

Matrix Calculator

Team Matrix Calculator

Team

Mason Park



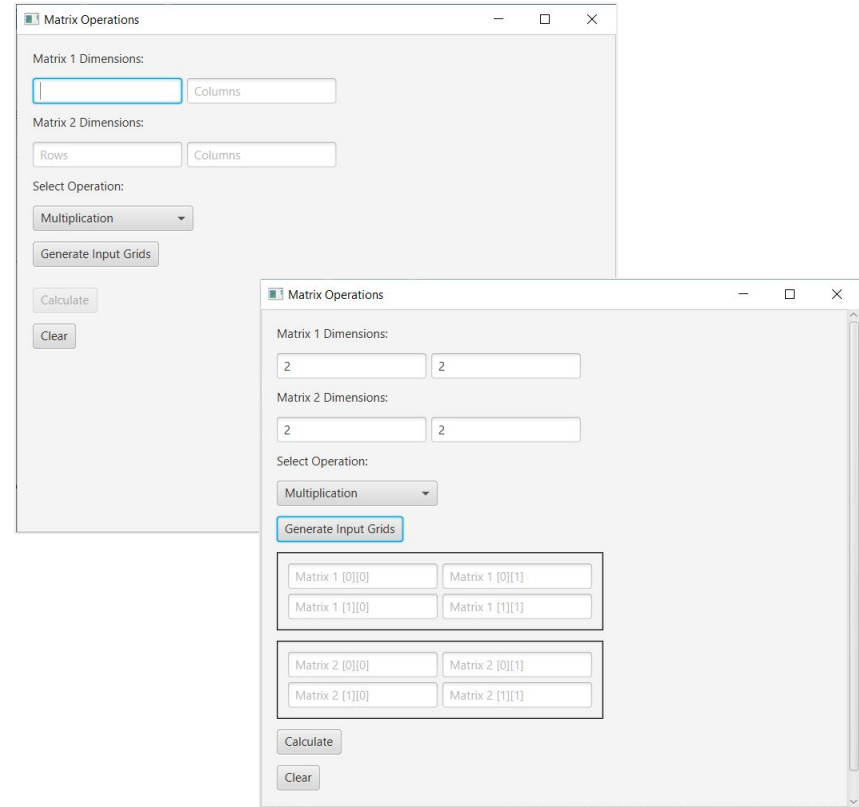
Garrett Vukovich

Problem

- Many problems in linear algebra and other related mathematical fields require performing various operations on matrices. These operations tend to be tedious to perform by hand, especially on large matrices. Most calculators don't have support for matrix operations and online calculators require an internet connection. Our goal is to make a calculator that is able to calculate matrix operations.

Solution

- We used javaFx to create a GUI that allows users to select an operation and input the dimensions of each operand matrix.
- After the operation and dimensions are selected, clicking the “Generate Input Grids” button, will generate the fields that allow the user to enter the elements for each matrix.



Solution

- After the user inputs the values for each matrix, clicking the calculate button will calculate the result of the operation and display it to the user.

Matrix Operations

2 2

Select Operation:
Multiplication

Generate Input Grids

1	2
3	4

1	0
0	1

1 2
3 4

Calculate

Clear

Matrix Operations

2 2

Select Operation:
Multiplication

Generate Input Grids

1	2
3	4

5	6
7	8

1 2
3 4

19 22
43 50

Calculate

Solution

- Some operations don't use two matrices as operands, such as the determinant operation which only uses one matrix or the exponent operation which requires a matrix and a scalar. These operations use the dimensions of matrix 1 for the matrix operand and ignore the dimensions of matrix 2.

The screenshot shows the 'Matrix Operations' application window. The 'Matrix 1 Dimensions' are set to 3 rows and 3 columns. The 'Matrix 2 Dimensions' are set to 'Rows' and 'Columns'. The 'Select Operation' dropdown is set to 'Determinant'. The 'Generate Input Grids' button is visible. Below the input grids, a large green-bordered box displays the result '0'. At the bottom, there are 'Calculate' and 'Clear' buttons.

1	2	3
4	5	6
7	8	9

0

The screenshot shows the 'Matrix Operations' application window. The 'Matrix 1 Dimensions' are set to 2 rows and 2 columns. The 'Matrix 2 Dimensions' are set to 'Rows' and 'Columns'. The 'Select Operation' dropdown is set to 'Exponent'. The 'Generate Input Grids' button is visible. Below the input grids, a large green-bordered box displays the result '37 54' and '81 118'. At the bottom, there is a 'Calculate' button.

