**TECNOLÓGICO DE COSTA RICA**

**Escuela de Ingeniería en Computación**

**Proyecto de Ingeniería de Software**

**Profesora:**

María Estrada Sánchez

**Entrega 3:**

Analizador Contextual:

Plan de pruebas, validación y verificación.

**Estudiantes:**

Christian León Guevara - 2013371982

Gabriel Ramírez Ramírez - 201020244

**Fecha de entrega:**

20-01-2019

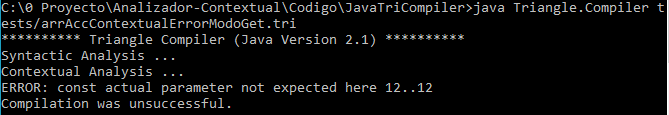
**Período Verano**

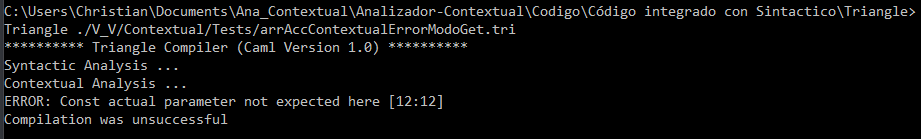
**Cartago**

1. **Ejecución de pruebas de los casos propuestos.**

**Caso 1. arrAccContextualErrorModoGet.tri**

Resultado en JAVA.



Resultado en OCAML. 

Análisis de resultado

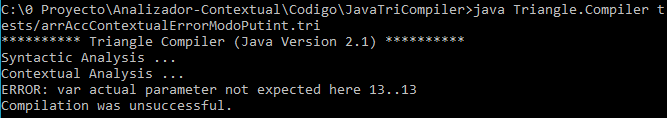
* En ambos compiladores se presentan el mismo error haciendo referencia a la posición en el documento

Resultado de prueba

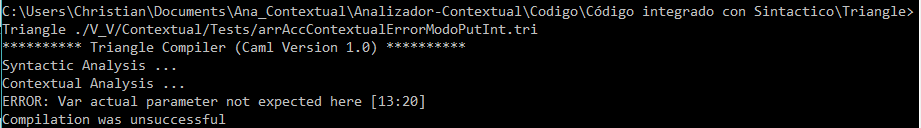
* La prueba es exitosa

**Caso 2. arrAccContextualModoPutInt**

Resultado en Java.



Resultado en OCAML.



Análisis de resultado

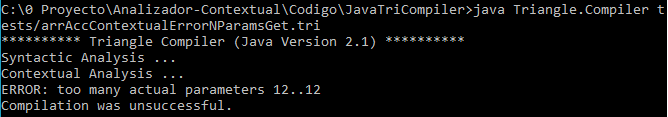
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

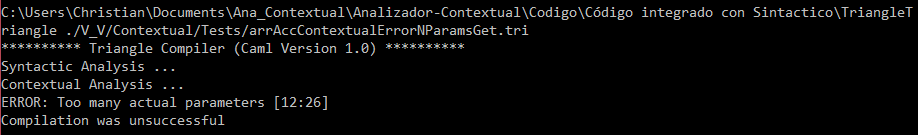
* La prueba es exitosa

**Caso 3. arrAccContextualNParamsGet**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

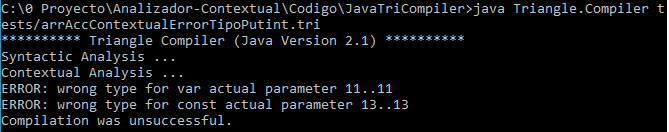
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

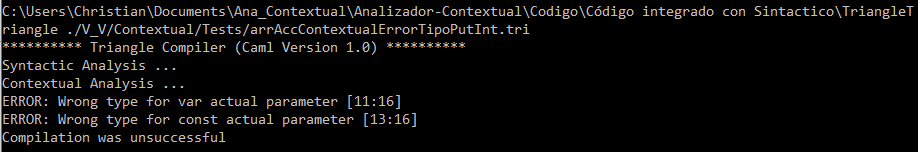
* La prueba es exitosa

**Caso 4. arrAccContextualTipoPutInt**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

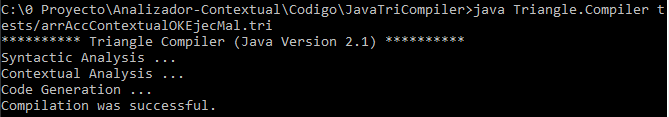
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

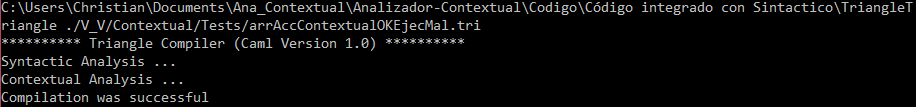
* La prueba es exitosa

**Caso 5. arrAccContextualOKEjecMal**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

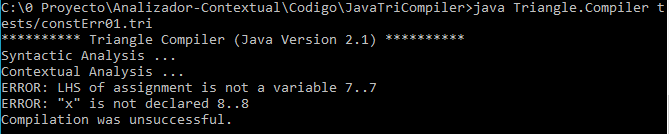
* En ambos compiladores se no se presentan errores.

