

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: $m \times n$ 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:-

```
1
2 #include <stdio.h>
3
4 #define MAX 100
5
6 void inputMatrix(int matrix[MAX][MAX], int m, int n) {
7     printf("Enter the elements of the matrix row by row:\n");
8     for (int i = 0; i < m; i++) {
9         printf("Row %d: ", i + 1);
10        for (int j = 0; j < n; j++) {
11            scanf("%d", &matrix[i][j]);
12        }
13    }
14 }
15
16 void displayMatrix(int matrix[MAX][MAX], int m, int n) {
17     printf("Matrix:\n");
18     for (int i = 0; i < m; i++) {
19         for (int j = 0; j < n; j++) {
20             printf("%d ", matrix[i][j]);
21         }
22         printf("\n");
23     }
24 }
25
26 void findSaddlePoint(int matrix[MAX][MAX], int m, int n) {
27     int found = 0;
```



```

28     for (int i = 0; i < m; i++) {
29         int minRow = matrix[i][0];
30         int colIndex = 0;
31         for (int j = 1; j < n; j++) {
32             if (matrix[i][j] < minRow) {
33                 minRow = matrix[i][j];
34                 colIndex = j;
35             }
36         }
37
38         int isSaddle = 1;
39         for (int k = 0; k < m; k++) {
40             if (matrix[k][colIndex] > minRow) {
41                 isSaddle = 0;
42                 break;
43             }
44         }
45
46         if (isSaddle) {
47             printf("Saddle point found at (%d, %d) with value %d\n"
48                 , i, colIndex, minRow);
49             found = 1;
50         }
51     }

```

```

52     if (!found) {
53         printf("No saddle point found.\n");
54     }
55 }
56
57 int isMagicSquare(int matrix[MAX][MAX], int n) {
58     int sumDiag1 = 0, sumDiag2 = 0;
59     for (int i = 0; i < n; i++) {
60         sumDiag1 += matrix[i][i];
61         sumDiag2 += matrix[i][n - i - 1];
62     }
63
64     if (sumDiag1 != sumDiag2)
65         return 0;
66
67     for (int i = 0; i < n; i++) {
68         int rowSum = 0, colSum = 0;
69         for (int j = 0; j < n; j++) {
70             rowSum += matrix[i][j];
71             colSum += matrix[j][i];
72         }
73
74         if (rowSum != sumDiag1 || colSum != sumDiag1)
75             return 0;
76     }

```

```
77
78     return 1;
79 }
80
81 void sparseMatrix(int matrix[MAX][MAX], int m, int n) {
82     int sparse[MAX][3];
83     int k = 0;
84
85     for (int i = 0; i < m; i++) {
86         for (int j = 0; j < n; j++) {
87             if (matrix[i][j] != 0) {
88                 sparse[k][0] = i;
89                 sparse[k][1] = j;
90                 sparse[k][2] = matrix[i][j];
91                 k++;
92             }
93         }
94     }
95
96     printf("Sparse Matrix Representation:\n");
97     printf("Row Column Value\n");
98     for (int i = 0; i < k; i++) {
99         printf("%d %d %d\n", sparse[i][0], sparse[i][1],
100             sparse[i][2]);
101     }
102 }
```

```

103 - int main() {
104     int matrix[MAX][MAX];
105     int m, n, choice;
106
107     printf("Enter the number of rows and columns of the matrix: ");
108     scanf("%d %d", &m, &n);
109
110     inputMatrix(matrix, m, n);
111
112     do {
113         printf("\nMenu:\n");
114         printf("1. Find Saddle Point\n");
115         printf("2. Check Magic Square\n");
116         printf("3. Sparse Matrix Representation\n");
117         printf("4. Exit\n");
118         printf("Enter your choice: ");
119         scanf("%d", &choice);
120
121         switch (choice) {
122             case 1:
123                 findSaddlePoint(matrix, m, n);
124                 break;
125             case 2:
126                 if (m != n) {
127                     printf("Matrix is not square. Please enter a
128                         square matrix:\n");

```

```

129                     scanf("%d", &m);
130                     n = m;
131                     inputMatrix(matrix, m, n);
132                 }
133                 if (isMagicSquare(matrix, n)) {
134                     printf("The matrix is a magic square.\n");
135                 } else {
136                     printf("The matrix is not a magic square.\n");
137                 }
138                 break;
139             case 3:
140                 sparseMatrix(matrix, m, n);
141                 break;
142             case 4:
143                 printf("Exiting program.\n");
144                 break;
145             default:
146                 printf("Invalid choice. Please try again.\n");
147         }
148     } while (choice != 4);
149
150     return 0;
151 }

```

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.