CASO: HOJA DE CÁLCULO RUDIMENTARIA

Algoritmos y estructuras de datos

INTEGRANTES

Martel Garay, Erick Akim 20240605I

Saavedra Nolberto, Dhemiz Thyago 20240456C

1.Caso de estudio

Escriba un programa de hoja de cálculo rudimentario. Despliegue una cuadrícula de celdas con las columnas A a H y las filas 1 a 20. Acepte la entrada en la primera fila de la pantalla. Los comandos contienen la forma columna fila entrada, donde *entrada* es ya sea un número, la dirección de una celda precedida por un signo más (por ejemplo, +A5), una cadena o una función precedida por un signo arroba, @. Las funciones son max, min, avg y sum. Durante la ejecución de su programa, construya y modifique el **grafo** que refleja la situación en la hoja de cálculo. Muestre los valores apropiados en las celdas adecuadas. Si el valor de una celda se actualiza, entonces los valores de todas las celdas que dependen de ella también deben modificarse. Por ejemplo, después de la secuencia siguiente de entradas:

A1 10
B1 20
A2 30
D1 +A1
C1 @sum(A1..B2)
D1 +C1

las dos celdas C1 y D1 deben mostrar el número 60.

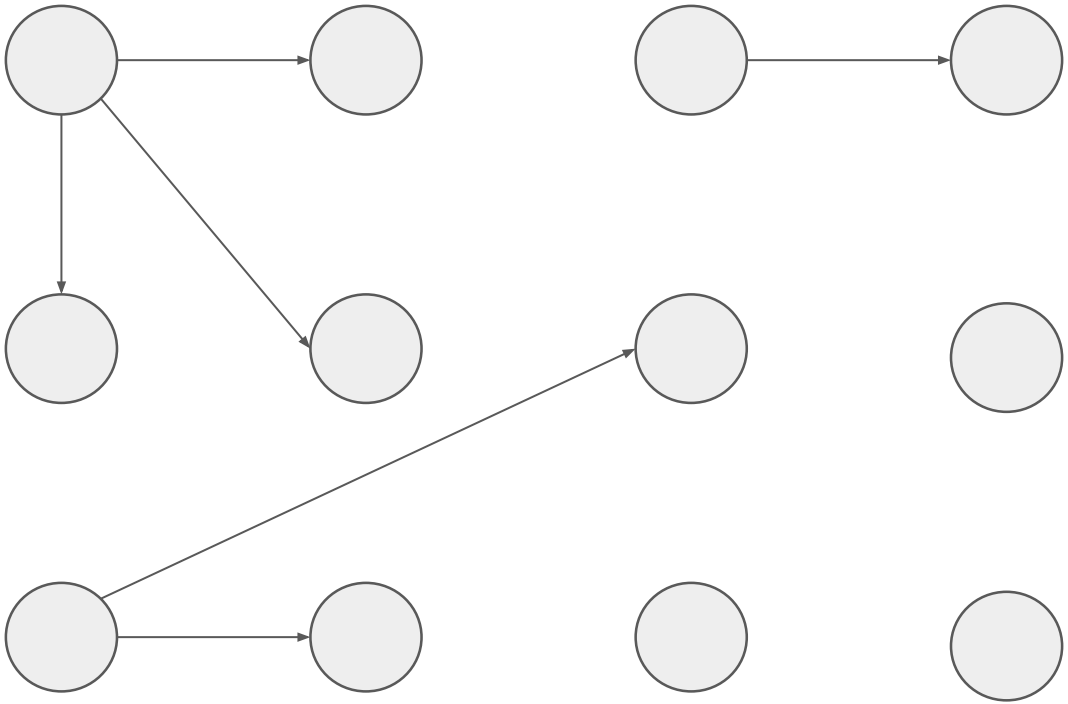
Considere usar una modificación del intérprete del capítulo 5 como una mejora de esta hoja de cálculo de manera que las expresiones aritméticas también podrían usarse para introducir valores, tales como

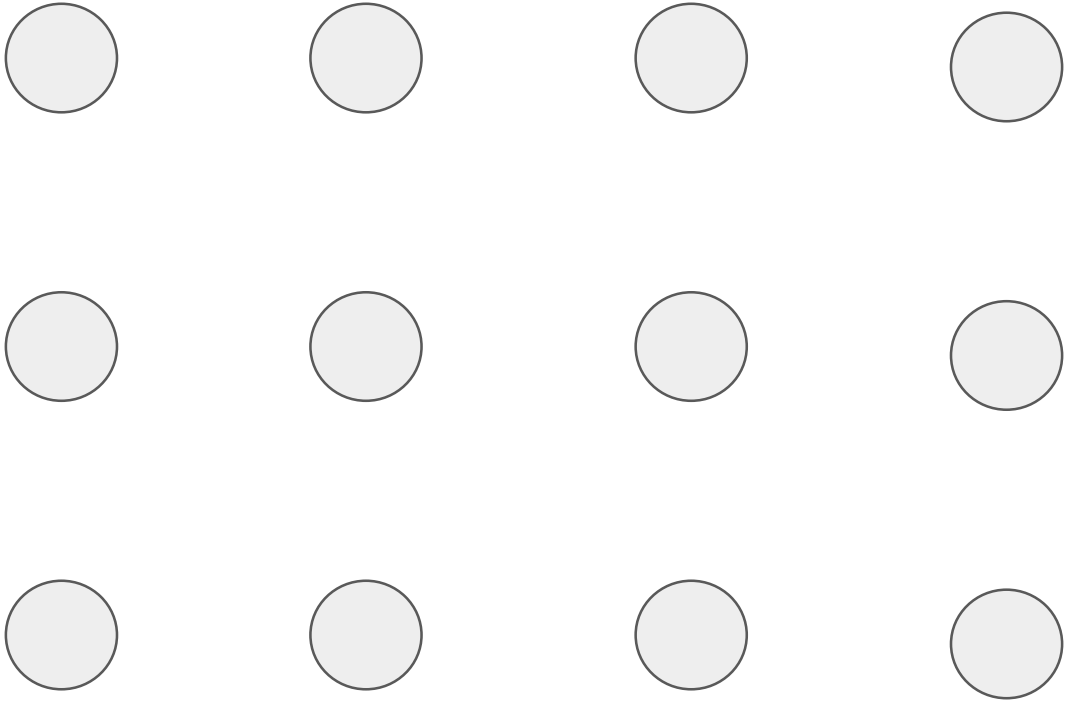
C3 2*A1
C4 @max(A1..B2) - (A2 + B2)