Resultado de prueba

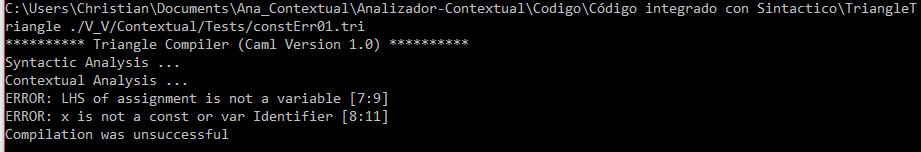
* La prueba es exitosa

**Caso 6. constErr01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

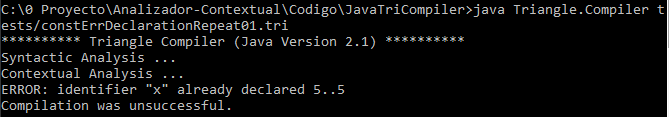
* En ambos compiladores se presentan los mismos errores, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

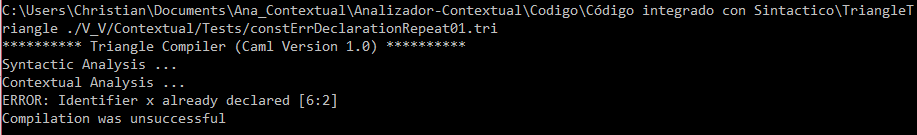
* La prueba es exitosa

**Caso 7. constErrDeclarationRepeat01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

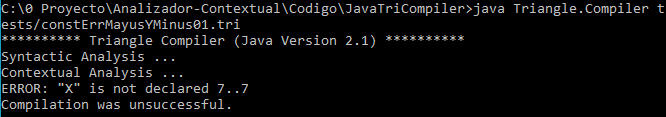
* En ambos compiladores se presentan el mismo error, tanto la columna como la fila están incorrectas, pero esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

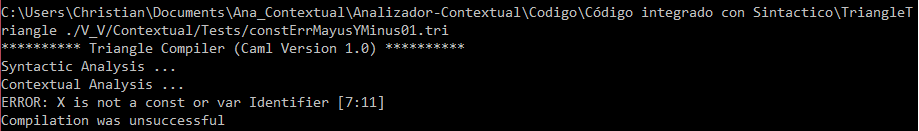
* La prueba es exitosa

**Caso 8. constErrMayusYMinus01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

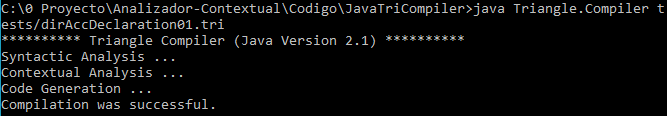
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

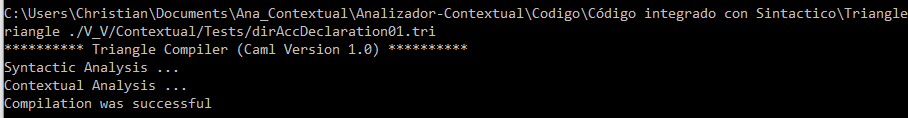
* La prueba es exitosa

**Caso 9. dirAccDeclaration01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

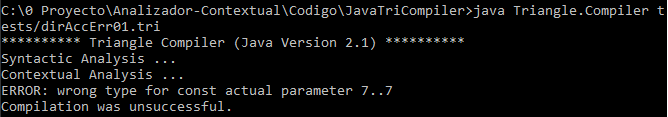
* En ambos compiladores se no se presentan errores.

Resultado de prueba

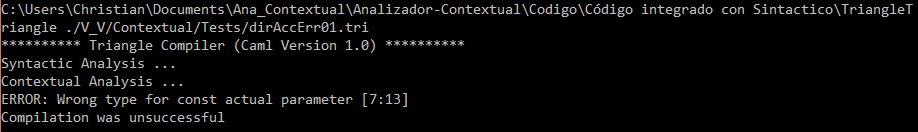
* La prueba es exitosa

**Caso 10. dirAccErr01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

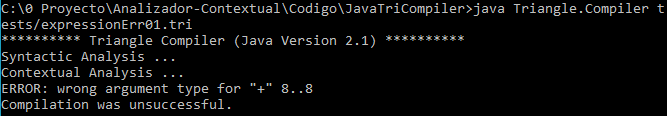
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

