

- The **ADT Matrix** is a container that represents a two-dimensional array.
- Each element has a unique position, determined by two indexes: its line and column.
- The domain of the ADT Matrix:  $\mathcal{MAT} = \{mat \mid mat \text{ is a matrix with elements of the type TElem}\}$
- What operations should we have for a Matrix?

Assume: 1-based indexing

- **init(mat, nrL, nrC)**
  - **descr:** creates a new matrix with a given number of lines and columns
  - **pre:**  $nrL \in N^*$  and  $nrC \in N^*$
  - **post:**  $mat \in \mathcal{MAT}$ ,  $mat$  is a matrix with  $nrL$  lines and  $nrC$  columns
  - **throws:** an exception if  $nrL$  or  $nrC$  is negative or zero

- **nrLines(mat)**
  - **descr:** returns the number of lines of the matrix
  - **pre:**  $mat \in \mathcal{MAT}$
  - **post:**  $nrLines \leftarrow$  returns the number of lines from  $mat$

# ADT Matrix - Interface III

- `nrColumns(mat)`
  - **descr:** returns the number of columns of the matrix
  - **pre:**  $mat \in \mathcal{MAT}$
  - **post:**  $nrCols \leftarrow$  returns the number of columns from  $mat$

- `element(mat, i, j)`
  - **descr:** returns the element from a given position from the matrix (assume 1-based indexing)
  - **pre:**  $mat \in \mathcal{MAT}, 1 \leq i \leq nrLines, 1 \leq j \leq nrColumns$
  - **post:**  $element \leftarrow$  the element from line  $i$  and column  $j$
  - **throws:** an exception if the position  $(i, j)$  is not valid (less than 1 or greater than  $nrLines/nrColumns$ )

- **modify**(mat, i, j, val)
  - **descr:** sets the element from a given position to a given value (assume 1-based indexing) and returns the previous value from the position
  - **pre:**  $mat \in \mathcal{MAT}$ ,  $1 \leq i \leq nrLines$ ,  $1 \leq j \leq nrColumns$ ,  $val \in TElem$
  - **post:** the value from position  $(i, j)$  is set to  $val$ .  $modify \leftarrow$  the old value from position  $(i, j)$
  - **throws:** an exception if position  $(i, j)$  is not valid (less than 1 or greater than nrLine/nrColumns)

- Other possible operations:
  - get the (first) position of a given element
  - create an iterator that goes through the elements by columns
  - create an iterator the goes through the elements by lines
  - etc.