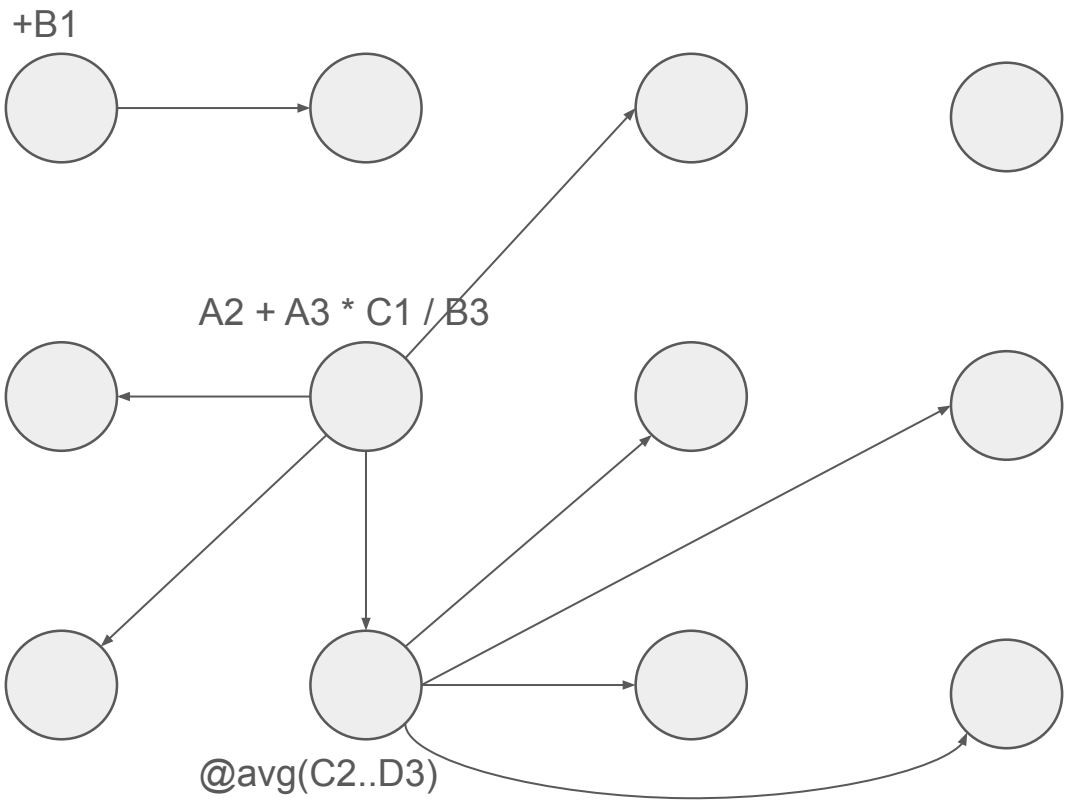
Objetivos del Programa

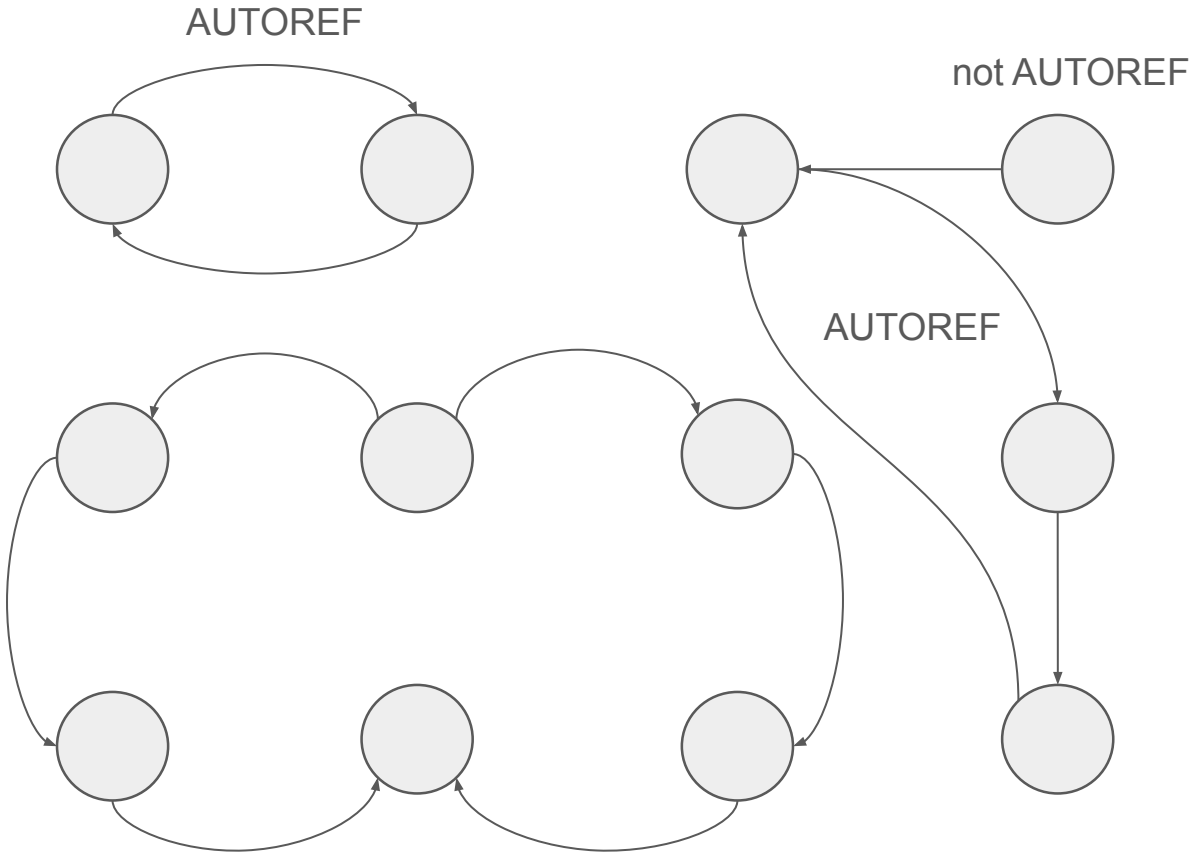
- Actualización instantánea de valores de celdas relacionadas.
- Manejo riguroso de expresiones (valores no planos).
  (+): referencia
  contiene(+-*/@): operación aritmética
  otro: literal
- Validación de celda destino y expresiones.
- Evaluación de expresiones.
- Manejo de ciclos (autorreferencias).