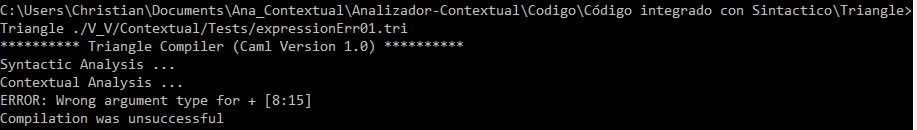
* La prueba es exitosa

**Caso 11. expressionErr01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

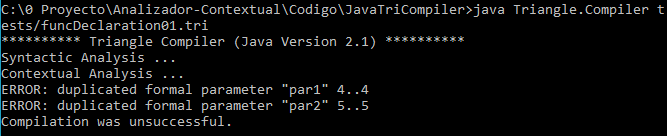
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

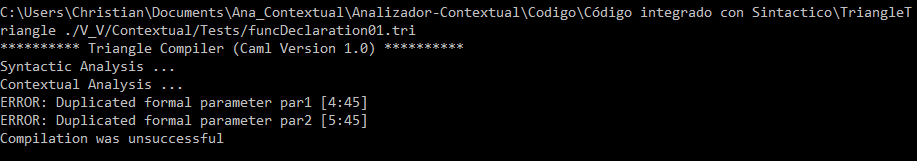
* La prueba es exitosa

**Caso 12. funcDeclaration01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

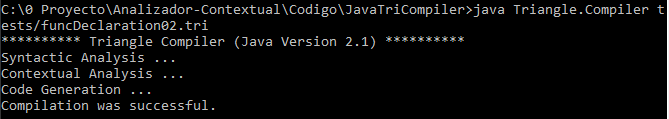
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra columna esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

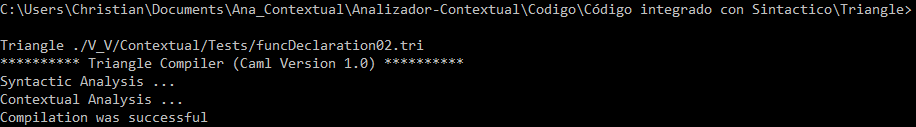
* La prueba es exitosa

**Caso 13. funcDeclaration02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

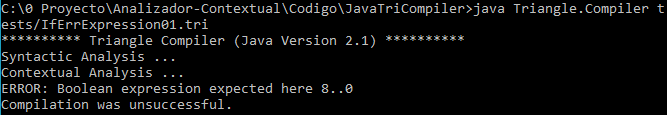
* En ambos compiladores se no se presentan errores.

Resultado de prueba

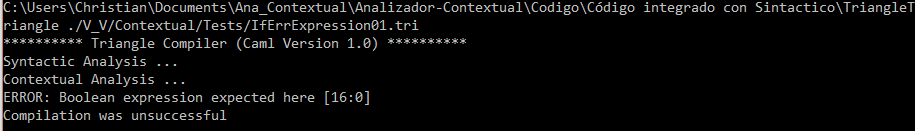
* La prueba es exitosa

**Caso 14. IfErrExpression01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

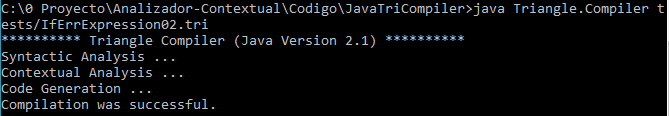
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

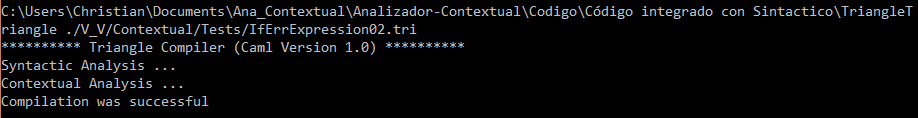
* La prueba es exitosa

**Caso 15. IfErrExpression02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

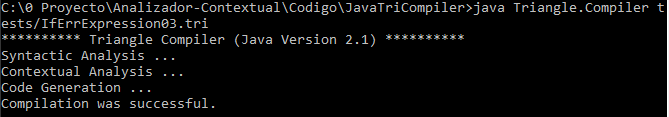
* En ambos compiladores se no se presentan errores.

Resultado de prueba

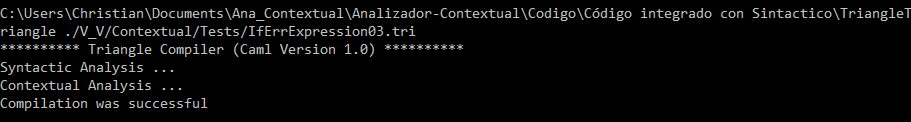
* La prueba es exitosa

**Caso 16. IfErrExpression03**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

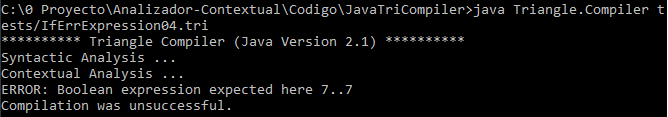
* En ambos compiladores se no se presentan errores.

