Shrihari K Mangle
CSB2-74 12010517

# LAB-1

Assignment Based on Sparse Matrix: Implement the following Problem Statement.

a. Converting Normal matrix to its Sparse Representation.

Code :

```c
#include <stdio.h>
#include <stdlib.h>

void printArray(int arr[][3], int count)
{
    for (int i = 0; i < count; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
}
int main()
{
    int m = 4, n = 4;
    int s[4][4] = {{0, 0, 5, 2},
                   {0, 0, 0, 6},
                   {9, 0, 0, 0},
                   {7, 0, 0, 0}};

    int sr[7][3];
    int a, b, c, k;
    a = b = c = 0;
    k = 1;
    int count = 0;
    sr[0][0] = 4;
    sr[0][1] = 4;
    sr[0][2] = 5;

    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            if (s[i][j] != 0)
            {
                count++;
                a = i;
                b = j;
                c = s[i][j];

                sr[k][0] = a;
                sr[k][1] = b;
                sr[k][2] = c;

                k++;
            }
        }
    }
    printf("Sparse Matrix representation is : \n");
    printArray(sr, count+1);

    return 0;
}
```

Output :

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE                                  Code

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB> cd "e:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURE
S LAB\" ; if ($?) { gcc sparseMatrixRepre.c -o sparseMatrixRepre } ; if ($?) { .\sparseMatrixRepre }
Sparse Matrix representation is :
4 4 5
0 2 5
0 3 2
1 3 6
2 0 9
3 0 7
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```

b. Generate Simple Transpose of Sparse Matrix.

Code :

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

void printArray(int arr[][3], int count)
{
    for (int i = 0; i < count; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
}


void simpleTranspose(int rows, int col,int** arr)
{

    int st[rows][col];
    st[0][0] = 4;
    st[0][1] = 4;
    st[0][2] = 5;

    int count = 1;
    for (int i = 0; i < arr[0][1]; i++)
    {
        for (int j = 1; j <= arr[0][2]; j++)
        {
            if (arr[j][1] == i)
            {
                st[count][0] = arr[j][1];
                st[count][1] = arr[j][0];
                st[count][2] = arr[j][2];
                count++;
```

Shrihari K Mangle
CSB2-74 12010517

```
            }

        }

    }

    printArray(st, rows);
}


int main()
{
    int col, rows;
    col = 3, rows = 6;
    //int s[6][3];
    int** s = (int**)malloc(sizeof(int*) * 6);

    for(int i = 0; i < 6; i++) {
        s[i] =(int*)malloc(sizeof(int) * 3);
    }

    printf("Enter the sparse representation of the matrix : \n");
    s[0][0] = 4;
    s[0][1] = 4;
    s[0][2] = 5;
    for (int i = 1; i < rows; i++)
    {
        for (int j = 0; j < col; j++)
        {
            scanf("%d", &s[i][j]);
        }
    }


    printf("\n");
    printf("Simple transpose : \n");

    simpleTranspose(rows, col, s);

    //Freeing Memory
    for(int i = 0; i < 6; i++) {
        free(s[i]);
    }
    free(s);
    s = NULL;

    return 0;
}
```

Output :

```
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB> cd "e:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURE
S LAB\" ; if ($?) { gcc SimpleTranspose.c -o SimpleTranspose } ; if ($?) { .\SimpleTranspose }
Enter the sparse representation of the matrix :
0 2 5
0 3 2
1 3 6
2 0 9
3 0 7

Simple transpose :
4 4 5
0 2 9
0 3 7
2 0 5
3 0 2
3 1 6
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```

Shrihari K Mangle
CSB2-74 12010517

c. Generate Fast Transpose of Sparse Matrix.

Code :

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void printArray(int arr[5][3], int count)
{
    for (int i = 0; i < count; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
}

void FastTranspose(int rows, int col, int s[5][3], int val)
{
    int st[rows][col];
    st[0][0] = 3;
    st[0][1] = 2;
    st[0][2] = 4;

    int total[s[0][1]];

    for (int i = 0; i < col; i++)
    {
        total[i] = 0;
    }

    int index[s[0][1] + 1];

    for (int i = 0; i < col; i++)
    {
        int a = 0;
        for (int j = 1; j < rows; j++)
        {
            if (s[j][1] == i)
            {
                a++;
            }
        }
        total[i] = a;
    }

    index[0] = 1;
    for (int i = 1; i <= col; i++)
    {
        index[i] = index[i - 1] + total[i - 1];

    }

    for (int i = 1; i < rows; i++)
    {
        int loc = index[s[i][1]];
        st[loc][0] = s[i][1];
        st[loc][1] = s[i][0];
        st[loc][2] = s[i][2];
        index[s[i][1]] = index[s[i][1]] + 1;
    }

    printArray(st, rows);
```

Shrihari K Mangle
CSB2-74  12010517

```c
}

int main()
{
    int rows, col, val;
    rows = 5, col = 3, val = 5;

    int s[rows][col];

    printf("Enter sparse matrix\n");

    s[0][0] = 2;
    s[0][1] = 3;
    s[0][2] = 4;

    for (int i = 1; i < rows; i++)
    {
        for (int j = 0; j < col; j++)
        {
            s[i][j]=0;
            scanf("%d", &s[i][j]);
        }
    }

    printf("\n");
    printf("Fast Transpose is : \n");
    FastTranspose(rows, col, s, val);

    return 0;
}
```

Output :

```
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB> cd "e:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURE
S LAB\" ; if ($?) { gcc FastTranspose.c -o FastTranspose } ; if ($?) { .\FastTranspose }
Enter sparse matrix
0 1 3
0 2 5
1 0 9
1 2 8

Fast Transpose is :
3 2 4
0 1 9
1 0 3
2 0 5
2 1 8
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```

Shrihari K Mangle
CSB2-74 12010517

d. Finding a Saddle point in Given Matrix.

Code :

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int m = 3, n = 3;
    int a[m][n];

    printf("Enter matrix\n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &a[i][j]);
        }
    }

    for (int i = 0; i < m; i++)
    {
        int small = a[i][0];
        int loc = 0;

        for (int j = 0; j < n; j++)
        {
            if (a[i][j] < small)
            {
                small = a[i][j];
                loc = j;
            }
        }

        int large = a[0][loc];
        int locnew = 0;

        for (int k = 0; k < m; k++)
        {
            if (a[k][loc] > large)
            {
                large = a[k][loc];
                locnew = k;
            }
        }

        if (i == locnew) // a[i][loc] == a[locnew][loc]
        {
            printf("Saddle element is at %d %d location in the matrix and value = %d\n",
locnew, loc, a[locnew][loc]);
        }
    }

    return 0;
}
```

Output :

```
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB> cd "e:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURE
S LAB\" ; if ($?) { gcc saddle_point.c -o saddle_point } ; if ($?) { .\saddle_point }
Enter matrix
1 2 3
4 5 6
7 8 9
Saddle element is at 2 0 location in the matrix and value = 7
```