Shafaqat Iqbal **1. assignment/1. task** 16th March 2021
NEPTUN
pmr75l@inf.elte.hu
Group 1

Task

Implement the chessboard matrix type which contains integers. In these matrices, every second entry is zero. The entries that can be nonzero are located like the same colored squares on a chessboard, with indices (1, 1), (1, 3), (1, 5), ..., (2, 2), (2, 4), ... The zero entries are on the indices (1, 2), (1, 4), ..., (2, 1), (2, 3), ... Store only the entries that can be nonzero in row-major order in a sequence. Don't store the zero entries. Implement as methods: getting the entry located at index (i, j), adding and multiplying two matrices and printing the matrix (in a shape of m by n).

Chessboard matrix type

Set of values

$Matrix(n) = \{ a \in \mathbb{Z}^{n\times n} \mid \forall i, j \in [1..n]: (i+j)\% \mathrel{!=} 0 \rightarrow a[i,j]=0 \}$

Operations

1. Getting entry
Getting the entry of the *i*th column and *j*th row ($i \in [1..n]$, $j \in [1..n]$): $e:=a[i,j]$.

Formally:

$$A: \underset{a}{Matrix(n)} \times \underset{i}{\mathbb{Z}} \times \underset{j}{\mathbb{Z}} \times \underset{e}{\mathbb{Z}}$$

$Pre = (a=a' \wedge i=i' \wedge j=j' \; \forall i,j \in [1..n])$
$Post = (Pre \wedge e=a[i,j])$

This operation needs any action only if 2 $(i+j)$ otherwise the output is zero.

2.
Sum
Sum of two matrices: $c:=a+b$. The matrices have the same size.

Formally:

$$A : \underset{a}{Matrix(n)} \times \underset{b}{Matrix(n)} \times \underset{c}{Matrix(n)}$$

$Pre = (a=a' \wedge b=b')$
$Post = (Pre \wedge \forall i, j \in [1..n]: (i+j)\% = 0 \rightarrow c[i, j]= a[i,j] + b[I,j]$ and
$\wedge \forall i, j \in [1..n]: (i+j)\% \mathrel{!=} 0 \rightarrow c[i, j]=0$

3. Multiplication
Multiplication of two matrices: $c:=a*b$. It is possible only if number of columns of first matrix is same as number of rows as second matrix.

Formally:

$$A : \underset{a}{Matrix(n)} \times \underset{b}{Matrix(n)} \times \underset{c}{Matrix(n)}$$

Pre = ($a=a' \wedge b=b'$)
Post = (Pre $\wedge \forall i, j \in [1..n]$: c[i,i]= Σk=1..n a[i,k] * b[k,j])

In case of chessboard matrices all entries at $2 \nmid (i+j)$ will be 0.

Representation

Only the entries at even indices of the *n* matrix have to be stored. The matrices can only be square matrices of even size.
Considering a $n \times n$ matrix:

$$a = \begin{matrix} a_{11} & 0 & a_{12} \ldots & 0 \\ 0 & a_{22} & 0 & \ldots a_{2n} \\ a_{31} & 0 & a_{33} \ldots & 0 \\ 0 & a_{42} & 0 & \ldots a_{4n} \end{matrix}$$

$\leftrightarrow$ $vec = \langle a_{11} \; a_{12} \; a_{22} \; a_{2n} \; a_{31} \; a_{33} \; a_{42} \; a_{4n} \rangle$

Only a one-dimension array (*vec*) is needed, with the help of which nonzero entries of the chessboard matrix can be stored:

$$a[i,j] = \{ \\ \quad vec[i] \quad if \;\; 2 \mid (i+j) \\ \quad 0 \quad if \;\; 2 \nmid (i+j) \\ \}$$

Implementation

1. Getting an entry

Getting the entry of the *i*th column and *jth* row ($i,j \in [1..n]$:) $e:=a[i,j]$ where the matrix is represented by vec: $1 \le (n/2 * i + (j/2)) \le n$ in case if size of matrix A(*n*) can be implemented as:

if (i+j) % 2 == 0:
A(*i*,*j*): vec[n/2 * i + (j/2)]
if (i+j) % 2 == 1:
A(*i*,*j*): 0

3. Sum

The sum of matrices *a* and *b* (represented by vectors A and *B*) goes to matrix *c* (represented by vector C), where all of the vectors have to have the same size.

$\forall\ i\ ,j \in [1..n]$: C[i] := A[i] + B[i]

4. Multiplication

The product of matrices *a* and (represented by vectors A and *B*) goes to matrix *c* (represented by vector C), which is possible if sizes of the matrices are same.

$\forall\ i\ ,j,k \in [1..n]$: C [i,j] := C[i,j] + A[i,k] * B[k,j]

CODE:

You can find the code by opening matrix.cpp, under assignment/src.

Testing

Testing the operations (black box testing)

Test Case 1:
File Constructor.
Must throw an error while having the wrong input file.
Test Case 2:
Index Out of Bound.
If the given indexes to access the element are out of bound then it must give the error.
Test Case 3:
Vector Constructor.
Will throw an error when data in vector is wrong
Test Case 4:
Negative Indexes.
If the given indexes to access the element are negative then it must throw an error.
Test Case 5:
Correct Indexes.
If the indexes are correct (positive and less than size of matrix) then it must give the expected result.
Test Case 6:
Checking Value at given index.
It must give the exact same value on a given index which is put in vector already.
Test Case 7:
Adding two valid matrices.
It must give the expected result on this operation.

Test Case 8:
Multiplying two valid matrices.
It must give the expected result on this operation.