Unit Tests

UT1: {FR1} The matrix class shall have a default constructor of 1 row and 1column of bytes.

Input: No variables.

Expected Result: matrix of bytes with 1 row and 1 column.

UT2: {FR2} The matrix class shall have a custom constructor of n rows and m columns of element_type.

Input: Number_of_rows, Number_of_columns, Element_type

Expected Result: matrix of element_type with **n** rows and **m** columns.

UT3: {FR3} The matrix element_type shall be any one of the following 4 types declared by the user at instantiation. BYTE_TYPE, FLOAT_TYPE, SINT_TYPE, and UINT_TYPE.

Input: Number_of_rows, Number_of_columns, Element_type = BYTE_TYPE.

Expected Result: matrix of bytes with **n** rows and **m** columns.

Input: Number_of_rows, Number_of_columns, Element_type = FLOAT_TYPE.

Expected Result: matrix of floats with **n** rows and **m** columns.

Input: Number_of_rows, Number_of_columns, Element_type = SINT_TYPE.

Expected Result: matrix of signed integers with **n** rows and **m** columns.

Input: Number_of_rows, Number_of_columns, Element_type = UINT_TYPE.

Expected Result: matrix of unsigned integers with **n** rows and **m** columns.

UT4: {FR4} The matrix elements shall be accessed via unit-based addressing. First element is accessed by (1,1). Last element accessed by (n,m).

Input:

Expected Result:

UT5: {FR5} The overloaded operator ***get*** shall obtain data from a specific element of the matrix. Element_contents = get_from(Row, Column);

Input:

Expected Result:

UT6: {FR6} The overloaded operator ***set*** shall load data into a specific element of the matrix. set(Row, Column, With_value);

Input:

Expected Result:

UT7: {FR7} The class shall support the operation of addition. Given two matrices **A** and **B** create a third matrix **C** with the elements of **A** and **B** added together.

$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} e & f \\ g & h \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$

Input:

Expected Result:

UT8: {FR8} The class shall support the operation of scalar multiplication. Given the matrix **A** and a scalar value **K** create a second matrix **B** with the elements of **A** multiplied by the scalar **K**.

$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \textbf{scalar} = k \quad \mathbf{B} = \begin{bmatrix} ka & kb \\ kc & kd \end{bmatrix}$

Input:

Expected Result:

UT9: {FR9} The class shall support the operation of matrix multiplication. Given two matrices **A** and **B** create a third matrix **C** with the elements of **A** row multiplied by the elements of **B** column and then added together.

$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} e & f \\ g & h \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{bmatrix}$

Input:

Expected Result:

UT10: {FR10} The class shall support the operation of transpose. Given a matrix **A** create a second matrix $\mathbf{A}^T$ whose elements are the transpose of **A.**

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & *** & a_{1m} \\ a_{21} & a_{22} & *** & a_{2m} \\ * & * & *** & * \\ * & * & *** & * \\ a_{n1} & a_{n2} & *** & a_{nm} \end{pmatrix} \qquad \mathbf{A}^T = \begin{pmatrix} a_{11} & a_{21} & *** & a_{n1} \\ a_{12} & a_{22} & *** & a_{n2} \\ * & * & *** & * \\ * & * & *** & * \\ a_{1m} & a_{2m} & *** & a_{nm} \end{pmatrix}$$

Input:

Expected Result: