PRÁCTICA 1: PRUEBAS DE CAJA BLANCA

Verificación y Validación 2022-2023

Utilizando la clase **SingleLinkedListImpl**, los profesores hemos desarrollado la clase Editor, que consiste en un editor de líneas de texto. Esta clase lee el contenido de un fichero de texto y permite realizar distintas operaciones sobre el mismo. Por simplicidad, este editor supone que las palabras y números solo están separados por espacios en blanco, coma, punto, o salto de línea. Las distintas líneas del fichero de texto se cargan en una estructura creada al efecto: una lista de listas.

Además, hemos creado una clase **Grafo** con una serie de algoritmos propios. No hacemos mucho hincapié en explicar esta estructura y sus algoritmos, dado el gran conocimiento que los alumnos tienen sobre la materia de grafos (estudiada en Estructura de Datos y Algorítmia).

Sea exhaustivo a la hora de realizar las prácticas, estudie con calma qué hacen los métodos de los cuales se solicita su estudio, **pero no pierda de vista el resto de los métodos que no se piden analizar**. **Recuerde que la práctica no se entrega, no la vamos a evaluar, pero tendrá que hacer un examen de ella.**

El objetivo de la práctica es probar los métodos que se indican más abajo utilizando técnicas de caja blanca.

Requisitos previos a la realización de la práctica:

- Maven 3.6.3: https://downloads.apache.org/maven/maven-3/3.6.3/binaries/
- Java JDK 1.8 o superior
- IDE de desarrollo (Eclipse, Netbeans, IntelliJ, etc.)
- Tener una cuenta en Github (github.com)

Para poder llevar a cabo la práctica se debe configurar el entorno de trabajo de la siguiente manera:

1- Un único miembro del equipo debe hacer un "fork" al proyecto base disponible en https://github.com/VyV-UPM/202223-caja-blanca para crear una copia del mismo en su usuario de Github. Posteriormente debe añadir como colaboradores al resto de compañeros de equipo. Se puede hacer el repositorio privado.
2- Todos los miembros del equipo se deben descargar de Github el proyecto base forkeado usando Git y abrirlo en el IDE escogido (Eclispe, IntelliJ, etc.). También se puede descargar directamente desde el IDE.
3- Añadir las dependencias necesarias para poder usar Junit5 (https://junit.org/junit5/docs/current/user-guide/)
4- Para el editor, añada a las dependencias del proyecto la librería "singleList.jar" (disponible en Moodle) que contiene las clases implementadas que hemos mencionado anteriormente.

De la práctica Editor lo que se pide es realizar pruebas de caja blanca en la carpeta "src/test/java" sobre la clase Editor disponible en la carpeta "src/main/java/". Concretamente, hay que probar los siguientes métodos:

```
- public AbstractSingleLinkedListImpl<String> getLinea(int linea) throws
  EmptyCollectionException;
- public int numApariciones(int inicio, int fin, String palabra);
- public int numPalabras() throws EmptyCollectionException;
- public String palabraMasLarga ();
- public boolean existePalabra (String palabra);
- public void sustituirPalabra (String palabra, String nuevaPalabra);
```

De Grafo lo que se pide es realizar pruebas de caja blanca en la carpeta "src/test/java" sobre la clase Editor disponible en la carpeta "src/main/java/". Concretamente, hay que probar los siguientes métodos:

- public void **printListaAdyacentes**(int v);
- private boolean **todosVisitados**();
- public String **componentsRelated**();
- public String **BFS**(int vertice);
- public Grafo **kruskal**();

Para realizar las pruebas hay que tener en cuenta las siguientes instrucciones:

a) Se debe justificar la complejidad ciclomática así como los caminos independientes de cada una de los métodos a analizar. Para ello es necesario dibujar el grafo normalizado de las mismas.
b) Identificar los casos de prueba necesarios para probar cada método.
c) Si se encontrase algún nodo con varias condiciones compuestas, es necesario generar las tablas de decisión correspondientes así como dibujar el grafo considerando el nodo de dos formas.
    1. Considerándolo como un nodo complejo con varias condiciones.
    2. Dividiendo el nodo en tantos nodos como condiciones tenga.

    En caso de encontrar nodos con condiciones múltiples, determinar la complejidad ciclomática de c.1 y c.2, así como los caminos independientes.

d) Implementar las pruebas con Junit5 para soportar los casos de prueba descritos.
e) Asegure una cobertura mínima del 80%.
f) Comprobar que Editor y Grafo funcionan correctamente (Constructor, carga desde fichero, etc.). Además, si alguno de los métodos de estudio no funciona, estudie el porqué del fallo y la solución que propondría.
g) Para homogeneizar los grafos, numerar las líneas desde la cabecera del método hasta la llave del final empezando en cada uno de ellos desde el 1. Ejemplo:

```
1    public int numApariciones(int inicio, int fin, String palabra){
2          int i;
3          String aux;
4          ……
5    }
```

El desarrollo de esta práctica será evaluado mediante preguntas en el **examen final**.