DATA STRUCTURES

Division	**CS(AIML) -A**
Batch	**2**
GR-no	**12311493**
Roll no	**54**
Name	**Atharva Kangralkar**

Assignment 1:

Write a C/C++ Program to implement an array program for following: Input: m* n matrix

a. Find saddle point in the Matrix.

B. Magic square Matrix. (Check)

c. Represent given matrix in its Sparse form.

Code:-

```
#include <stdio.h>

#define MAX 100

void inputMatrix(int matrix[MAX][MAX], int m, int n) {
    printf("Enter the elements of the matrix row by row:\n");
    for (int i = 0; i < m; i++) {
        printf("Row %d: ", i + 1);
        for (int j = 0; j < n; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
}

void displayMatrix(int matrix[MAX][MAX], int m, int n) {
    printf("Matrix:\n");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}

void findSaddlePoint(int matrix[MAX][MAX], int m, int n) {
```

```
    int found = 0;
    for (int i = 0; i < m; i++) {
        int minRow = matrix[i][0];
        int colIndex = 0;
        for (int j = 1; j < n; j++) {
            if (matrix[i][j] < minRow) {
                minRow = matrix[i][j];
                colIndex = j;
            }
        }

        int isSaddle = 1;
        for (int k = 0; k < m; k++) {
            if (matrix[k][colIndex] > minRow) {
                isSaddle = 0;
                break;
            }
        }

        if (isSaddle) {
            printf("Saddle point found at (%d, %d) with value %d\n", i, colIndex, minRow);
            found = 1;
        }
    }

    if (!found) {
        printf("No saddle point found.\n");
    }
}
```

```
int isMagicSquare(int matrix[MAX][MAX], int n) {
    int sumDiag1 = 0, sumDiag2 = 0;
    for (int i = 0; i < n; i++) {
        sumDiag1 += matrix[i][i];
        sumDiag2 += matrix[i][n - i - 1];
    }

    if (sumDiag1 != sumDiag2)
        return 0;

    for (int i = 0; i < n; i++) {
        int rowSum = 0, colSum = 0;
        for (int j = 0; j < n; j++) {
            rowSum += matrix[i][j];
            colSum += matrix[j][i];
        }

        if (rowSum != sumDiag1 || colSum != sumDiag1)
            return 0;
    }

    return 1;
}

void sparseMatrix(int matrix[MAX][MAX], int m, int n) {
    int sparse[MAX][3];
    int k = 0;
```

```
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (matrix[i][j] != 0) {
                sparse[k][0] = i;
                sparse[k][1] = j;
                sparse[k][2] = matrix[i][j];
                k++;
            }
        }
    }

    printf("Sparse Matrix Representation:\n");
    printf("Row Column Value\n");
    for (int i = 0; i < k; i++) {
        printf("%d %d %d\n", sparse[i][0], sparse[i][1], sparse[i][2]);
    }
}

int main() {
    int matrix[MAX][MAX];
    int m, n, choice;

    printf("Enter the number of rows and columns of the matrix: ");
    scanf("%d %d", &m, &n);

    inputMatrix(matrix, m, n);

    do {
        printf("\nMenu:\n");
```

```
printf("1. Find Saddle Point\n");
printf("2. Check Magic Square\n");
printf("3. Sparse Matrix Representation\n");
printf("4. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        findSaddlePoint(matrix, m, n);
        break;
    case 2:
        if (m != n) {
            printf("Matrix is not square. Please enter a square matrix:\n");
            printf("Enter the size of the square matrix: ");
            scanf("%d", &m);
            n = m;
            inputMatrix(matrix, m, n);
        }
        if (isMagicSquare(matrix, n)) {
            printf("The matrix is a magic square.\n");
        } else {
            printf("The matrix is not a magic square.\n");
        }
        break;
    case 3:
        sparseMatrix(matrix, m, n);
        break;
    case 4:
```

```
            printf("Exiting program.\n");
            break;
        default:
            printf("Invalid choice. Please try again.\n");
    }
} while (choice != 4);

return 0;
}
```

