**Name – Akshat Jaiswal**

**Roll No. – 21052646**

**Section – CSE 37**

**DSA LAB 3**

1. **WAP for entering a sparse matrix and convert in triplet format.**

**Input:**

#include <stdio.h>

*int* main (*void*)

{

*int* mat1[100][100],m,n,count=0,y=0;

printf("Enter the number of rows of the 1st matrix: ");

scanf("%d",&m);

printf("Enter the number of columns of the 1st matrix: ");

scanf("%d",&n);

printf("Enter the elements in the matrix: ");

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

scanf("%d",&mat1[i][j]);

printf("\nElements in the matrix are:\n");

for (*int* i = 0; i < m; i++){

for (*int* j = 0; j < n; j++)

printf("%d  ",mat1[i][j]);

printf("\n");

}

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

    if((mat1[i][j]) != 0)

    count++;

*int* T1[count][3];

printf("Triplet form: \n");

printf("row\tcol\tvalue\n");

T1[0][0]=m;

T1[0][1]=n;

T1[0][2]=count;;

y=1;

for (*int* i = 0; i < count; i++)

{

for (*int* j = 0; j < n; j++)

{

    if((mat1[i][j]) != 0)

    {

    T1[y][0]=i;

    T1[y][1]=j;

    T1[y][2]=mat1[i][j];

    y++;

    // break;

    }

}

}

for (*int* i = 0; i < count; i++){

for (*int* j = 0; j < n; j++)

printf("%d\t",T1[i][j]);

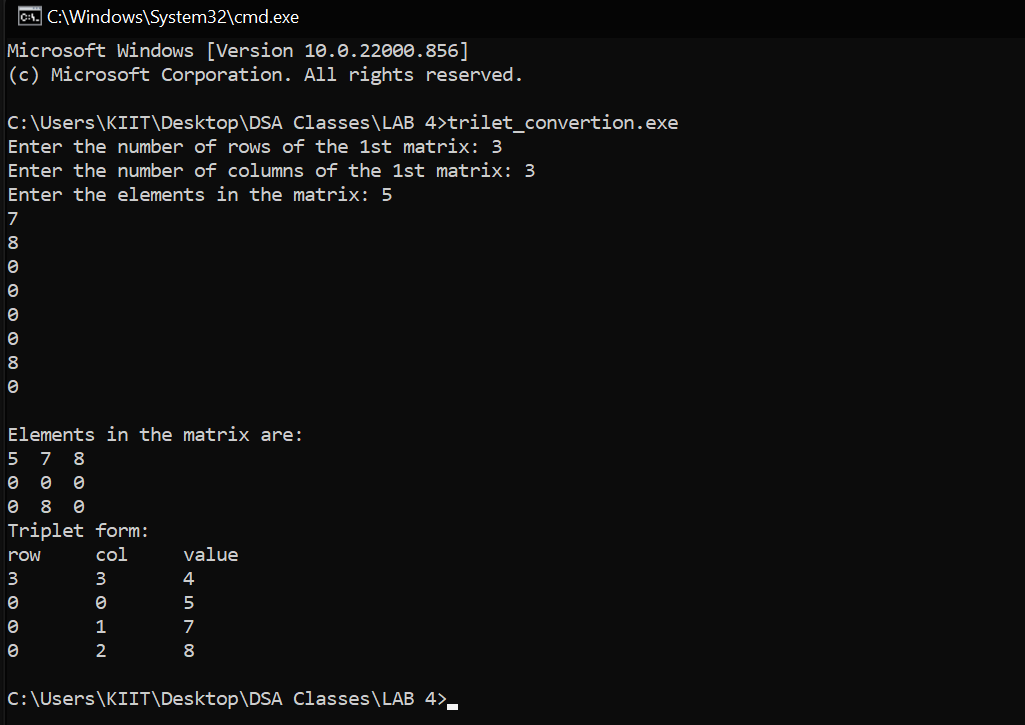
printf("\n");

}

return 0;

}

**Output:**

****

1. **WAP for to find the largest element of a sparse matrix using triplet format.**

**Input:**

#include <stdio.h>

*int* main (*void*)

{

*int* mat1[100][100],m,n,count=0,x=0,y=0,max=0;

printf("Enter the number of rows of the matrix: ");

scanf("%d",&m);

printf("Enter the number of columns of the matrix: ");

scanf("%d",&n);

printf("Enter the elements in the matrix: ");

