**Enterprise Gear set**

**Plan de pruebas ciclo 1**

**Documentación de plan de pruebas para ciclo 1 de TSP**

2014

# Tabla de contenidos

Contenido

[Tabla de contenidos 2](#_Toc384729038)

[Objetivo del documento 3](#_Toc384729039)

[Casos de prueba 4](#_Toc384729040)

[Caso 1 4](#_Toc384729041)

[Objetivo 4](#_Toc384729042)

[Datos de entrada 4](#_Toc384729043)

[Pasos de la prueba 4](#_Toc384729044)

[Salida 4](#_Toc384729045)

[Caso 2 5](#_Toc384729046)

[Objetivo 5](#_Toc384729047)

[Datos de entrada 5](#_Toc384729048)

[Pasos de la prueba 5](#_Toc384729049)

[Salida 6](#_Toc384729050)

[Caso 3 6](#_Toc384729051)

[Objetivo 6](#_Toc384729052)

[Datos de entrada 6](#_Toc384729053)

[Pasos de la prueba 6](#_Toc384729054)

[Salida 7](#_Toc384729055)

[Caso 2 7](#_Toc384729056)

[Objetivo 7](#_Toc384729057)

[Datos de entrada 7](#_Toc384729058)

[Pasos de la prueba 8](#_Toc384729059)

[Salida 8](#_Toc384729060)

[Caso 4 8](#_Toc384729061)

[Objetivo 8](#_Toc384729062)

[Datos de entrada 8](#_Toc384729063)

[Pasos de la prueba 8](#_Toc384729064)

[Salida 9](#_Toc384729065)

[Caso 5 9](#_Toc384729066)

[Objetivo 9](#_Toc384729067)

[Datos de entrada 9](#_Toc384729068)

[Pasos de la prueba 9](#_Toc384729069)

[Salida 9](#_Toc384729070)

[Caso 6 9](#_Toc384729071)

[Objetivo 10](#_Toc384729072)

[Datos de entrada 10](#_Toc384729073)

[Pasos de la prueba 10](#_Toc384729074)

[Salida 10](#_Toc384729075)

[Caso 7 10](#_Toc384729076)

[Objetivo 10](#_Toc384729077)

[Datos de entrada 10](#_Toc384729078)

[Pasos de la prueba 10](#_Toc384729079)

[Salida 11](#_Toc384729080)

# Objetivo del documento

En el presente documento se plasma la documentación de la ejecución de pruebas unitarias sobre el sistema de conteo y monitoreo de LOC, el cual se conforma de los requerimientos que se encuentran en el documento: Software Requirements Specification.docx.

# Casos de prueba

Caso 1

Objetivo

Verificar la validación de lectura y recorrido de lectura de LOC comparando dos versiones de código que representan un método (parte) de una clase Java; como resultado de esta validación se encuentra el total de LOC añadidas y eliminadas. En esta prueba se revisa si logicamente se le indican varios índices en una misma posición.

Datos de entrada

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Actual | | |
| Índice | Contenido | Posición |
| 0 | Línea 34 | 0 |
| 1 | Línea 2 | 0 |
| 2 | Línea 5 | 0 |
| 3 | Línea 4 | 0 |

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Anterior | | |
| Índice | Contenido | Posición |
| 0 | Línea 1 | 0 |
| 1 | Línea 2 | 0 |
| 2 | Línea 3 | 0 |
| 3 | Línea 4 | 0 |

Nombre de función: functionTest

Pasos de la prueba

1. Se alimenta el sistema con dos arreglos que emulan el contenido del fichero leído el cual contiene un método (parte) en su estado anterior y un fichero que contiene el método en su estado actual.
2. El sistema realiza un conteo de las líneas eliminadas y las líneas añadidas, enviando como parámetro para cada calculo el arreglo del código anterior más el arreglo del código actual.
3. El sistema extrae el conteo total de LOC.
4. Se alimenta un objeto de resumen con los resultados.
5. Se presenta resultado al usuario final.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Total LOC | 4 | 4 |
| TotalLOCAdded | 2 | 2 |
| TotalLOCDeleted | 2 | 2 |
| Compare Name | functionTest | functionTest |

Caso 2

Objetivo

Verificar la validación de lectura y recorrido de lectura de LOC comparando dos versiones de código que representan un método (parte) de una clase Java; como resultado de esta validación se encuentra el total de LOC añadidas y eliminadas. En esta prueba se revisa lógicamente agregando líneas con posiciones en desorden en los indicies de la listas a comparar.