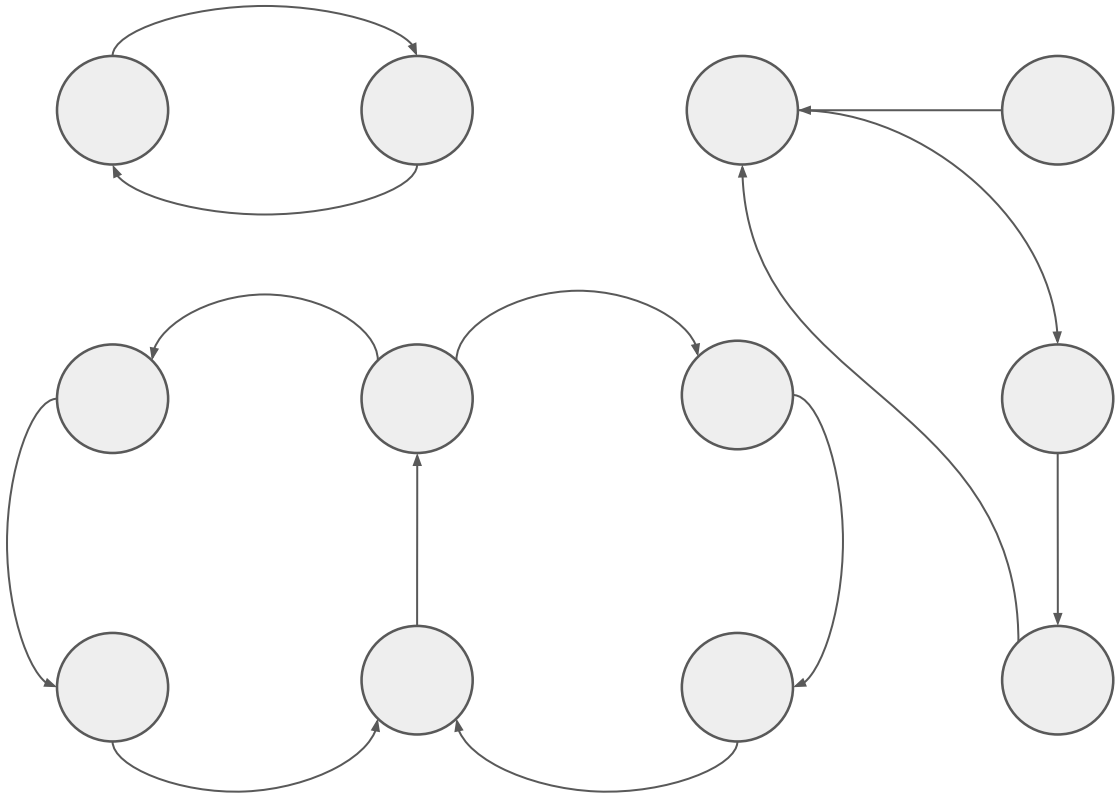
Estructura de datos

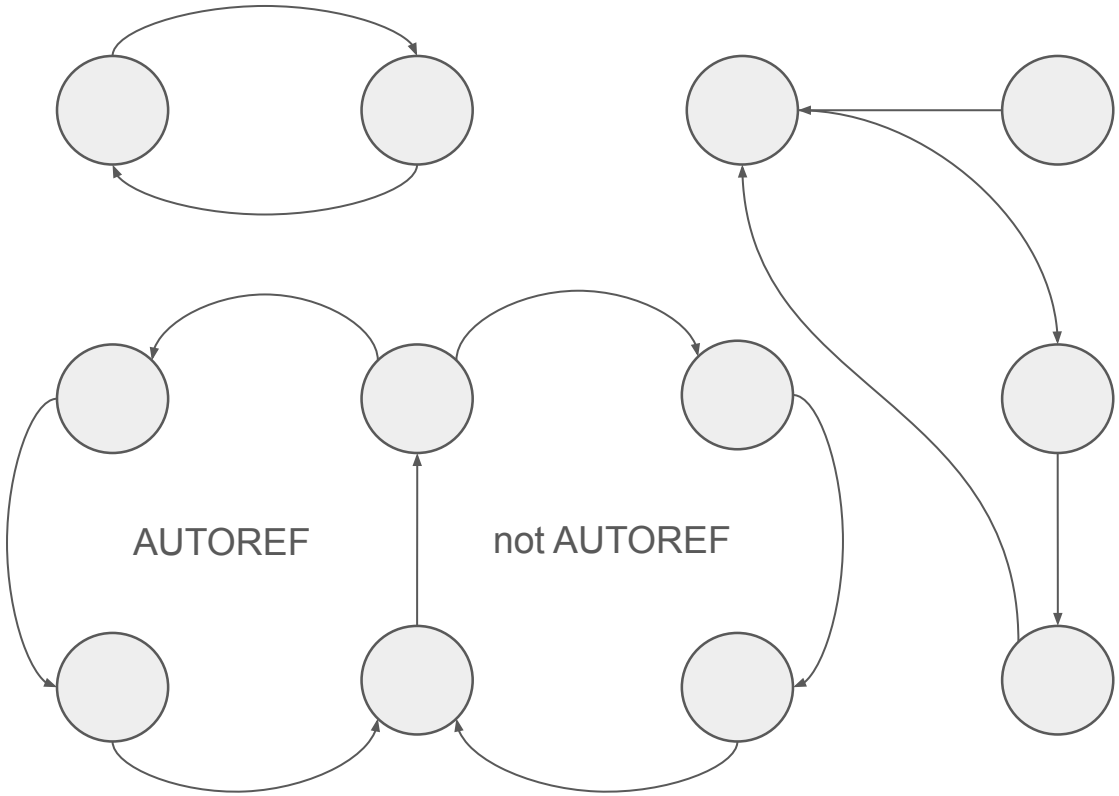