Code Screenshot:-

```c

#include <stdio.h>

#define MAX 100

void inputMatrix(int matrix[MAX][MAX], int m, int n) {
    printf("Enter the elements of the matrix row by row:\n");
    for (int i = 0; i < m; i++) {
        printf("Row %d: ", i + 1);
        for (int j = 0; j < n; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
}

void displayMatrix(int matrix[MAX][MAX], int m, int n) {
    printf("Matrix:\n");
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}

void findSaddlePoint(int matrix[MAX][MAX], int m, int n) {
    int found = 0;
```

```
    for (int i = 0; i < m; i++) {
        int minRow = matrix[i][0];
        int colIndex = 0;
        for (int j = 1; j < n; j++) {
            if (matrix[i][j] < minRow) {
                minRow = matrix[i][j];
                colIndex = j;
            }
        }

        int isSaddle = 1;
        for (int k = 0; k < m; k++) {
            if (matrix[k][colIndex] > minRow) {
                isSaddle = 0;
                break;
            }
        }

        if (isSaddle) {
            printf("Saddle point found at (%d, %d) with value %d\n"
                , i, colIndex, minRow);
            found = 1;
        }
    }

    if (!found) {
        printf("No saddle point found.\n");
    }
}

int isMagicSquare(int matrix[MAX][MAX], int n) {
    int sumDiag1 = 0, sumDiag2 = 0;
    for (int i = 0; i < n; i++) {
        sumDiag1 += matrix[i][i];
        sumDiag2 += matrix[i][n - i - 1];
    }

    if (sumDiag1 != sumDiag2)
        return 0;

    for (int i = 0; i < n; i++) {
        int rowSum = 0, colSum = 0;
        for (int j = 0; j < n; j++) {
            rowSum += matrix[i][j];
            colSum += matrix[j][i];
        }

        if (rowSum != sumDiag1 || colSum != sumDiag1)
            return 0;

```

```c

    return 1;
}

void sparseMatrix(int matrix[MAX][MAX], int m, int n) {
    int sparse[MAX][3];
    int k = 0;

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (matrix[i][j] != 0) {
                sparse[k][0] = i;
                sparse[k][1] = j;
                sparse[k][2] = matrix[i][j];
                k++;
            }
        }
    }

    printf("Sparse Matrix Representation:\n");
    printf("Row Column Value\n");
    for (int i = 0; i < k; i++) {
        printf("%d %d %d\n", sparse[i][0], sparse[i][1],
            sparse[i][2]);
    }
}

```

```c
int main() {
    int matrix[MAX][MAX];
    int m, n, choice;

    printf("Enter the number of rows and columns of the matrix: ");
    scanf("%d %d", &m, &n);

    inputMatrix(matrix, m, n);

    do {
        printf("\nMenu:\n");
        printf("1. Find Saddle Point\n");
        printf("2. Check Magic Square\n");
        printf("3. Sparse Matrix Representation\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                findSaddlePoint(matrix, m, n);
                break;
            case 2:
                if (m != n) {
                    printf("Matrix is not square. Please enter a
                        square matrix:\n");

                    scanf("%d", &m);
                    n = m;
                    inputMatrix(matrix, m, n);
                }
                if (isMagicSquare(matrix, n)) {
                    printf("The matrix is a magic square.\n");
                } else {
                    printf("The matrix is not a magic square.\n");
                }
                break;
            case 3:
                sparseMatrix(matrix, m, n);
                break;
            case 4:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 4);

    return 0;
}
```

Output:-1)Saddle Point and Sparse Matrix:-

```
Enter the number of rows and columns of the matrix: 2
2
Enter the elements of the matrix row by row:
Row 1: 1
2
Row 2: 2
3

Menu:
1. Find Saddle Point
2. Check Magic Square
3. Sparse Matrix Representation
4. Exit
Enter your choice: 1
Saddle point found at (1, 0) with value 2

Menu:
1. Find Saddle Point
2. Check Magic Square
3. Sparse Matrix Representation
4. Exit
Enter your choice: 2
The matrix is not a magic square.
```

```
Menu:
1. Find Saddle Point
2. Check Magic Square
3. Sparse Matrix Representation
4. Exit
Enter your choice: 3
Sparse Matrix Representation:
Row Column Value
0 0 1
0 1 2
1 0 2
1 1 3

Menu:
1. Find Saddle Point
2. Check Magic Square
3. Sparse Matrix Representation
4. Exit
```

2)Magic Square:-

```
Enter the number of rows and columns of the matrix: 3
3
Enter the elements of the matrix row by row:
Row 1: 8
1
6
Row 2: 3
5
7
Row 3: 4
9
2

Menu:
1. Find Saddle Point
2. Check Magic Square
3. Sparse Matrix Representation
4. Exit
Enter your choice: 2
The matrix is a magic square.
```