Resultado de prueba

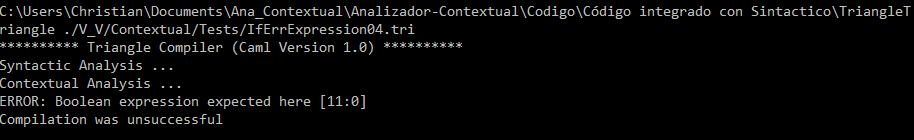
* La prueba es exitosa

**Caso 17. IfErrExpression04**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

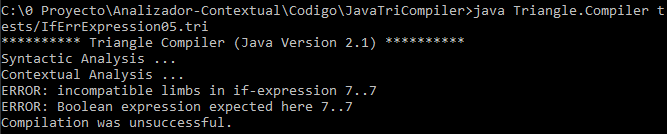
* En ambos compiladores se presentan el mismo error, tanto la columna como la fila están incorrectas, pero esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

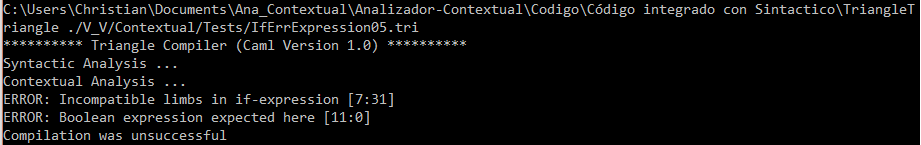
* La prueba es exitosa

**Caso18. IfErrExpression05**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

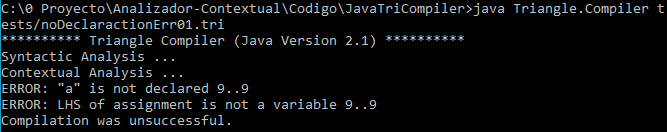
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

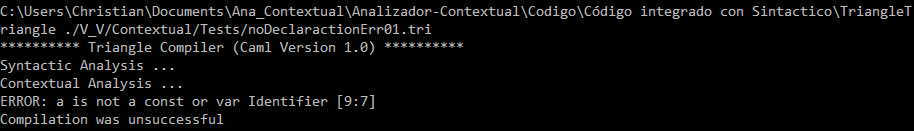
* La prueba es exitosa

**Caso 19. noDeclaractionErr01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

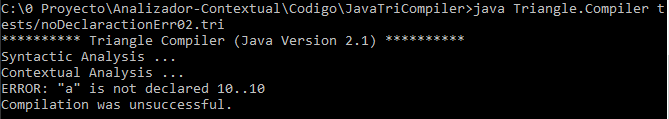
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

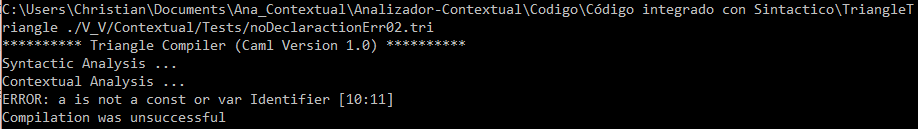
* La prueba es exitosa

**Caso 20. noDeclaractionErr02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

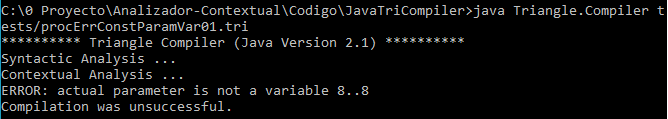
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

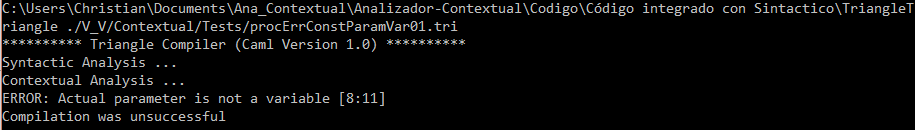
* La prueba es exitosa

**Caso 21. procErrConstParamVar01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

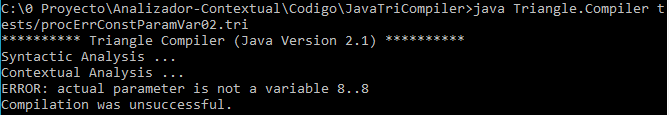
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

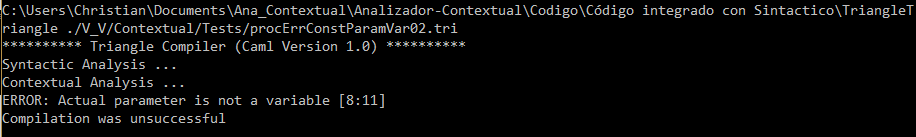
* La prueba es exitosa

**Caso 22. procErrConstParamVar02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

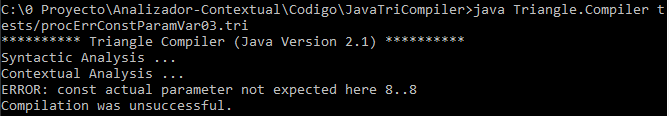
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

