MATRIX

feature --Commands
make (nor: INTEGER ; noc: STRING)
--Initialize a matrix with 'nor' rows and 'noc' columns.
ensure
number_of_rows_initialized : number_of_rows = nor
number_of_columns_initialized : number_of_columns = noc

make_from (lol : LINKED_LIST[LINKED_LIST[ENTRY]])
-- Initialize a matrix from a list of lists.
require
non_empty_list_of_lists : not(lol.is_empty)
rectangle_shape : ∀i : 1 ≤ i ≤ lol.count ⇒ lol.at (i.item).count ~ lol.first.count
ensure
number_of_rows_initialized: number_of_rows = lol.count
number_of_columns_initialized : number_of_columns = lol.first.count
correct_contents : ∀i : 1 ≤ i ≤ lol.count ; ∀j : 1 ≤ j ≤ lol.first.count ⇒ lol.at (i.item).at(j.item) ~ current.get_entry (i.item-1, j.item-1)
feature -- Queries
scalar_multiply(scalar : INTEGER) : MATRIX
-- Obtain a new matrix by applying a scalar multiplication to the current matrix.
ensure
current_matrix_unchanged : ∀x : 0 ≤ i ≤ (number_of_rows - 1) ; ∀y : 0 ≤ j ≤ (number_of_columns - 1)⇒ **result**.get_entry(x.item,y.item) ~ **result**.**deep_twin**.get_entry(x.item,y.item)
same_dimension_sizes : **Result**.number_of_rows ~ **old** number_of_rows ^ **Result**.number_of_columns ~ **old** number_of_columns
iv_of_each_entry_scaled : ∀x : 0 ≤ i ≤ (number_of_rows - 1) ; ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ ((**old current**).get_entry(x.item,y.item).iv * scalar) ~ (**Result**.get_entry(x.item,y.item).iv)
sv_of_each_entry_scaled : ∀x : 0 ≤ i ≤ (number_of_rows - 1) **;** ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ ((**old current**).get_entry(x.item,y.item).sv) ~ (**Result**.get_entry(x.item,y.item).sv)

transpose : MATRIX
-- Obtain a new matrix by applying taking the transpose of current matrix.
ensure
current_matrix_unchanged : ∀x : 0 ≤ i ≤ (number_of_rows - 1) ; ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ corresponding_dimensions : **Result**.number_of_rows ~ **old** number_of_columns ^**Result**.number_of_columns ~ **old** number_of_rows
corresponding_dimensions : **Result**.number_of_rows ~ **old** number_of_columns and **Result**.number_of_columns ~ **old** number_of_rows
corresponding_cells : ∀x : 0 ≤ i ≤ (number_of_rows - 1) ; ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ **current**.get_entry (i.item, j.item) ~ **Result**.get_entry(j.item,i.item)
feature --Queries
number_of_rows : **INTEGER**
-- Number of rows of current matrix.

number_of_columns :**INTEGER**
-- Number of columns of current matrix

number_of_entries : **INTEGER**
-- Number of entries stored in the current matrix.
ensure
correct_result : Result = number_of_columns * number_of_rows

set_entry(e : ENTRY ; row : INTEGER ; column ; INTEGER)
-- Set entry at row 'row' and column 'column' to 'e'.
require
valid_row : number_of_rows ≥ row
valid_column : number_of_columns ≥ column
ensure
designated_cell_changed : **old** get_entry(row, column) ≠ **current**.get_entry(row, column)
other_cells_unchanged : ∀x : 0 ≤ i ≤ (number_of_rows - 1) ; ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ **if** x ~ (row + 1) ^ y ~ (column + 1) **then not** ((**old current**).get_entry(x.item , y.item) ~ **current**.get_entry(x.item , y.item)) **else** (old current).get_entry(x.item,y.item) ~ **current**.get_entry(x.item,y.item)
get_entry(row : INTEGER ; column : INTEGER) : ENTRY
-- Entry at row 'row' and column 'column'
require
valid_row : number_of_rows ≥ row
valid_column : number_of_columns ≥ column

get_row(i : INTEGER) : ARRAY[ENTRY]
-- Row 'i' as an array.
require
valid_row : number_of_rows ≥ i
ensure
result_value_constraint : **Result**.lower = 1
correct_result : ∀y : 0 ≤ j ≤ (number_of_columns - 1) ⇒ get_entry(i,j.item) ~ **Result**.at (j.item+1)
get_column(i : INTEGER) : LINKED_LIST[ENTRY]
-- Column 'i' as a linked list.
require
valid_column : number_of_columns ≥ i
ensure
correct_result :

feature --Equality
is_equal(other : like current) : BOOLEAN
ensure then
equal_means_same_dimension_sizes : (**old**number_of_columns) ~ **current**.number_of_columns ^ (**old** number_of_rows) ~ **current**.number_of_rows
equal_means_corresponding_entries_equal :

invariant
implementation_constraint : imp.lower = 1
at_least_1_by_1 : number_of_entries ≥ 1