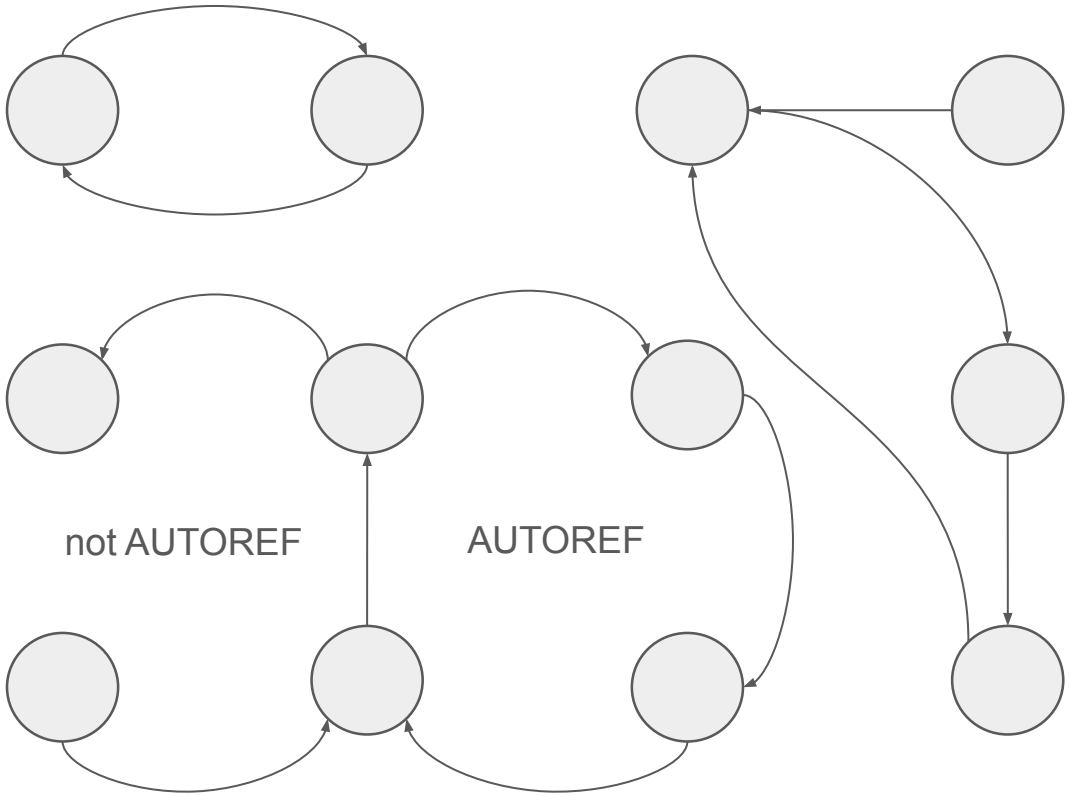


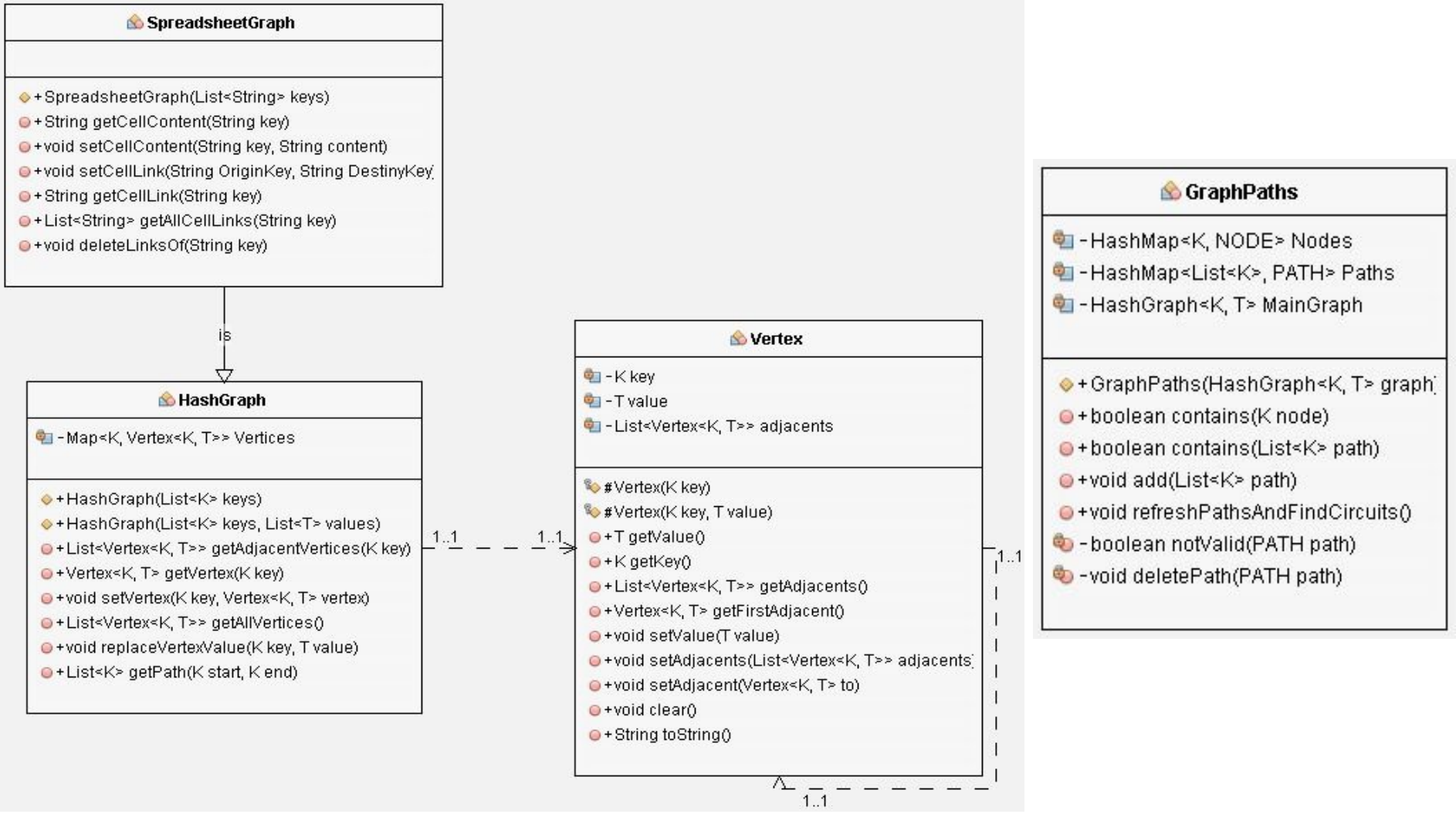

+B1
A2 + A3 * C1 / B3
@avg(C2..D3)

