NAME:- KETAN DILIP ATTARDE

REG NO:- 24-27-06

MTECH DATA SCIENCE

Algorithm: Converting a Matrix to Echelon Form

1. **Input**: A matrix `A` of dimensions m x n.
2. Initialize `current_row` as 0.
3. **For each column `col` in range(n):**
   - If `current_row` >= m, exit loop.

     a. **Find the Pivot Row:**

     - Initialize `pivot_row` as `current_row`.
     - While `pivot_row` < m and `A[pivot_row, col]` == 0, increment `pivot_row`.

     b. If `pivot_row` == m, continue to the next column.

     c. **Swap Rows:**

     - Swap the `current_row` and `pivot_row` rows of matrix `A`.

     d. **Scale Pivot Row:**

     - Divide the `current_row` row by `A[current_row, col]` to make the pivot element 1.

     e. **Eliminate Values Below Pivot Element:**

     - For each row `row` from `current_row + 1` to `m - 1`:
       - Compute `factor = A[row, col]`.

- Subtract `factor * A[current_row]` from row `A[row]`.

f. Increment `current_row`.

4. **Output**: The matrix `A` in echelon form.

```
import numpy as np


def echelon_form(matrix):
    """
    Convert the input matrix to echelon form using Gaussian elimination.

    Parameters:
    matrix (numpy.ndarray): The input matrix to be converted to echelon form.

    Returns:
    numpy.ndarray: The matrix in echelon form.
    """
    m, n = matrix.shape
    current_row = 0

    for col in range(n):
        if current_row >= m:
            break

        # Find the first non-zero pivot element in the column
        pivot_row = current_row
        while pivot_row < m and matrix[pivot_row, col] == 0:
            pivot_row += 1

        if pivot_row == m:
            continue

        # Swap the current row with the pivot row
        matrix[[current_row, pivot_row]] = matrix[[pivot_row, current_row]]
```

```
            # Scale the pivot row so that the pivot element becomes 1
            pivot_element = matrix[current_row, col]
            matrix[current_row] /= pivot_element

            # Eliminate values below the pivot element
            for row in range(current_row + 1, m):
                factor = matrix[row, col]
                matrix[row] -= factor * matrix[current_row]

            current_row += 1

        return matrix


    # Example matrix
    input_matrix = np.array([[2, 1, -1, 8],
                             [-3, -1, 2, -11],
                             [-2, 1, 2, -3]], dtype=float)  # Specify dtype as float

    print("The input Matrix is:- ")
    print(input_matrix)
    print("-"*100)
    result = echelon_form(input_matrix)
    print("Echelon Form:")
    print(result)
```

```
The input Matrix is:- 
[[  2.   1.  -1.   8.]
 [ -3.  -1.   2. -11.]
 [ -2.   1.   2.  -3.]]
----------------------------------------------------------------------------------------------------
Echelon Form:
[[ 1.   0.5 -0.5  4. ]
 [ 0.   1.   1.   2. ]
 [-0.  -0.   1.  -1. ]]
```