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

scanf("%d",&mat1[i][j]);

printf("\nElements in the 1st matrix are:\n");

for (*int* i = 0; i < m; i++){

for (*int* j = 0; j < n; j++)

printf("%d  ",mat1[i][j]);

printf("\n");

}

for (*int* i = 0; i < m; i++)

{

for (*int* j = 0; j < n; j++)

{

    if((mat1[i][j])!=0){

    count++;

    }

}

}

*int* T1[count][3];

printf("\n\nTriplet form: \n");

printf("row\tcol\tvalue\n");

T1[0][0]=m;

T1[0][1]=n;

T1[0][2]=count;

y=1;

for (*int* i = 0; i <= m; i++)

{

for (*int* j = 0; j <= n; j++)

{

    if((mat1[i][j]) != 0)

    {

    T1[y][0]=i;

    T1[y][1]=j;

    T1[y][2]=mat1[i][j];

    y++;

    }

}

}

for (*int* i = 0; i < count; i++){

for (*int* j = 0; j < 3; j++)

printf("%d\t",T1[i][j]);

printf("\n");

}

max=T1[1][2];

for (*int* i = 1; i < count; i++)

{

for (*int* j = i+1; j < count; j++)

{

if(T1[j][2]>T1[i][2])

max=T1[j][2];

}

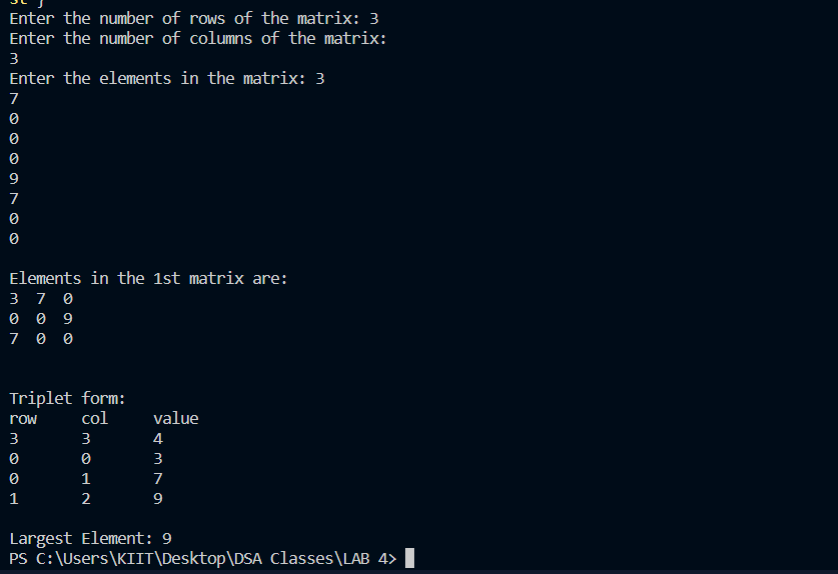
}

printf("\nLargest Element: %d",max);

return 0;

}

**Output:**



1. **WAP to check if a matrix is a lower triangular or upper triangular matrix.**

**Input:**

#include <stdio.h>

*int* main (*void*)

{

*int* mat1[100][100],m,n,count1=0,count2=0;

printf("Enter the number of rows of the matrix: ");

scanf("%d",&m);

printf("Enter the number of columns of the matrix: ");

scanf("%d",&n);

printf("Enter the elements in the matrix: ");

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

scanf("%d",&mat1[i][j]);

printf("\nElements in the matrix are:\n");

for (*int* i = 0; i < m; i++){

for (*int* j = 0; j < n; j++)

printf("%d  ",mat1[i][j]);

printf("\n");

}

for (*int* i = 0; i < m; i++)

{

for (*int* j = 0; j < n; j++)

{

if(i>j && mat1[i][j]==0)

    count1++;

else if(j>i && mat1[i][j]==0)

    count2++;

}

}

if(count1>count2)

printf("Matrix is in Lower Triangular Form!");

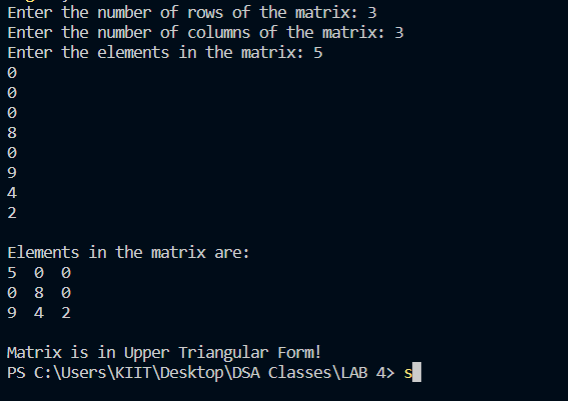
else

printf("Matrix is in Upper Triangular Form!");

return 0;