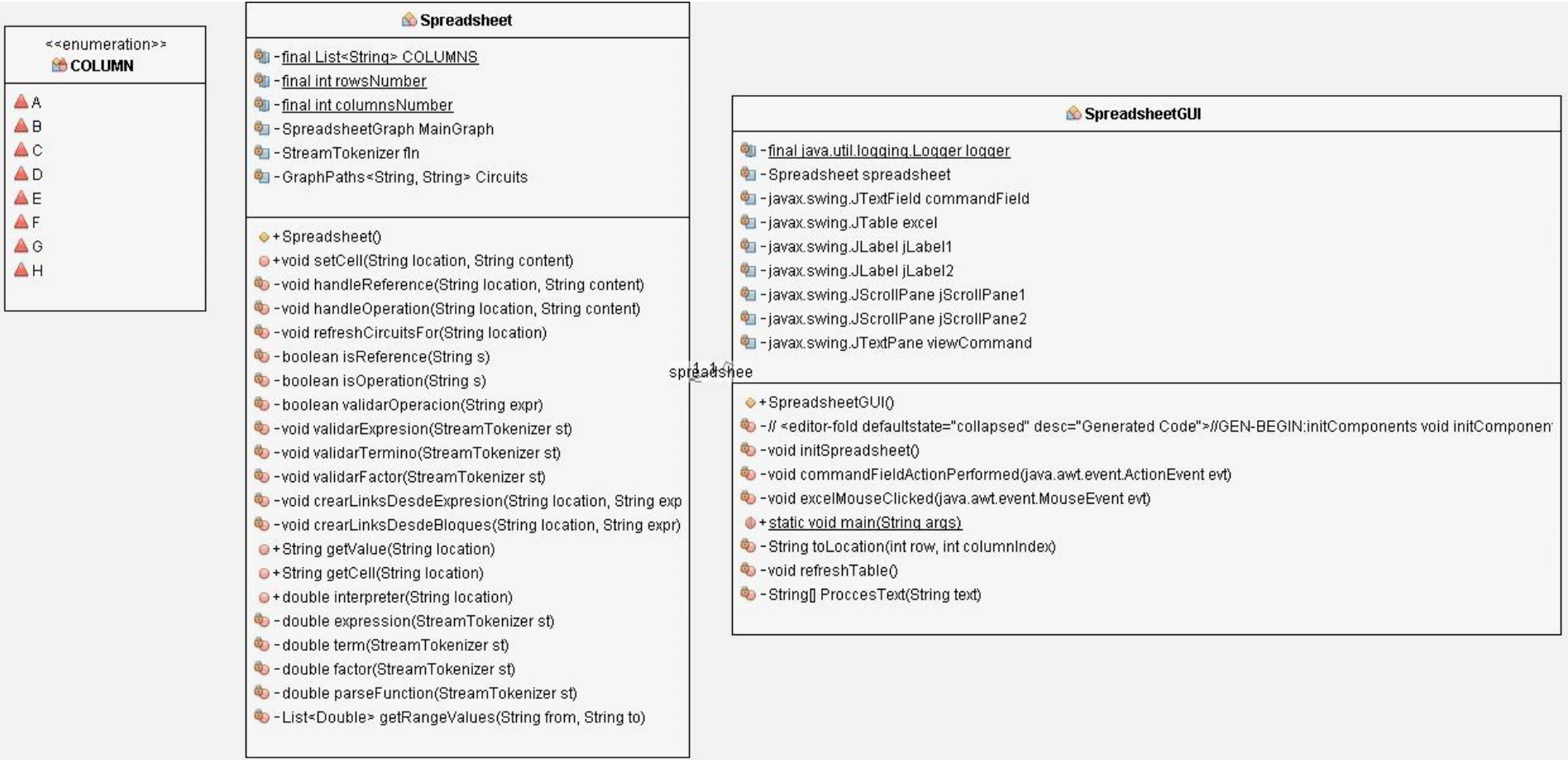

AUTOREF
not AUTOREF
AUTOREF

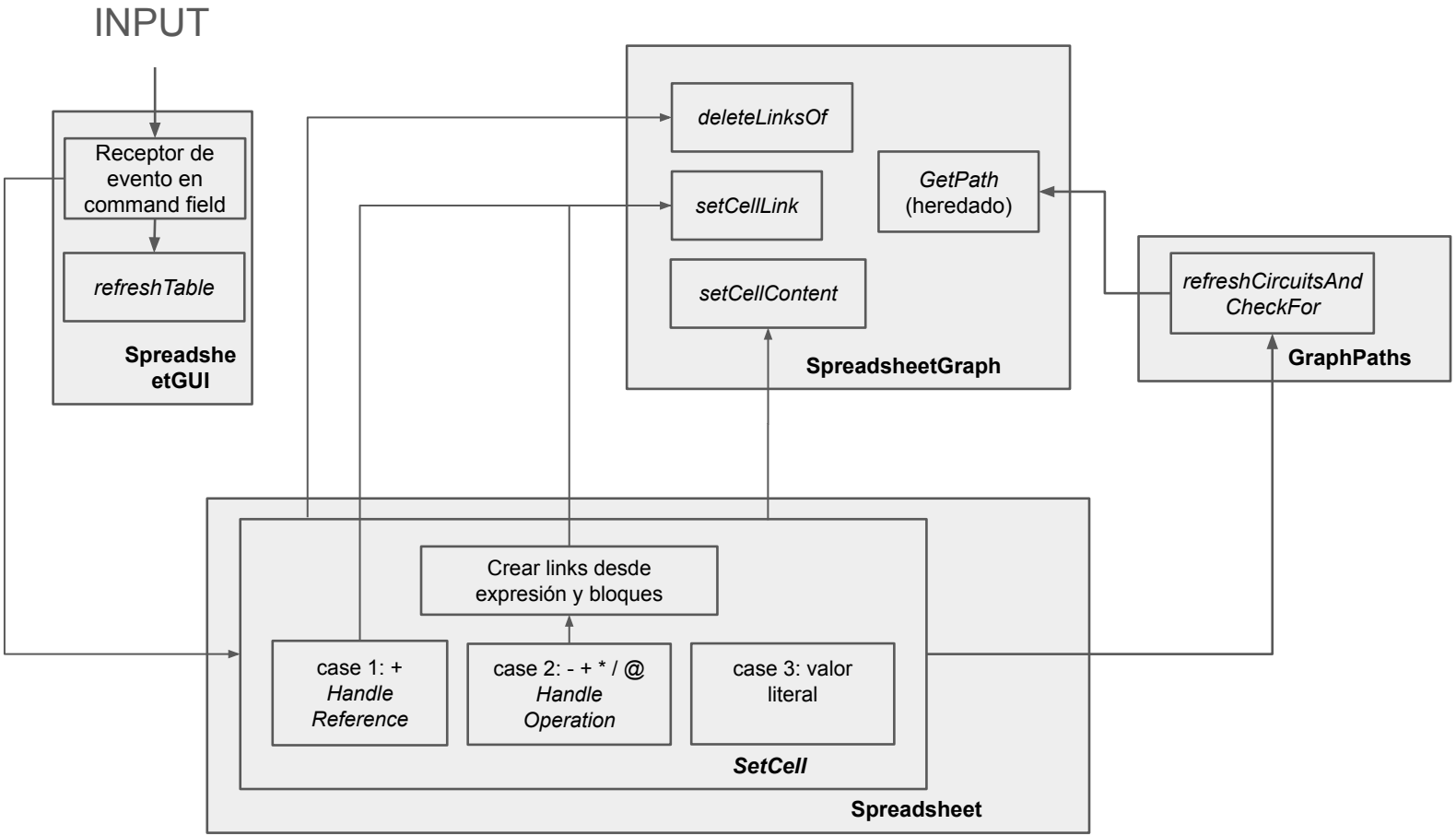


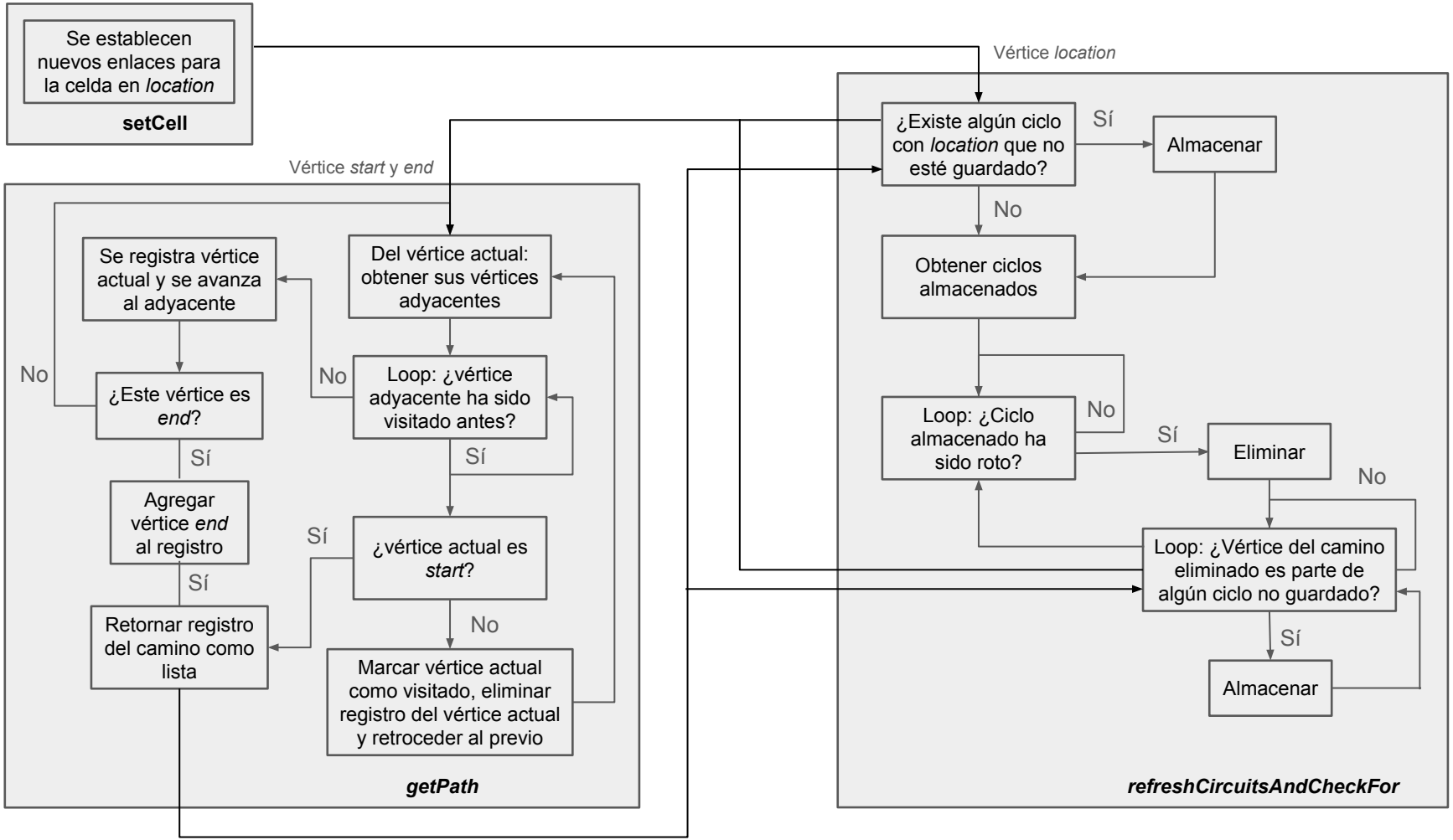

AUTOREF
not AUTOREF

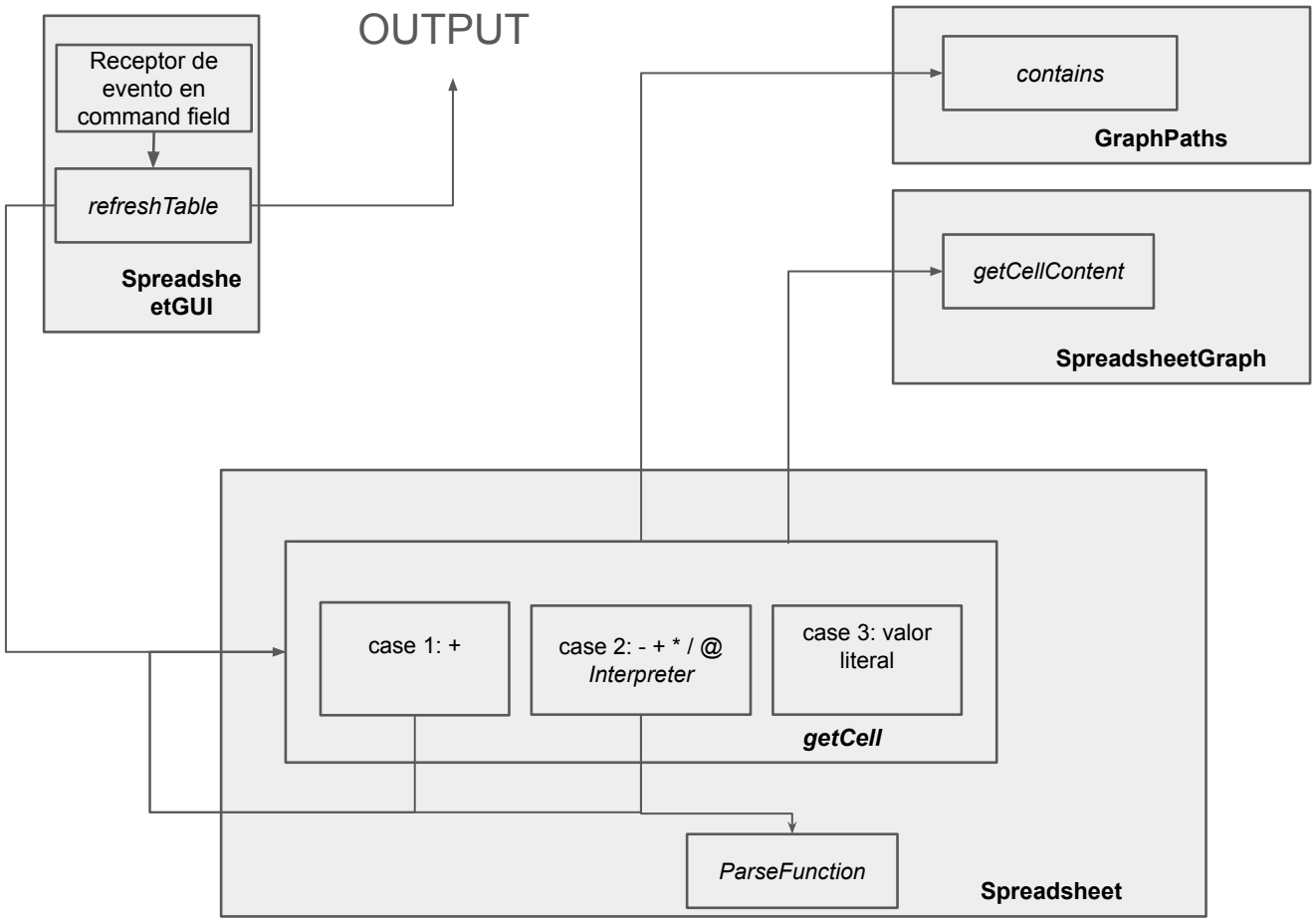

not AUTOREF
AUTOREF

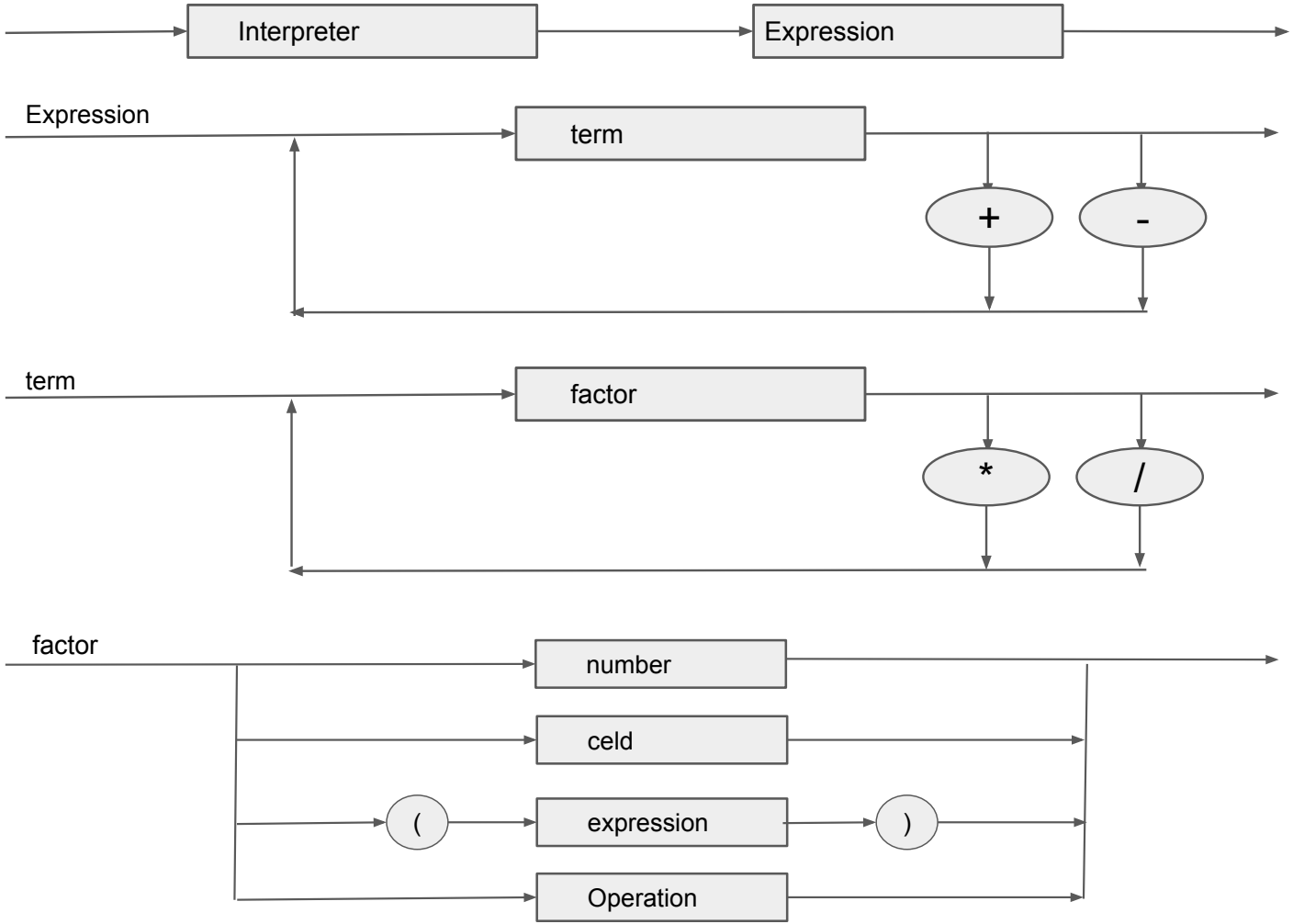

SpreadsheetGraph
+SpreadsheetGraph(List<String> keys)
+String getCellContent(String key)
+void setCellContent(String key, String content)
+void setCellLink(String OriginKey, String DestinyKey
+String getCellLink(String key)
+List<String> getAllCellLinks(String key)
+void deleteLinksOf(String key)
is
HashGraph
-Map<K, Vertex<K, T>> Vertices
+HashGraph(List<K> keys)
+HashGraph(List<K> keys, List<T> values)
+List<Vertex<K, T>> getAdjacentVertices(K key)
+Vertex<K, T> getVertex(K key)
+void setVertex(K key, Vertex<K, T> vertex)
+List<Vertex<K, T>> getAllVertices()
+void replaceVertexValue(K key, T value)
+List<K> getPath(K start, K end)
1..1
1..1
Vertex
-K key
-T value
-List<Vertex<K, T>> adjacents
#Vertex(K key)
#Vertex(K key, T value)
+T getValue()
+K getKey()
+List<Vertex<K, T>> getAdjacents()
+Vertex<K, T> getFirstAdjacent()
+void setValue(T value)
+void setAdjacents(List<Vertex<K, T>> adjacents
+void setAdjacent(Vertex<K, T> to)
+void clear()
+String toString()
1..1
1..1
GraphPaths
-HashMap<K, NODE> Nodes