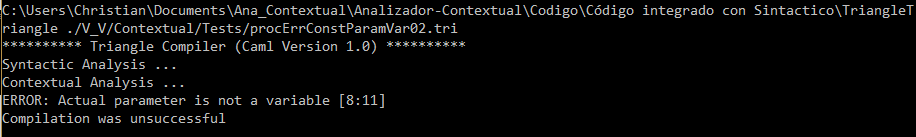
* La prueba es exitosa

**Caso 23. procErrConstParamVar03**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

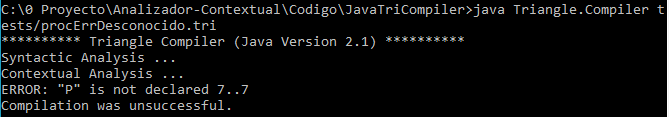
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

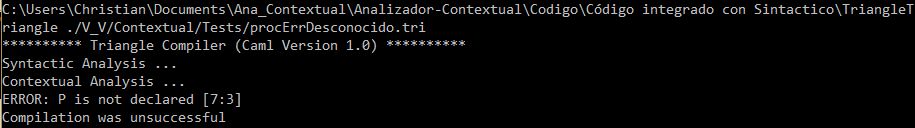
* La prueba es exitosa

**Caso 24. procErrDesconocido**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

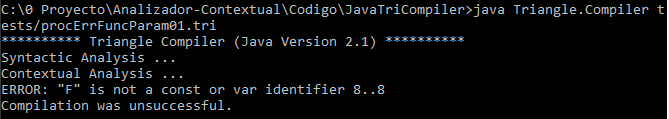
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

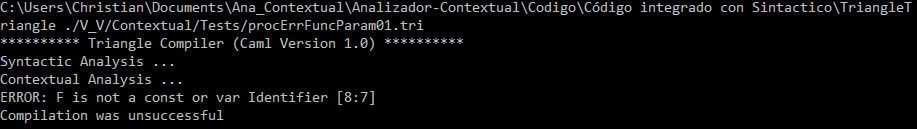
* La prueba es exitosa

**Caso 25. procErrFuncParam01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

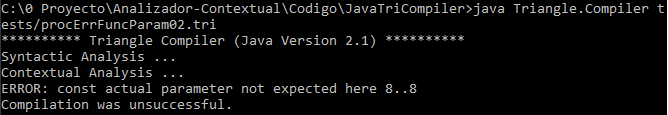
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

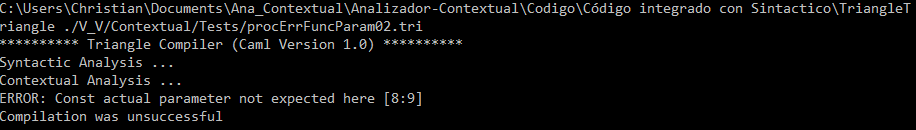
* La prueba es exitosa

**Caso 26. procErrFuncParam02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

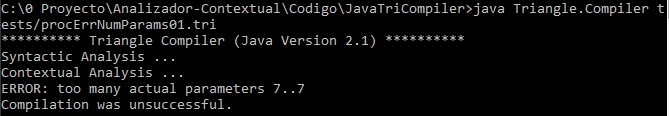
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

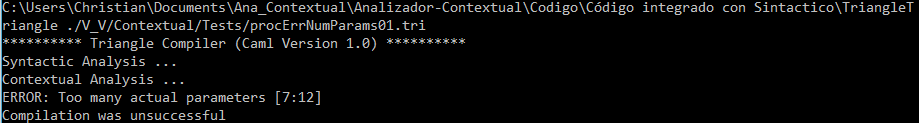
* La prueba es exitosa

**Caso 27. procErrNumParams01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

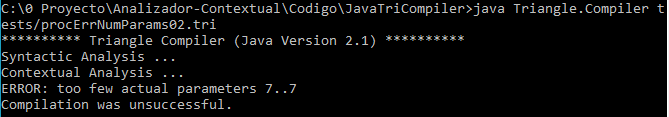
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

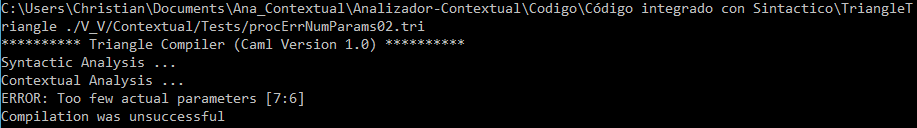
* La prueba es exitosa

**Caso 28. procErrNumParams02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

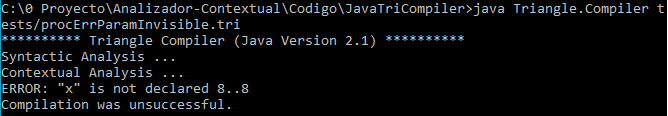
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

