Lab assignment ||

Name : Raj Roushan Date : 11/09/20

Sch Id : 1912089 Sec : B

Question :-

Given two matrices, A[ ][ ] and B[ ][ ] of mxn size. Most of the entries are null in both matrices. Take user input. Display output with proper format.

1. Represent the matrices in sparse matrix format.

2. Add these two matrices.

3. Transpose any one matrix.

4. Multiply these two matrices.

Ans :-

#include<stdio.h>

void swap(int \*a, int \*b)

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

void addition(int ktemp[3][100], int ltemp[3][100], int k, int l);

void sort(int k[3][100], int count);

void transpose(int k[3][100], int count);

void multiply(int k[3][100], int count, int r1, int c1);

int main() {

int m,n;

printf("Enter no of rows and columns\n");

scanf("%d",&m);

scanf("%d",&n);

int A[m][n];

int B[m][n];

printf("Enter matrix A");

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

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

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

}

}

printf("Enter matrix B");

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

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

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

}

}

int size = 0;

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

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

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

size++;

int AS[3][100];

int y=size;

size = 0;

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

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

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

size++;

int BS[3][100];

int z=size;

int k = 0;

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

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

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

{

AS[0][k] = i;

AS[1][k] = j;

AS[2][k] = A[i][j];

k++;

}

k=0;

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

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

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

{

BS[0][k] = i;

BS[1][k] = j;

BS[2][k] = B[i][j];

k++;

}

int qq=1;

//printf("1) View Sparse matrices\n2)Add\n3)Transpose of second matrix\n4)Multiply matrix 1 with matrix you want\n");

while(qq){

printf("1) View Sparse matrices\n2)Add\n3)Transpose of second matrix\n4)Multiply matrix 1 with matrix you want\n5)Enter Any other no to terminate\n");

int kk;

int sum[m][n];

scanf("%d",&kk);

switch(kk) {

case 1:

printf("Sparse matrix of A-\n");

for (int i = 0; i < 3; i++) {

for (int j = 0; j < y; j++) {

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

}

printf("\n");

}

printf("Sparse matrix of B-\n");

for (int i = 0; i < 3; i++) {

for (int j = 0; j < z; j++) {

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

}

printf("\n");

};

break;

case 2:

printf("Addition of A&B in Sparse form-\n");

addition(AS,BS,y,z);

break;

case 3:

printf("Transpose of B in Sparse form-\n");

transpose(BS, y);

for (int i = 0; i < 3; i++) {

for (int j = 0; j < y; j++) {

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

}

printf("\n");

}

break;

case 4:

printf("Multiplication of A&B in Sparse form-\n");

multiply(AS, y, m, n);

break;

default:

qq=0;

break;

}

}

}

void addition(int ktemp[3][100], int ltemp[3][100], int k, int l)

{

int i = 0, j = 0, sparse[3][100], x = 0;

while (i < k && j < l) {

if ((ktemp[0][i] == ltemp[0][j]) && (ltemp[1][j] == ktemp[1][i])) {

sparse[0][x] = ktemp[0][i];

sparse[1][x] = ktemp[1][i];

sparse[2][x] = ltemp[2][j] + ktemp[2][i];

x++;

i++;

j++;

}

else {

if (ktemp[0][i] < ltemp[0][j]) {

sparse[0][x] = ktemp[0][i];

sparse[1][x] = ktemp[1][i];

sparse[2][x] = ktemp[2][i];

x++;

i++;

}

else {

if ((ktemp[0][i] == ltemp[0][j]) && (ktemp[1][i] < ltemp[1][j])) {

sparse[0][x] = ktemp[0][i];

sparse[1][x] = ktemp[1][i];

sparse[2][x] = ktemp[2][i];

x++;

i++;

}

else {

sparse[0][x] = ktemp[0][j];

sparse[1][x] = ktemp[1][j];

sparse[2][x] = ktemp[2][j];

x++;

j++;

}

}

}

}

while (i < k) {

sparse[0][x] = ktemp[0][i];

sparse[1][x] = ktemp[1][i];

sparse[2][x] = ktemp[2][i];

x++;

i++;

}

while (j < l) {

sparse[0][x] = ktemp[0][j];

sparse[1][x] = ktemp[1][j];

sparse[2][x] = ktemp[2][j];

x++;

j++;

}

for (i = 0; i < 3; i++) {

for (j = 0; j < x; j++) {

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

}

printf("\n");

}

}

void sort(int k[3][100], int count) {

int i, j;

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

{

for (j = 0; j < count - i - 1; j++)

{

if (k[0][j] > k[0][j + 1])

{

swap(&k[0][j], &k[0][j + 1]);

swap(&k[1][j], &k[1][j + 1]);

swap(&k[2][j], &k[2][j + 1]);

}

else if (k[0][j] == k[0][j + 1])

{

if (k[1][j] > k[1][j + 1])

{

swap(&k[0][j], &k[0][j + 1]);

swap(&k[1][j], &k[1][j + 1]);

swap(&k[2][j], &k[2][j + 1]);

}

}

}

}

}

void transpose(int k[3][100], int count) {

int i, j, temp;

printf("\n");

for (j = 0; j < count; j++)