-HashMap<List<K>, PATH> Paths
-HashGraph<K, T> MainGraph
+GraphPaths(HashGraph<K, T> graph
+boolean contains(K node)
+boolean contains(List<K> path)
+void add(List<K> path)
+void refreshPathsAndFindCircuits()
-boolean notValid(PATH path)
-void deletePath(PATH path)


<<enumeration>>
COLUMN
A
B
C
D
E
F
G
H
Spreadsheet
-final List<String> COLUMNS
-final int rowsNumber
-final int columnsNumber
-SpreadsheetGraph MainGraph
-StreamTokenizer fln
-GraphPaths<String, String> Circuits
+Spreadsheet()
+void setCell(String location, String content)
-void handleReference(String location, String content)
-void handleOperation(String location, String content)
-void refreshCircuitsFor(String location)
-boolean isReference(String s)
-boolean isOperation(String s)
-boolean validarOperacion(String expr)
-void validarExpresion(StreamTokenizer st)
-void validarTermino(StreamTokenizer st)
-void validarFactor(StreamTokenizer st)
-void crearLinksDesdeExpresion(String location, String exp
-void crearLinksDesdeBloques(String location, String expr)
+String getValue(String location)
+String getCell(String location)
+double interpreter(String location)
-double expression(StreamTokenizer st)
-double term(StreamTokenizer st)
-double factor(StreamTokenizer st)
-double parseFunction(StreamTokenizer st)