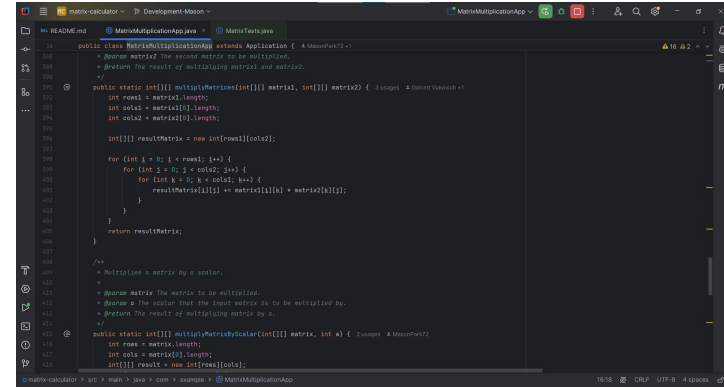
1	2
3	4

3

37 54
81 118

Solution

- If the user would like to perform another calculation, they can either change the values and click calculate again, click the “Generate Input Grids” button to reset all of the input grids, or click the clear button to clear all fields and reset to the beginning.
- All of the mathematical operations are handled in Java using 2D integer arrays. Unit tests are used to ensure that the computations are returning accurate results.
- All inputs are validated before computations. Meaningful error messages are displayed when input is invalid.



```
public class MatrixMultiplicationApp extends Application {
    // Generate matrix2 the second matrix to be multiplied.
    // Generate the result of multiplying matrix and matrix2.

    public static int[][] multiplyMatrixes(int[][] matrix1, int[][] matrix2) {
        // 3 inputs: A Matrix, A Matrix, A Matrix
        int rows = matrix1.length;
        int cols1 = matrix1[0].length;
        int cols2 = matrix2[0].length;

        int[][] resultMatrix = new int[rows][cols2];

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols2; j++) {
                for (int k = 0; k < cols1; k++) {
                    resultMatrix[i][j] += matrix1[i][k] * matrix2[k][j];
                }
            }
        }

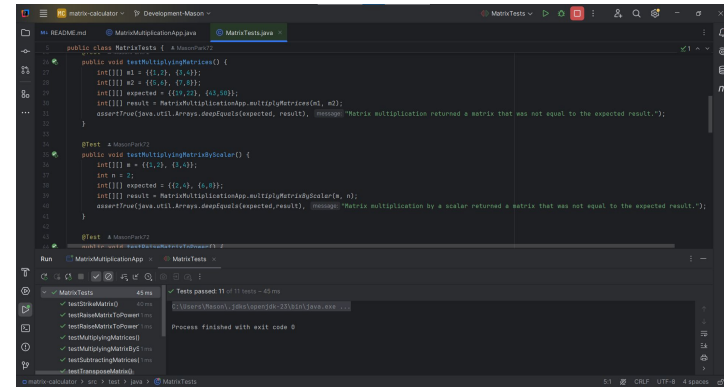
        return resultMatrix;
    }

    // Multiply a matrix by a scalar.
    // Generate matrix the matrix to be multiplied.
    // Generate the scalar that the input matrix is to be multiplied by.
    // Generate the result of multiplying matrix by a.

    public static int[][] multiplyMatrixByScalar(int[][] matrix, int a) {
        // 2 inputs: A Matrix, A Number
        int rows = matrix.length;
        int cols = matrix[0].length;
        int[][] result = new int[rows][cols];

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                result[i][j] = matrix[i][j] * a;
            }
        }

        return result;
    }
}
```



```
public class MatrixTests {
    // Test multiplyMatrixes
    public void testMultiplyMatrixes() {
        int[][] m1 = {{1,2}, {3,4}};
        int[][] m2 = {{5,6}, {7,8}};
        int[][] expected = {{19,28}, {43,50}};
        int[][] result = MatrixMultiplicationApp.multiplyMatrixes(m1, m2);
        assertEquals("Matrix multiplication returned a matrix that was not equal to the expected result.", expected, result);
    }

    // Test multiplyMatrixByScalar
    public void testMultiplyMatrixByScalar() {
        int[][] m = {{1,2}, {3,4}};
        int n = 2;
        int[][] expected = {{2,4}, {6,8}};
        int[][] result = MatrixMultiplicationApp.multiplyMatrixByScalar(m, n);
        assertEquals("Matrix multiplication by a scalar returned a matrix that was not equal to the expected result.", expected, result);
    }
}
```

Demo

Lessons Learned

- We learned the importance of developing with modularity in mind from the start, as it allows for greater testability, understandability, and ease of expansion.
- We learned the importance of identifying our scope and features early on in the process, as it would have saved time by not having to rewrite code to add new features.

Issues/Bugs Known

- Changing the operation after the input grids are generated can lead to undefined behavior.
- Entering a negative value when raising a matrix to a power results in the matrix being raised to the first power.

Future Versions

- Allow users to enter doubles as values.
- Add a convert to Reduced Row Echelon Form operation.
- Add a matrix inverse operation.
- Add a diagonalization operation.
- Add an operation to check if a matrix is orthogonal.
- Add a calculate Eigenvalues operation.
- Add a calculate Eigenvectors operation.
- Add a button to move a calculated matrix to one of the input slots.
- Add the ability to copy/paste entire matrices.

Thank you!

Any questions?