleatorio entre 1 y el número de palabras que hay en el array. Esta opción se debe hacer con la función CargaPalabra(..), la cual debe devolver la palabra escogida de forma aleatoria por el programa.
- Opción 2: Esta opción permite empezar el juego. De esta opción no se sale hasta que haya adivinado la palabra o haya perdido (con 5 intentos). En cada intento de juego se va a pedir que introduzca un nuevo carácter y el programa debe ver si ese carácter está en la palabra o no. Los intentos se deben ir sumando, si el carácter no se encuentra en la palabra

Ejercicio 2:

A partir del ejercicio anterior modificar el método de guardar y leer fichero de frases de modo que se le aplique una encriptación a las palabras que se registraren el fichero y desencriptación cuando se leen

Funciones para obtener el cifrado y el descifrado

Primero agregar a nuestra clase o formulario la siguiente librería:

using System.Security.Cryptography; // Libreria de cifrado.

Crear nuestra llave de cifrado:

public string clave = "cadenadecifrado"; // Clave de cifrado. NOTA: Puede ser cualquier combinación de carácteres.

```
public string cifrar(string cadena) // Función para cifrar una cadena.

    {  byte[] llave; //Array donde guardaremos la llave para el cifrado 3DES.
       byte[] arreglo = UTF8Encoding.UTF8.GetBytes(cadena); //Array donde guardaremos la cadena a cifrar.

       // Ciframos utilizando el Algoritmo MD5.
       MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();
       llave = md5.ComputeHash(UTF8Encoding.UTF8.GetBytes(clave));
       md5.Clear();

       //Ciframos utilizando el Algoritmo 3DES.
       TripleDESCryptoServiceProvider tripledes = new TripleDESCryptoServiceProvider();
       tripledes.Key = llave;
       tripledes.Mode = CipherMode.ECB;
       tripledes.Padding = PaddingMode.PKCS7;
       ICryptoTransform convertir = tripledes.CreateEncryptor(); // Iniciamos la conversión de la cadena
       byte[] resultado = convertir.TransformFinalBlock(arreglo, 0, arreglo.Length); //Arreglo de bytes donde guardaremos
la cadena cifrada.
       tripledes.Clear();

       return Convert.ToBase64String(resultado, 0, resultado.Length); // Convertimos la cadena y la regresamos.
    }
```

```
public string descifrar(string cadena)
    {  byte[] llave;
       byte[] arreglo = Convert.FromBase64String(cadena); // Arreay donde guardaremos la cadena descovertida.

       // Ciframos utilizando el Algoritmo MD5.
       MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();
       llave = md5.ComputeHash(UTF8Encoding.UTF8.GetBytes(clave));
       md5.Clear();

       //Ciframos utilizando el Algoritmo 3DES.
       TripleDESCryptoServiceProvider tripledes = new TripleDESCryptoServiceProvider();
```

```
        tripledes.Key = llave;
        tripledes.Mode = CipherMode.ECB;
        tripledes.Padding = PaddingMode.PKCS7;
        ICryptoTransform convertir = tripledes.CreateDecryptor();
        byte[] resultado = convertir.TransformFinalBlock(arreglo, 0, arreglo.Length);
        tripledes.Clear();

        string cadena_descifrada = UTF8Encoding.UTF8.GetString(resultado); // Obtenemos la cadena
        return cadena_descifrada; // Devolvemos la cadena
    }
```