-List<Double> getRangeValues(String from, String to)
1..1
spreadshee
SpreadsheetGUI
-final java.util.logging.Logger logger
-Spreadsheet spreadsheet
-javax.swing.JTextField commandField
-javax.swing.JTable excel
-javax.swing.JLabel jLabel1
-javax.swing.JLabel jLabel2
-javax.swing.JScrollPane jScrollPane1
-javax.swing.JScrollPane jScrollPane2
-javax.swing.JTextPane viewCommand
+SpreadsheetGUI()
-// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents void initComponen
-void initSpreadsheet()
-void commandFieldActionPerformed(java.awt.event.ActionEvent evt)
-void excelMouseClicked(java.awt.event.MouseEvent evt)
+static void main(String args)
-String toLocation(int row, int columnIndex)
-void refreshTable()
-String[] ProccesText(String text)


INPUT
Receptor de evento en command field
refreshTable
SpreadsheetGUI
deleteLinksOf
setCellLink
setCellContent
GetPath (heredado)
SpreadsheetGraph
refreshCircuitsAndCheckFor
GraphPaths
Crear links desde expresión y bloques
case 1: + Handle Reference
case 2: - + * / @ Handle Operation
case 3: valor literal
SetCell
Spreadsheet


Se establecen nuevos enlaces para la celda en *location*
setCell
Vértice *location*
¿Existe algún ciclo con *location* que no esté guardado?
Sí
Almacenar
No
Obtener ciclos almacenados
Loop: ¿Ciclo almacenado ha sido roto?
