ADJOINT OF MATRIX CALCULATOR

@by jatingangare

The adjoint of a matrix, also known as the adjugate or the classical adjoint, is a fundamental concept in linear algebra. It is used to find the inverse of a square matrix when it exists. The adjoint of a matrix is essentially a way to represent the cofactors of the elements of the matrix. To compute the adjoint of a matrix, you'll need to follow these steps:

1. Cofactor Calculation: Compute the cofactor of each element in the matrix. The cofactor of an element is found by taking the determinant of the matrix formed by removing the row and column containing that element, multiplying it by (-1) raised to the power of the sum of the row and column indices, and finally, multiplying it by the minor.

2. Transpose: After calculating the cofactors for all elements, transpose the resulting matrix. This means that you swap rows and columns.

Here's a step-by-step explanation with an example:

Example Matrix (A):

A = | 2 3 1 |
| 0 1 4 |
| 5 2 3 |

Step 1 - Cofactor Calculation:

To calculate the cofactor for each element, we need to find the determinant of the matrix formed by removing the row and column containing that element. We'll also need to determine whether to use a positive or negative sign for each cofactor based on the position of the element in the matrix.

For element A[1][1] (2):
Cofactor = (-1)^(1+1) * Det(|1 4 |) = 1 * (1*3 - 4*2) = -5

For element A[1][2] (3):
Cofactor = (-1)^(1+2) * Det(|0 4 |) = -1 * (0*3 - 4*5) = 20

For element A[1][3] (1):
Cofactor = (-1)^(1+3) * Det(|0 1 |) = 1 * (0*3 - 1*5) = -5

For element A[2][1] (0):
Cofactor = (-1)^(2+1) * Det(|3 1 |) = -1 * (3*3 - 1*5) = -8

For element A[2][2] (1):
Cofactor = (-1)^(2+2) * Det(|2 1 |) = 1 * (2*3 - 1*5) = 1

For element A[2][3] (4):
Cofactor = (-1)^(2+3) * Det(|2 3 |) = -1 * (2*1 - 3*0) = -2

For element A[3][1] (5):
Cofactor = (-1)^(3+1) * Det(|3 1 |) = 1 * (3*2 - 1*4) = 2

For element A[3][2] (2):
Cofactor = (-1)^(3+2) * Det(|2 1 |) = -1 * (2*2 - 1*5) = -1

For element A[3][3] (3):
Cofactor = (-1)^(3+3) * Det(|2 3 |) = 1 * (2*1 - 3*0) = 2

Step 2 - Transpose:
Now, we transpose the matrix formed by the cofactors:

Adj(A) = | -5 -8   2 |
         | 20   1 -1 |
         | -5 -2   2 |

The resulting matrix is the adjoint of matrix A (Adj(A)).
The adjoint is useful in finding the inverse of a square matrix. If the determinant of the matrix is non-zero, you can use the following formula to find the inverse:

A^(-1) = (1 / det(A)) * Adj(A)

Where `det(A)` is the determinant of matrix A. If `det(A)` is zero, the matrix is singular, and its inverse does not exist.

ABOUT CODE :

The provided HTML and JavaScript code creates a web application called "Matrix Adjoint Finder," which allows users to input a square matrix and calculates its adjoint. Here's how the code works and the key concepts used:

HTML Structure:
- The HTML code defines the structure of the web page, including form inputs, buttons, and result display areas.
- Users can input the number of rows and columns for the square matrix, and they can also input the matrix values or generate an empty matrix.
- The results, including the adjoint matrix, are displayed on the web page.

CSS Styling:
- CSS styles are used to format and style the HTML elements, making the web page visually appealing.
- Styles are applied to the header, form elements, buttons, result display, and footer to create a professional appearance.

JavaScript Functionality:
- JavaScript code adds interactivity to the web page and performs the matrix-related calculations.

Key JavaScript Concepts and Logic:

1. Event Listeners: The code uses event listeners to capture user interactions. For example:
   - The "Generate Matrix" button's click event listener generates an empty matrix input area based on user-defined row and column values.

- The "Calculate Adjoint" button's click event listener calculates the adjoint matrix when the user provides input.
- The "Clear" button's click event listener clears the input and result area.
- The "Download Input and Result" button's click event listener downloads the user's input and the calculated result as a text file.

2. Matrix Input Parsing: The `parseMatrixInput` function parses the user's input into a 2D array (matrix) by splitting rows and columns and ensuring valid numeric values.

3. Square Matrix Validation: The `isSquareMatrix` function checks if the provided matrix is square (having the same number of rows and columns).

4. Matrix Adjoint Calculation: The `calculateAdjoint` function calculates the adjoint of the input square matrix. It uses the concept of cofactors and determinants to calculate individual elements of the adjoint matrix.

5. Matrix to String Conversion: The `matrixToString` function converts a 2D matrix into a formatted string for display.

6. Blob Creation for Download: When the user clicks the "Download Input and Result" button, the code creates a Blob containing the user's input and result in a plain text format. It then generates a downloadable link for the Blob, allowing the user to download the data as a text file.

Overall, this code combines HTML for structure and presentation, CSS for styling, and JavaScript for interactivity and matrix-related calculations to create a web application for finding the adjoint of a square matrix. It employs event-driven programming to respond to user actions and mathematical logic for matrix operations.

Thank you for visiting !

@by jatingangare