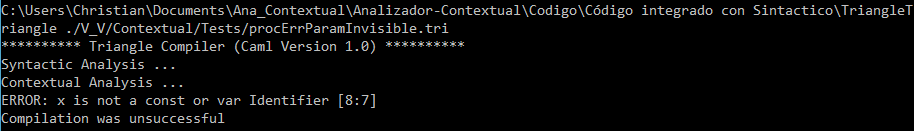
* La prueba es exitosa

**Caso 29. procErrParamInvisible**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

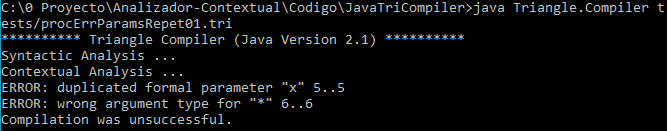
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

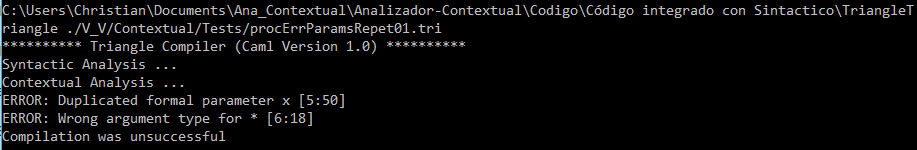
* La prueba es exitosa

**Caso 30. procErrParamsRepet01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

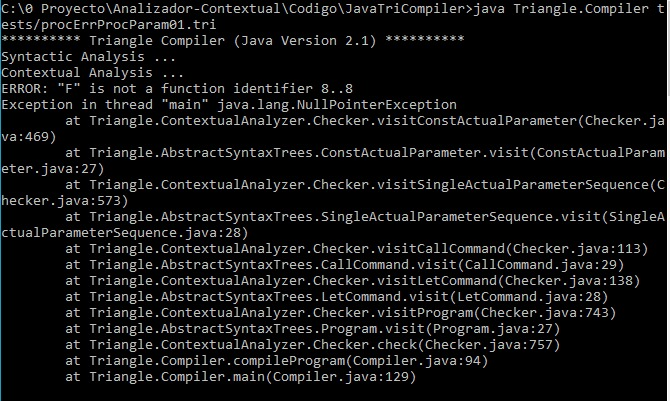
Resultado de prueba

* La prueba es exitosa

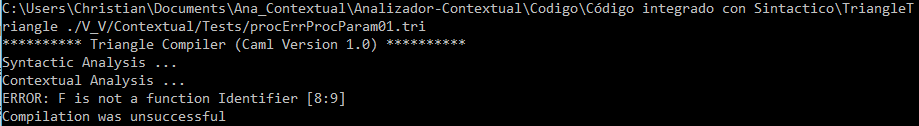
**Caso 31. procErrProcParam01**

Resultado en JAVA.

Al realizar la ejecución de este caso en el compilador de Java se presentó un error y no se realiza el análisis contextual por tal motivo no se ha colocado dicha evidencia.



Resultado en OCAML.



Análisis de resultado

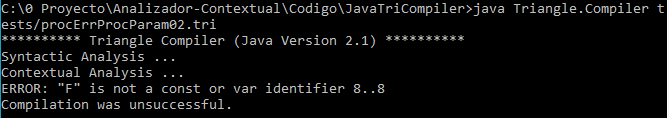
* No pudo realizar

Resultado de prueba

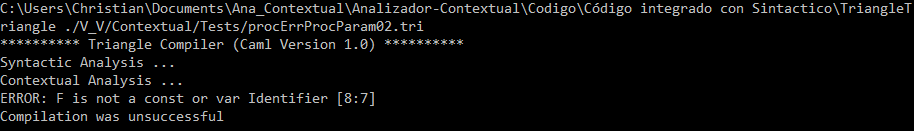
* Prueba inválida

**Caso 32. procErrProcParam02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

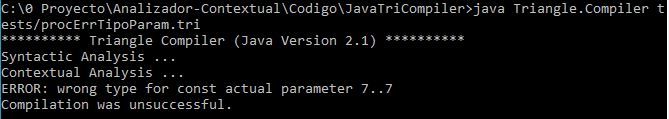
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

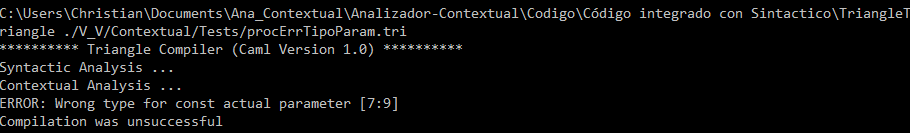
* La prueba es exitosa

**Caso 33. procErrTipoParam**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

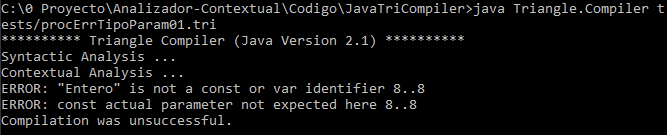
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