}

**Output:**

****

1. **WAP for finding the transpose of a sparse matrix using triplet format.**

**Input:**

#include <stdio.h>

*int* main (*void*)

{

*int* mat1[100][100],m,n,count=0,y=0;

printf("Enter the number of rows of the 1st matrix: ");

scanf("%d",&m);

printf("Enter the number of columns of the 1st matrix: ");

scanf("%d",&n);

printf("Enter the elements in the matrix: ");

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

scanf("%d",&mat1[i][j]);

printf("\nElements in the matrix are:\n");

for (*int* i = 0; i < m; i++){

for (*int* j = 0; j < n; j++)

printf("%d  ",mat1[i][j]);

printf("\n");

}

for (*int* i = 0; i < m; i++)

for (*int* j = 0; j < n; j++)

    if((mat1[i][j]) != 0)

    count++;

*int* T1[100][3];

printf("\nTriplet form: \n");

printf("row\tcol\tvalue\n");

T1[0][0]=m;

T1[0][1]=n;

T1[0][2]=count;;

y=1;

for (*int* i = 0; i < count; i++)

{

for (*int* j = 0; j < n; j++)

{

    if((mat1[i][j]) != 0)

    {

    T1[y][0]=i;

    T1[y][1]=j;

    T1[y][2]=mat1[i][j];

    y++;

    // break;

    }

}

}

for (*int* i = 0; i < count; i++){

for (*int* j = 0; j < n; j++)

printf("%d\t",T1[i][j]);

printf("\n");

}

printf("\n\nTranspose:\n");

printf("row\tcol\tvalue\n");

for (*int* i = 0; i < count; i++)

{

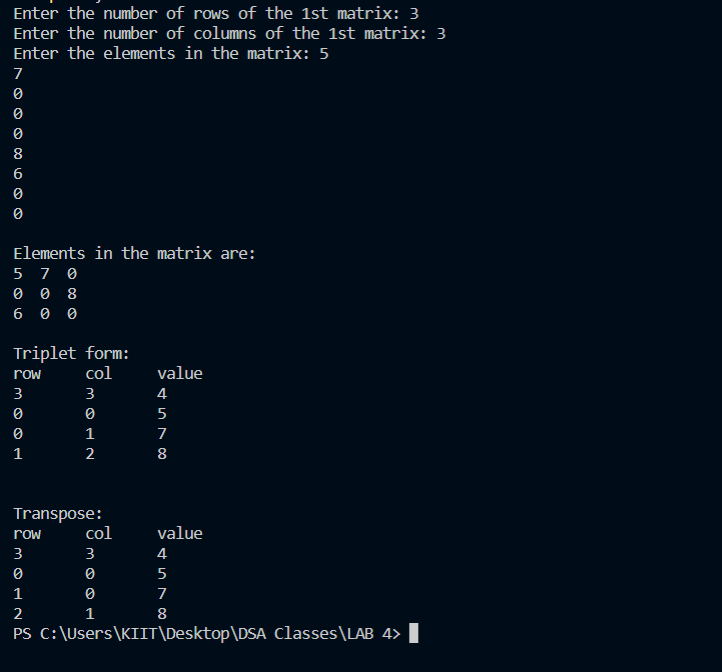
    printf("%d\t%d\t%d\n",T1[i][1],T1[i][0],T1[i][2]);

}

return 0;

}

**Output:**

****

1. **WAP for the addition of two sparse matrices using triplet format.**

**Input:**

#include <stdio.h>

*void* product();

*int* main(*void*)

{

*int* mat1[100][100], mat2[100][100], m, n, p, q, count1 = 0, count2 = 0;

    printf("Enter the number of rows of the 1st matrix: ");

    scanf("%d", &m);

    printf("Enter the number of columns of the 1st matrix: ");

    scanf("%d", &n);

    printf("Enter the number of rows of the 2nd matrix: ");

    scanf("%d", &p);

    printf("Enter the number of columns of the 2nd matrix: ");

    scanf("%d", &q);

    printf("Enter the elements in the 1st matrix: ");

    for (*int* i = 0; i < m; i++)

        for (*int* j = 0; j < n; j++)

            scanf("%d", &mat1[i][j]);

    printf("Enter the elements in the 2nd matrix: ");

    for (*int* i = 0; i < p; i++)

        for (*int* j = 0; j < q; j++)

            scanf("%d", &mat2[i][j]);

    printf("\nElements in the 1st matrix are:\n");

    for (*int* i = 0; i < m; i++)

