Javlonbek Kosimov
EFYBOX
efybox@inf.elte.hu
Group 2

1. assignment/1. task

31st March 2023

Task
Implement the X matrix type which contains integers. These are square matrices that can contain nonzero entries only in their two diagonals. Don't store the zero entries. Store only the entries that can be nonzero in a sequence. Implement as methods: getting the entry located at index (i, j), adding and multiplying two matrices, and printing the matrix (in a square shape).

Double Diagonal X matrix type

Set of values

The Double diagonal matrix (X matrix) contains non-zero values along its two diagonals only.
The size of the matrix is represented by the variable "n" as the matrix is square the rows and columns must be equal.
Indexes are represented are represented by variables "i" and "j" and due to the nature of the double diagonal matrix, non-zero values are only stored on the following index pairs:
All those (i,j) pairs where i=j (first diagonal).
All those (i,j) pairs where i+j=size-1 (second diagonal).

XMatrix(n) = { (a, b) | a ∈ℤ^n, b ∈ℤ^(n-1), (a, b) represent a square matrix M of size n×n such that ∀i,j ∈ [0..n-1] : (i ≠ j) ∧ (i + j ≠ n - 1) → M[i,j] = 0 }

Operations

1. Getting an entry
Function: A
Input: XMatrix a, integer i, integer j
Output: integer e

Formally: $A = XMatrix(n) \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$
$a \quad i \quad j \quad e$
$Pre = (a=a' \wedge i=i' \wedge j=j' \wedge i, j \in [0..n-1])$
$Post = (Pre \wedge e=a[i,j])$

2. Setting an entry
You are given the option of setting an entry from either matrix A or B, the user is prompted to enter index "i" and "j" and a new value "n" to replace the existing value at that index, errors are thrown if the entered indices to not match a diagonals index or if the indices specified do not exist, are invalid or are out of bounds

Function: SetEntry
Input: XMatrix a, integer i, integer j, integer e
Output: XMatrix a

Formally: $A = XMatrix(n) \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$
$a \quad i \quad j \quad e$
$Pre = (e=e' \wedge a=a' \wedge i=i' \wedge j=j' \wedge i,j \in [0..n-1] \wedge (i=j \vee i+j = n-1))$
$Post = (Pre \wedge a[i,j] = e$

3. Sum
Sum of two matrices: $c:=a+b$. The matrices have the same size.

Formally: $A = XMatrix(n) \times XMatrix(n) \times XMatrix(n)$
$a \quad b \quad c$
$Pre = (a=a' \wedge b=b')$
$Post = (Pre \wedge \forall i,j \in [0..n-1] : c[i, j] = a[i, j] + b[i, j]$

4. Multiplication
Multiplication of two X matrices: c := a * b. Both matrix a and matrix b must be of the same size (n x n).

Formally: $A = XMatrix\ (n) \times XMatrix\ (n) \times XMatrix\ (n)$
$a \quad b \quad c$
$Pre = (a=a' \wedge b=b')$
$Post = (Pre \wedge \forall i,j \in [0..n-1] : c[i,j] = \Sigma_{k=0..n-1}\ a[i,k] * b[k,j])$

Representation

XMatrix can be represented using two one-dimensional arrays, one for the main diagonal and the other for the secondary diagonal.

mainDiagonal = < a[0, 0], a[1, 1], ..., a[n – 1, n – 1] >
secondaryDiagonal = < a[0, n – 1], a[1, n – 2], ..., a[n – 1, 0] >

$$a = \begin{matrix} a_{11} & 0 & 0 & \dots & a_{nn} \\ 0 & a_{22} & a_{23} & \dots & 0 \\ 0 & a_{22} & a_{33} & \dots & 0 \\ a_{21} & 0 & 0 & \dots & a_{nn} \end{matrix}$$

Only two sequences are needed, with the help of which any entry of the XMatrix can be retrieved:

$$a[i, j] = \begin{cases} mainDiagonal[i] \text{ if } i = j \\ secondaryDiagonal[i] \text{ if } i + j = n - 1 \\ 0 \text{ if } i \neq j \text{ and } i + j \neq n - 1 \end{cases}$$

Implementation[1]

1. Getting an entry

Getting the entry of the *i*th row and *jth column* ($i,j \in [1..n]$) $e:=a[i,j]$ where the matrix is represented by $v, 1 \leq i \leq i*n$ and *n* stands for the row size and n stands for the column size of the matrix can be implemented as

$i=j \wedge i+j=n-1$	
e:=v[i,j)]	*e*:=*0*

2. Setting an entry

Setting the entry of the *i*th row and *jth* column ($i,j \in [1..n]$) $a[i,j]:=e$ where the matrix isrepresented by $v, 1 \leq i \leq i*n$, and *n* stands for the row size and *n* stands for the column size of the matrix can be implemented as

$i=j \wedge i+j=n+1$	
v[i,j]:=*e*	*SKIP*

3. Sum

The sum of matrices *a* and *b* (represented by arrays *s* and *v*) goes to matrix *c* (represented by array *u*), where all the arrays have to have the same size.

$\forall i \in [0..n-1]$: $u[i]:= s[i] + v[i]$

4. Multiplication

The product of matrices *a* and *b* (represented by arrays *s* and *v*) goes to matrix *c* (represented by array *u*), where all of the arrays have to have the same size.

$\forall i, j \in [0..n-1]$: *c.SetEntry(i, j,* Σ *(a.GetEntry(i, k) * b.GetEntry(k, j)) for k in [0..n-1])*

[1] To implement an operation, a program has to be given (not necessarily structogram).

Testing

Testing the operations (black box testing)

1) Checking constructor

   a) by giving valid size for the matrix (>=1)

   b) by giving invalid size for the matrix (<=0) – throws an exception.

2) Getting and setting an entry

   a) Getting and setting an entry in the diagonals

   b) Getting and setting an entry outside the diagonals

   c) Illegal index, indexing a 0-size matrix

3) Getting a value from the matrix, valid index (in diagonals) is tested as well as an index that is not in the diagonals.

4) Setting values in the matrix, valid index (in diagonals) is successful while invalid index (not in diagonals) throws an error.

5) Addition of matrices – c:=a+b.

   a) matrices of the same size are tested

   b) matrices of different sizes are tested – throws an exception.

6) Multiplication of matrices – c:=a*b.

   a) matrices of the same size is tested

   b) matrices of different sizes is tested – throws an exception