Datos de entrada

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Actual | | |
| Índice | Contenido | Posición |
| 0 | Línea 1 | 0 |
| 1 | Línea 2 | 1 |
| 2 | Línea 3 | 2 |
| 3 | Línea 4 | 4 |

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Anterior | | |
| Índice | Contenido | Posición |
| 0 | Línea 1 | 1 |
| 1 | Línea 3 | 2 |
| 2 | Línea 3 | 0 |
| 3 | Línea4 | 4 |

Nombre de función: functionTest2

Pasos de la prueba

1. Se alimenta el sistema con dos arreglos que emulan el contenido del fichero leído el cual contiene un método (parte) en su estado anterior y un fichero que contiene el método en su estado actual.
2. El sistema realiza un conteo de las líneas eliminadas y las líneas añadidas, enviando como parámetro para cada calculo el arreglo del código anterior más el arreglo del código actual.
3. El sistema extrae el conteo total de LOC.
4. Se alimenta un objeto de resumen con los resultados.
5. Se presenta resultado al usuario final.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Total LOC | 4 | 4 |
| TotalLOCAdded | 1 | 1 |
| TotalLOCDeleted | 1 | 1 |
| Compare Name | functionTest2 | functionTest2 |

Caso 3

Objetivo

Verificar la validación de lectura y recorrido de lectura de LOC comparando dos versiones de código que representan un método (parte) de una clase Java; como resultado de esta validación se encuentra el total de LOC añadidas y eliminadas. En esta prueba se revisa si logicamente se le indican varios índices en una misma posición.

Datos de entrada

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Actual | | |
| Índice | Contenido | Posición |
| 0 | Línea 6 | 0 |
| 1 | Línea 10 | 1 |
| 2 | Línea5 | 2 |
| 3 | Línea | 3 |

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Anterior | | |
| Índice | Contenido | Posición |
| 0 | Línea 6 | 0 |
| 1 | Línea 10 | 1 |
| 2 | Línea 5 | 2 |
| 3 | Línea4 | 3 |

Nombre de función: functionTest3

Pasos de la prueba

1. Se alimenta el sistema con dos arreglos que emulan el contenido del fichero leído el cual contiene un método (parte) en su estado anterior y un fichero que contiene el método en su estado actual.
2. El sistema realiza un conteo de las líneas eliminadas y las líneas añadidas, enviando como parámetro para cada calculo el arreglo del código anterior más el arreglo del código actual.
3. El sistema extrae el conteo total de LOC.
4. Se alimenta un objeto de resumen con los resultados.
5. Se presenta resultado al usuario final.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Total LOC | 4 | 4 |
| TotalLOCAdded | 2 | 2 |
| TotalLOCDeleted | 2 | 2 |
| Compare Name | functionTest2 | functionTest2 |

Caso 2

Objetivo

Verificar la validación de lectura y recorrido de lectura de LOC comparando dos versiones de código que representan un método (parte) de una clase Java; como resultado de esta validación se encuentra el total de LOC añadidas y eliminadas. En esta prueba se revisa lógicamente agregando líneas con posiciones en desorden en los indicies de la listas a comparar.