No
Sí
Eliminar
No
Loop: ¿Vértice del camino eliminado es parte de algún ciclo no guardado?
Sí
Almacenar
refreshCircuitsAndCheckFor
Vértice *start* y *end*
Se registra vértice actual y se avanza al adyacente
Del vértice actual: obtener sus vértices adyacentes
No
¿Este vértice es *end*?
No
Loop: ¿vértice adyacente ha sido visitado antes?
Sí
Sí
Agregar vértice *end* al registro
Sí
¿vértice actual es *start*?
Sí
Retornar registro del camino como lista
No
Marcar vértice actual como visitado, eliminar registro del vértice actual y retroceder al previo
getPath


OUTPUT
Receptor de evento en command field
refreshTable
SpreadsheetGUI
contains
GraphPaths
getCellContent
SpreadsheetGraph
case 1: +
case 2: - + * / @
Interpreter
case 3: valor literal
getCell
ParseFunction
Spreadsheet


Interpreter
Expression
Expression
term
+
-
term
factor
*
/
factor
number
celd
(
expression
)
Operation

Ejemplos

Literales:
A1 123
A2 54.23
A3 Algoritmos y estructuras de datos

Referencias y operaciones aritméticas:
A4 +A1
A5 +A2
A6 +A3
B1 @sum(A1..A6)
B2 @avg(A1..A6)
B3 (A2 + B2) * @min(A4..A6) / 2
> 354.46
> 88.615
> 3,873.242175

Ciclo doble:
E1 D1 * F1
D1 +D2
D2 +E2
F1 +F2
F2 +E2
E2 +E1
> Ciclo doble pero se reconoce uno.
D2 Uni
> Se rompe ciclo reconocido y se detecta el siguiente