INFORME UT4 Problema2. DAVID GARCÍA VALERO

(Tuve desde el principio en cuenta el ejercicio opcional, estableciendo un método para que el usuario introduzca la longitud de la palabra GLOBAl y CONSTANTE secreta a adivinar)
Para hacer una buena estructura, he separado los métodos en tres clases: una clase para la creación de la palabra aleatoria, una clase para la lectura de input y otra clase para la generación de pistas/verificación resultado.

En la clase de creación de palabra hice dos métodos, uno para generar una letra aleatoria (Asigno un número aleatorio del 0 al 25 generado por Random para que este sea igual al número de posición de la letra del alfabeto) y otro para generar una palabra aleatoria (Invoco el método de letra aleatoria tantas veces como LONGITUDPALABRA y transformo el carácter en String, una vez hecho eso, concateno estos caracteres String en un String)

Más tarde, en la clase de lectura de input hice el método para leer una cadena de texto (un simple String que equivale a un input.nextLine) y después otro método para validar la entrada de la palabra del usuario (Es, en resumen, un bucle que se repite si encuentra una letra del abecedario en mayúscula en la cadena de texto, que también se repite si existe un número del 0 al 9 en la cadena de texto y que también se repetirá si existen más discrepancias entre el cómputo de letras actuales y letras del alfabeto actuales en el bucle for (al existir más discrepancias, existe un carácter raro, un espacio en blanco, un interrogante, algo que hace que haya una discrepancia de más))

Por último, una clase resolverRespuesta, en la que establezco los métodos respuestaCorrecta (Al introducir dos string, dirá si ambos son iguales en forma de boolean (String.equals)) y generarPista (Este método crea el String de la pista de una palabra, si el los caracteres actuales del bucle son iguales, concatenará al String "La respuesa es [" un '0'. Si la diferencia de la posición ascendente en el alfabeto de cada carácter es menor a 10 generará un 'X', y en cualquier otro caso generará un '-')

En el primer método de UT4Problema2 en el que el programa se inicia, invocará el método de crear una palabra secreta, a la que será asignada a un String palabraSecreta. Entonces se inicia un bucle, en el que creo un String el cual será igual a la invocación del método de leerRespuesta.respuesta y invoco el método generarPista (primero el string de la palabra secreta y segundo el string del input del usuario) para generar una pista. Invoco para el bucle el método resolverRespuesta.respuestaCorrecta; teniendo en cuenta los dos strings, si ambos siguen sin ser iguales, se repetirá el bucle, en caso contrario, termina el programa.

Este diseño es muy útil para la visión general del proyecto, hace que sea muy fácil captar errores, ver donde se hallan, para poder empezar a resolverlos, se podría decir que es un diseño eficiente. Y no solo eso, también a la hora de estructurar mentalmente el ámbito del proyecto hace que sea una tarea más amena, ligera y menos exhaustiva, pues se tarda menos en completar el programa con una estructura clara.

Sobre la programación modular, he aprendido que es una forma de trabajar realista y eficaz, útil para el mantenimiento del programa.

Durante el programa los únicos problemas que he encontrado han sido de índole lógica, pues he tenido que parar a pensar cómo representar los problemas a solucionar en código. Por ejemplo, para validar la entrada del usuario y que un carácter fuese imposible de introducir tuve que llegar a una conclusión después un proceso de prueba, error y debugging largo, hasta que simplemente me vino la idea de una buena solución de repente, cambiar unos signos y establecer un cálculo relacionado con el número total de las letras del alfabeto, la longitud de la palabra secreta y el número de equivalencias entre ambas palabras.