Datos de entrada

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Actual | | |
| Índice | Contenido | Posición |
| 0 | Línea 1 | 0 |
| 1 | Línea 2 | 1 |
| 2 | Línea 3 | 2 |
| 3 | Línea 4 | 4 |

|  |  |  |
| --- | --- | --- |
| Arreglo: Versión Anterior | | |
| Índice | Contenido | Posición |
| 0 | Línea 1 | 1 |
| 1 | Línea 3 | 2 |
| 2 | Línea 3 | 0 |
| 3 | Línea4 | 4 |

Nombre de función: functionTest2

Pasos de la prueba

1. Se alimenta el sistema con dos arreglos que emulan el contenido del fichero leído el cual contiene un método (parte) en su estado anterior y un fichero que contiene el método en su estado actual.
2. El sistema realiza un conteo de las líneas eliminadas y las líneas añadidas, enviando como parámetro para cada calculo el arreglo del código anterior más el arreglo del código actual.
3. El sistema extrae el conteo total de LOC.
4. Se alimenta un objeto de resumen con los resultados.
5. Se presenta resultado al usuario final.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Total LOC | 4 | 4 |
| TotalLOCAdded | 1 | 1 |
| TotalLOCDeleted | 1 | 1 |
| Compare Name | functionTest2 | functionTest2 |

Caso 4

Objetivo

Verificar la validación de lectura y conversión de un arreglo de string que conforma una lista de líneas físicas a líneas lógicas (LOC). Haciendo comparación de las líneas obtenidas versus las esperadas.

Datos de entrada

|  |  |
| --- | --- |
| Bloque Líneas 1 | |
| Índice | Contenido |
| 0 | line 1; |
| 1 | line 2; |
| 2 | line 3; |

Pasos de la prueba

1. Se alimenta el sistema con un bloque de líneas físicas en una lista
2. Se ejecuta en el sistema el llamado de extracción de líneas lógicas del bloque dado.
3. Una vez retorna se compara el total de líneas lógicas obtenidas
4. 4. Seguido se compara contenidos esperados en el nuevo arreglo de líneas lógicas (LOC) obtenidos.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Size | 3 | 3 |
| Content index 1 | line 1 | line 1 |

Caso 5

Objetivo

Verificar la validación de lectura y conversión de un arreglo de string que conforma una lista de líneas físicas a líneas lógicas (LOC). Haciendo comparación de las líneas obtenidas versus las esperadas.

Datos de entrada

|  |  |
| --- | --- |
| Bloque Líneas 1 | |
| Índice | Contenido |
| 0 | line 1; |
| 1 | line 2; |
| 2 | If(a) |
| 3 | { |
| 4 | content; |
| 5 | } |

Pasos de la prueba

1. Se alimenta el sistema con un bloque de líneas físicas en una lista
2. Se ejecuta en el sistema el llamado de extracción de líneas lógicas del bloque dado.
3. Una vez retorna se compara el total de líneas lógicas obtenidas
4. 4. Seguido se compara contenidos esperados en el nuevo arreglo de líneas lógicas (LOC) obtenidos.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Size | 6 | 6 |
| Content index 4 | content; | content; |

Caso 6

Objetivo

Verificar la funcionalidad de lectura y escritura de ficheros en el proyecto. En este caso se crea un archivo texto con líneas físicas de código determinadas en la lista de entrada.

Datos de entrada

|  |  |
| --- | --- |
| Bloque Líneas 1 | |
| Índice | Contenido |
| 0 | line 1; |
| 1 | line 2; |
| 2 | line 3; |

Ruta escritura: [path raíz del proyecto java] + “file.txt”

Pasos de la prueba

1. Se alimenta el sistema con un bloque de líneas físicas en una lista
2. Se construye una ruta real (directorio) del sistema operativo con un archivo llamado: file.txt
3. Se ejecuta la función de guardado del File Manager.
4. Validar si el archivo nuevo existe en el directorio.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| True | Si existe | Si existe |

Caso 7

Objetivo

Verificar la funcionalidad de lectura y escritura de ficheros en el proyecto. El archivo creado en el Caso 6 se usa como insumo para leer el contenido de este.

Datos de entrada

Ruta escritura: [path raíz del proyecto java] + “file.txt”

Pasos de la prueba

1. Se alimenta el sistema con un directorio y el nombre de archivo a leer.
2. Se valida si el archivo existe, si este no existe la prueba falla automáticamente.
3. Se ejecuta la función de lectura del File Manager.
4. Después d ela lectura, se elimina archivo, ya que es solo para pruebas.

Salida

|  |  |  |
| --- | --- | --- |
| Assert | Esperado | Encontrado |
| Size | 3 | 3 |
| Content index 1 | line 2; | line 2; |