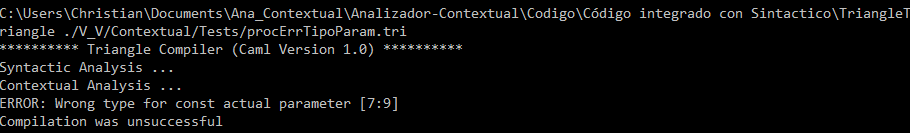
* La prueba es exitosa

**Caso 34. procErrTipoParam01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

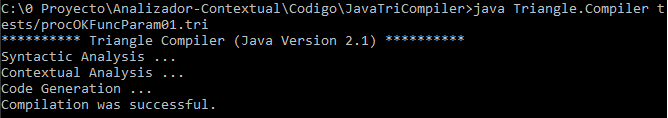
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

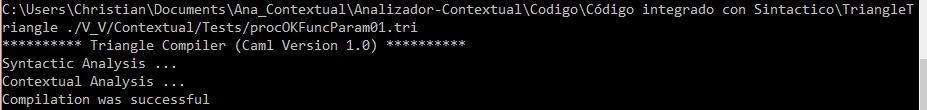
* La prueba es exitosa

**Caso 35. procOKFuncParam01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

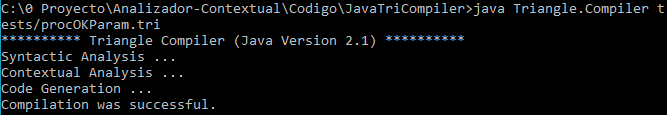
* En ambos compiladores se no se presentan errores.

Resultado de prueba

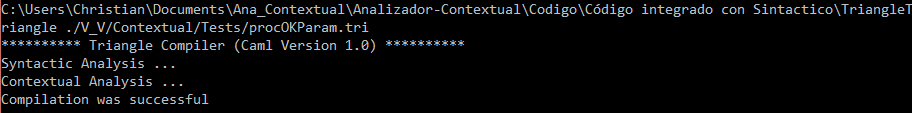
* La prueba es exitosa

**Caso 36. procOKParam**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

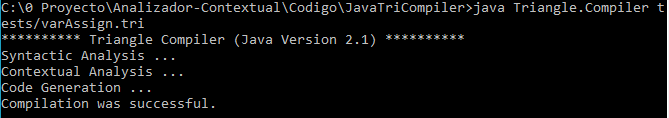
* En ambos compiladores se presentan el mismo error, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

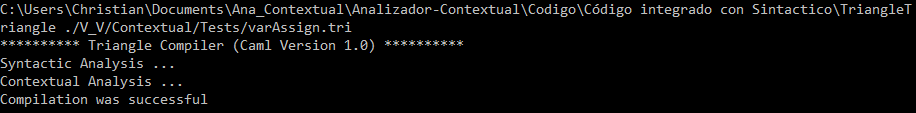
* La prueba es exitosa

**Caso 37. varAssign**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

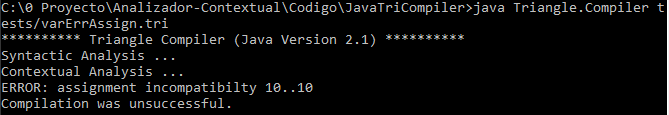
* En ambos compiladores se no se presentan errores.

Resultado de prueba

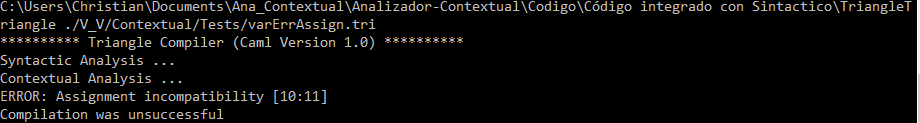
* La prueba es exitosa

**Caso 38. varErrAssign**

Resultado en JAVA.

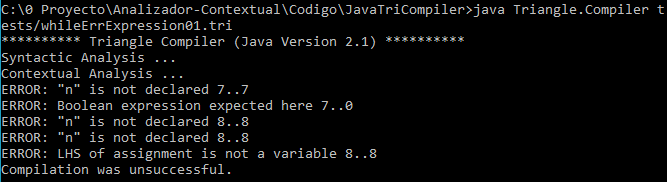


Resultado en OCAML.

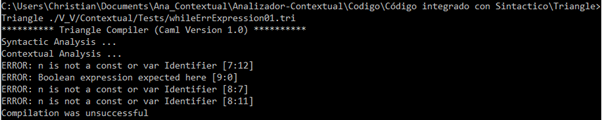


**Caso 39. whileErrExpression01**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

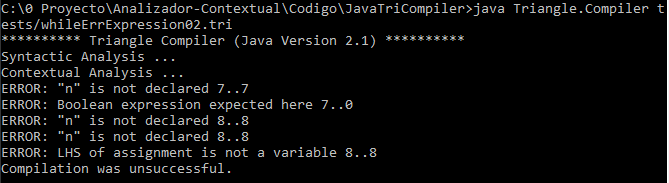
* En ambos compiladores se presentan los mismos errores, pero aun que hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador sintáctico no es responsable de ello.