    {

        for (*int* j = 0; j < n; j++)

            printf("%d  ", mat1[i][j]);

        printf("\n");

    }

    printf("\nElements in the 2nd matrix are:\n");

    for (*int* i = 0; i < p; i++)

    {

        for (*int* j = 0; j < q; j++)

            printf("%d  ", mat2[i][j]);

        printf("\n");

    }

        // checking for non zero in first matrix

        for (*int* i = 0; i < m; i++)

            for (*int* j = 0; j < n; j++)

                if ((mat1[i][j]) != 0)

                    count1++;

        // checking for non zero in second matrix

        for (*int* i = 0; i < p; i++)

            for (*int* j = 0; j < q; j++)

                if ((mat2[i][j]) != 0)

                    count2++;

        // creating transpose triplet

*int* T1[100][3];

*int* T2[100][3];

*int* T3[100][3];

        // initializing first row of 1st triplet

        T1[0][0] = m;

        T1[0][1] = n;

        T1[0][2] = count1;

        // initializing first row of 2nd triplet

        T2[0][0] = p;

        T2[0][1] = q;

        T2[0][2] = count2;

*int* x=1,y=1,z=1;

        //creating first triplet

        for (*int* i = 0; i < count1; i++)

        {

            for (*int* j = 0; j < n; j++)

            {

                if ((mat1[i][j]) != 0)

                {

                    T1[x][0] = i;

                    T1[x][1] = j;

                    T1[x][2] = mat1[i][j];

                    x++;

                    // break;

                }

            }

        }

        //creating second triplet

        for (*int* i = 0; i < count2; i++)

        {

            for (*int* j = 0; j < q; j++)

            {

                if ((mat2[i][j]) != 0)

                {

                    T2[y][0] = i;

                    T2[y][1] = j;

                    T2[y][2] = mat2[i][j];

                    y++;

                    // break;

                }

            }

        }

        printf("\nTriplet form of 1st Matrix: \n");

        printf("row\tcolumn\tvalue\n");

        for (*int* i = 0; i < count1; i++)

        {

            for (*int* j = 0; j < 3; j++)

                printf("%d\t", T1[i][j]);

            printf("\n");

        }

        printf("\nTriplet form of 2nd Matrix: \n");

        printf("row\tcolumn\tvalue\n");

        for (*int* i = 0; i < count2; i++)

        {

            for (*int* j = 0; j < 3; j++)

                printf("%d\t", T2[i][j]);

            printf("\n");

        }

        //creating 3rd triplet

        T3[0][0] = m + p;

        T3[0][1] = n + q;

        T3[0][2] = count1 + count2;

        x=1,y=1,z=1;

        while(x<count1 && y<count2)

        {

            if (T1[x][0] < T2[y][0])

            {

                T3[z][0] = T1[x][0];

                T3[z][1] = T1[x][1];

                T3[z][2] = T1[x][2];

                x++;

                z++;

            }

            else if (T2[y][0] < T1[x][0])

            {

                T3[z][0] = T2[y][0];

                T3[z][1] = T2[y][1];

                T3[z][2] = T2[y][2];

                y++;

                z++;

            }

            else if (T1[x][0] == T2[y][0])

            {

                if (T1[x][1] == T2[y][1])

                {

                    T3[z][0] = T1[x][0];

                    T3[z][1] = T1[x][1];

                    T3[z][2] = T1[x][2] + T2[y][2];

                    x++;

                    y++;

                    z++;

                }

            else if (T1[x][1] < T2[y][1])

            {

                T3[z][0] = T1[x][0];

                T3[z][1] = T1[x][1];

                T3[z][2] = T1[x][2];

                x++;

                z++;

            }

            else (T2[y][1] < T1[x][1]);

            {

                T3[z][0] = T2[y][0];

                T3[z][1] = T2[y][1];

                T3[z][2] = T2[y][2];

                y++;

                z++;

            }

            }

        }

        printf("\nTriplet form of 3rd Added Matrix: \n");

        printf("row\tcolumn\tvalue\n");

        for (*int* i = 0; i < (count1 + count2); i++)

        {

            for (*int* j = 0; j < 3; j++)

                printf("%d\t", T3[i][j]);

            printf("\n");

        }

    return 0;

    }

**Output:**

