LINEAR ALGEBRA

Assignment
[Due Date: **04 September 2023** Midnight]

Implement a **C++ class** named **Matrix** to represent a matrix. The file **matrix.h** shall contain the implementation. Following implementation should be attempted.

Private members

nb_row and **nb_col** store the number of rows and the number of columns of the matrix.

mat_data is a **std::vector<double>**, it stores the matrix data, the ij values.

Constructors

The class has 5 constructors. A constructor to generate random matrices (uniformly or normaly), a constructor to load data from a CSV file and create a matrix, a constructor to create a matrix using an initializer list or a vector of vectors.

Overloaded operators

Multiplication *, *=

Addition +, +=

Subtraction -, -=

Division \

Equal ==

Difference !=

Member functions

column and **row** to get a column or a row of a matrix.

sub_matrix returns a submatrix of a matrix.

shape print the dimension of a matrix.

reshape reshape (change the number of rows and columns) of a matrix.

add_row and **add_column** add a new column or a new row to a matrix.

remove_column delete a column of a matrix.
reorder_column sort a matrix column, or flat matrix.

sort_matrix sort a matrix by column, using indexes

T() and **transpose (a friend function):** return the transpose of the matrix.

Id create a unitary matrix**.**

sum returns the sum a flattened matrix.

avg compute the average of a flattened matrix.

head print first row of matrix

print formatted print of a matrix, a value, a string.

to_csv save a matrix in a csv file.

Non Member Functions

diag_matrix returns a diagonal matrix.
is_triangular Check if a matrix is triangular
LU decompostion of a matrix using the basic LU algorithm
QR_factorization Matrix factorization using QR-factorization
eigen_decomposition eigen decompostion of a matrix using the basic QR algorithm
SVD Compute the singular value decomposition of a matrix using the QR algorithm