Resultado de prueba

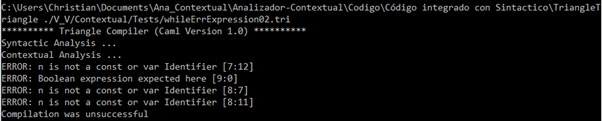
* La prueba es exitosa

**Caso 40. whileErrExpression02**

Resultado en JAVA.



Resultado en OCAML.



Análisis de resultado

* En ambos compiladores se presentan los mismos errores, pero aunque hace referencia a otra fila esto se debe a cuando se leen los tokens del árbol. El analizador contextual no es responsable de ello.

Resultado de prueba

* La prueba es exitosa

1. **Ejecución de las revisiones validación y verificación del código.**
   1. **Lista de cotejo de legibilidad.**

**Checkel.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  |  |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  |  |
| Se respetó el ancho de 80 columnas dentro del código. | X |  | Algunas líneas se dejaron excediendo el ancho de 80 columnas. |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |

**IdentificationTable.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  |  |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  |  |
| Se respetó el ancho de 80 columnas dentro del código. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |

**IdEntry.mli**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  | Se describe los autores y fecha de modificación. |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. |  | X |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  | Hace referencia al nombre de la entrada de la tabla de identificación. |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. |  | X |  |
| Se respetó el ancho de 80 columnas dentro del código. |  | X | Es una única declaración por lo que se dejó en un ancho superior a las 80 columnas. |
| Se rompieron líneas de código muy extensas. |  | X |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |

**Ast.mli**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  |  |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  | En algunos casos no se respeta la identación de 2 espacios establecida en el estándar. |
| Se respetó el ancho de 80 columnas dentro del código. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |

**ErrorReporter.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  | Existe un error en el nombre de la función write\_x\_m\_l\_errors. |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  |  |
| Se respetó el ancho de 80 columnas dentro del código. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |

**IdentificationTablePrinter\_XML.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  |  |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  |  |
| Se respetó el ancho de 80 columnas dentro del código. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |

**TreeDrawer.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Presenta un encabezado en los archivos con los derechos de propiedad y los autores. | X |  |  |
| Las importaciones se colocan al comienzo del archivo. | X |  |  |
| El nombre de las funciones es significativo y descriptivo. | X |  |  |
| El nombre de las variables e identificadores es significativo y descriptivo. | X |  | Existe un error en el nombre de la función write\_x\_m\_l\_tree. |
| Se mantiene el nivel de identación de 2 espacios establecido en el estándar de estilos para Ocaml. | X |  |  |
| Se respetó el ancho de 80 columnas dentro del código. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |

* 1. **Lista de cotejo de comprensibilidad.**

**Checkel.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. | X |  |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. |  | X |  |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  | Algunas líneas de código superan el ancho de 80 columnas. |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |

**IdentificationTable.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. | X |  |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. |  | X |  |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |

**IdEntry.mli**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. |  | X |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. |  | X |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. |  | X |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. |  | X |  |
| Existe una estandarización de los nombres. |  | X |  |
| Se mantuvo la identación de 2 espacios. |  | X |  |
| Se rompieron líneas de código muy extensas. |  | X |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. |  | X |  |

**Ast.mli**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. | X |  |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. |  | X |  |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |

**ErrorReporter.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. | X |  |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. | X |  | Existe un error en la definición de la función write\_x\_m\_l\_errors. |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |

**IdentificationTablePrinter\_XML.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. |  | X |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. |  | X |  |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. |  | X |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |

**TreeDrawer.ml**

Fecha: 21 de enero de 2019.

Nombre del revisor: Gabriel Ramírez Ramírez.

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspecto de revisión** | **Presente** | **No presente** | **Observaciones** |
| Dispone un encabezado para las funciones y lo que estas realizan. | X |  |  |
| El nombre de las funciones se establece mediante el estilo snake\_case. | X |  |  |
| El nombre de las variables e identificadores se establecen mediante el estilo snake\_case. | X |  |  |
| El nombre de los Constructores se establece mediante el estilo Snake\_case. | X |  |  |
| Los acrónimos técnicos se escribieron todo en mayúscula. | X |  | Existe un error en la definición de la función write\_x\_m\_l\_tree. |
| Existe una estandarización de los nombres. | X |  |  |
| Se mantuvo la identación de 2 espacios. | X |  |  |
| Se rompieron líneas de código muy extensas. | X |  |  |
| Se utilizó la agrupación e identación utilizando paréntesis. | X |  |  |
| Existen comentarios innecesarios. |  | X |  |
| Existen comentarios de código que ha sido eliminado. |  | X |  |
| Relación lógica del código en relación a la documentación. | X |  |  |