# Documentation for the 4x4 Matrix Manipulation Program

This document provides an overview and explanation of the assembly language code for a program that performs various operations on a 4x4 matrix of bytes. The program allows users to interactively enter the matrix values, and then it provides options to rotate and shift the matrix, displaying the results on the screen.

# Purpose:

The purpose of this program is to demonstrate basic matrix manipulation operations such as rotation and shifting using a 4x4 matrix of bytes. Users can input the matrix values, and the program will apply specific operations and display the resulting matrix.

# 2. Program Description:

The program is written in assembly language and follows a multi-segment executable file template. It uses the data, stack, and code segments to manage the memory and logic flow.

# 3. Data Segment:

The data segment includes the following data elements:

**text:** A buffer to store a 4x4 matrix of bytes (16 bytes).

**counter:** A word (2 bytes) to store the number of rows (or columns) in the matrix.

**RM:** An array of bytes containing some pre-defined values used for rotating the matrix.

**txt:** A string to display the message "press any key to continue.." when required.

**cc:** A byte (1 byte) to store a character (usually used for reading a single keypress).

## 4. Procedures:

The program defines several procedures to perform different operations on the matrix and to print the matrix contents:

**press\_key:** Displays the message "press any key to continue.." and waits for a keypress.

**rotateB:** Rotates the matrix counter-clockwise using the values in the RM array.

**shRowB:** Shifts each row of the matrix to the right (circular shift) by 1 position.

**shColB:** Shifts each column of the matrix down (circular shift) by 1 position.

**rotate:** Rotates the matrix clockwise using the values in the RM array.

**shCol:** Shifts each column of the matrix up by 1 position.

**shRow:** Shifts each row of the matrix to the left by 1 position.

**new\_line:** Prints a newline character (line feed and carriage return).

**read:** Reads a 4x4 matrix of bytes from the user and calculates the number of rows (or columns).

**print\_st:** Prints the matrix (in its current state) as a 4x4 grid of bytes.

**print:** Prints the matrix as a formatted 4x4 grid with spaces between elements.

## 5. Code Execution Flow:

The start label is the entry point of the program. Upon execution, the program will:

Initialize the data segment registers (DS and ES).

Prompt the user to input a 4x4 matrix of bytes.

Display the entered matrix in a grid format.

Prompt the user to press any key to continue.

Perform various matrix operations one after another, displaying the intermediate results on the screen.

The available matrix operations are:

Rotate the matrix clockwise.

Rotate the matrix counter-clockwise.

Shift the matrix rows to the left.

Shift the matrix columns up.

# 6. Limitations:

The program assumes that the user will input a 4x4 matrix with valid byte values.

It may not handle invalid inputs gracefully, and unexpected behavior can occur if the user inputs non-byte values or other erroneous data.

The specific operations performed on the matrix are predefined in the RM array, and they may not cover all possible matrix manipulations.

The program might not be optimized for large matrices or efficient memory usage.

# 7. Conclusion:

This assembly language program provides a basic demonstration of matrix manipulation using a 4x4 matrix of bytes. It allows users to interactively input the matrix, apply various operations, and visualize the transformations.