{

swap(&k[0][j], &k[1][j]);

}

sort(k, count);

}

void multiply(int k[3][100], int count, int r1, int c1) {

int b[20][20], l[3][100], i, j, r2, c2, size = 0, kpos, lpos, result[3][100], r, c, tempk, templ, sum, rcount = 0;

printf("Enter no of rows");

scanf("%d", &r2);

printf("Enter no of columns");

scanf("%d", &c2);

for (i = 0; i < r2; i++)

{

for (j = 0; j < c2; j++)

{

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

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

{

l[0][size] = i;

l[1][size] = j;

l[2][size] = b[i][j];

size++;

}

}

}

if (c1 != r2)

{

printf("Not possible");

return;

}

transpose(l, size);

for (kpos = 0; kpos < count;)

{

r = k[0][kpos];

for (lpos = 0; lpos < size;)

{

c = l[0][lpos];

tempk = kpos;

templ = lpos;

sum = 0;

while (tempk < count && k[0][tempk] == r && templ < size && l[0][templ] == c)

{

if (k[1][tempk] < l[1][templ])

{

tempk++;

}

else if (l[1][templ] > k[1][tempk])

{

templ++;

}

else

{

sum += k[2][tempk++] \* l[2][templ++];

}

}

if (sum != 0)

{

result[0][rcount] = r;

result[1][rcount] = c;

result[2][rcount] = sum;

rcount++;

}

while (lpos < size && l[0][lpos] == c)

{

lpos++;

}

}

while (kpos < count && k[0][kpos] == r)

{

kpos++;

}

}

// printf(result, rcount);

for(int i=0;i<3;i++) {

for(int j=0;j<rcount;j++) {

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

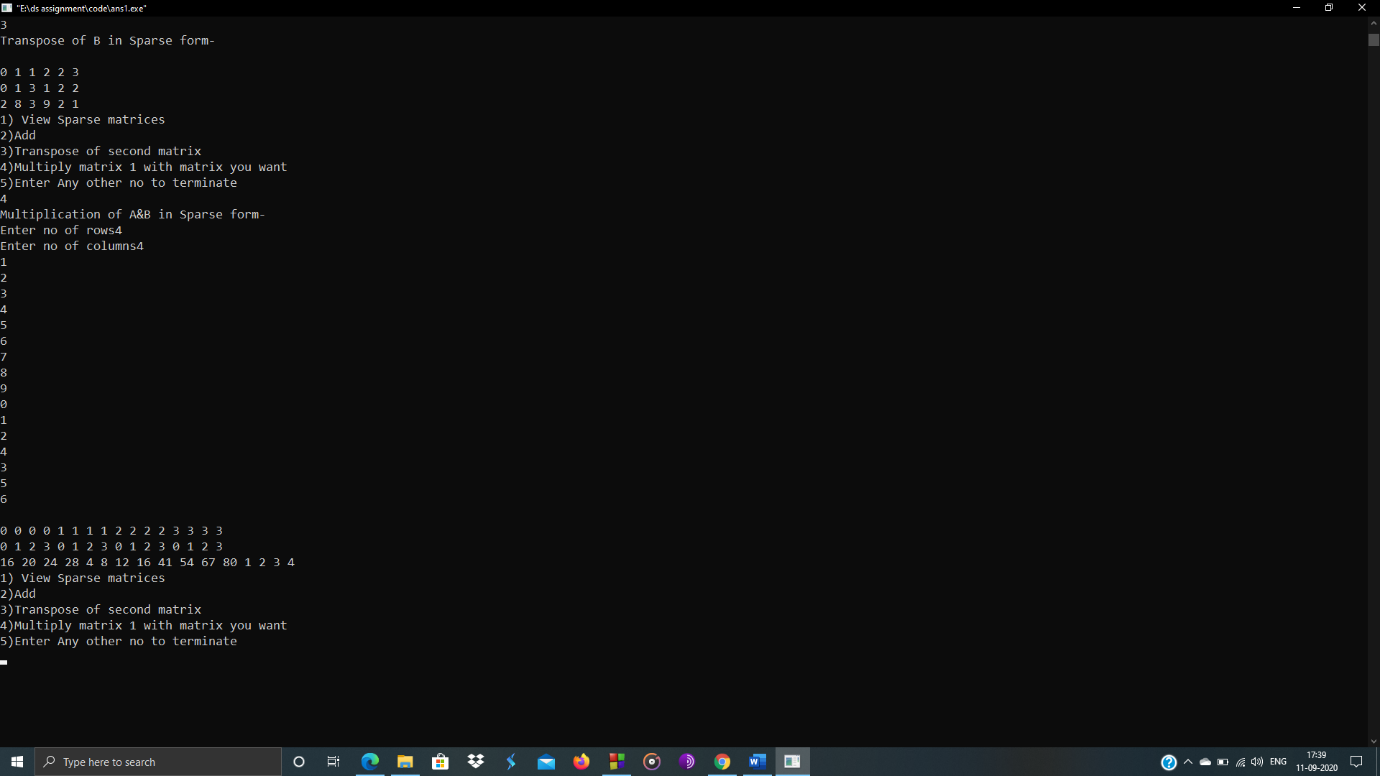
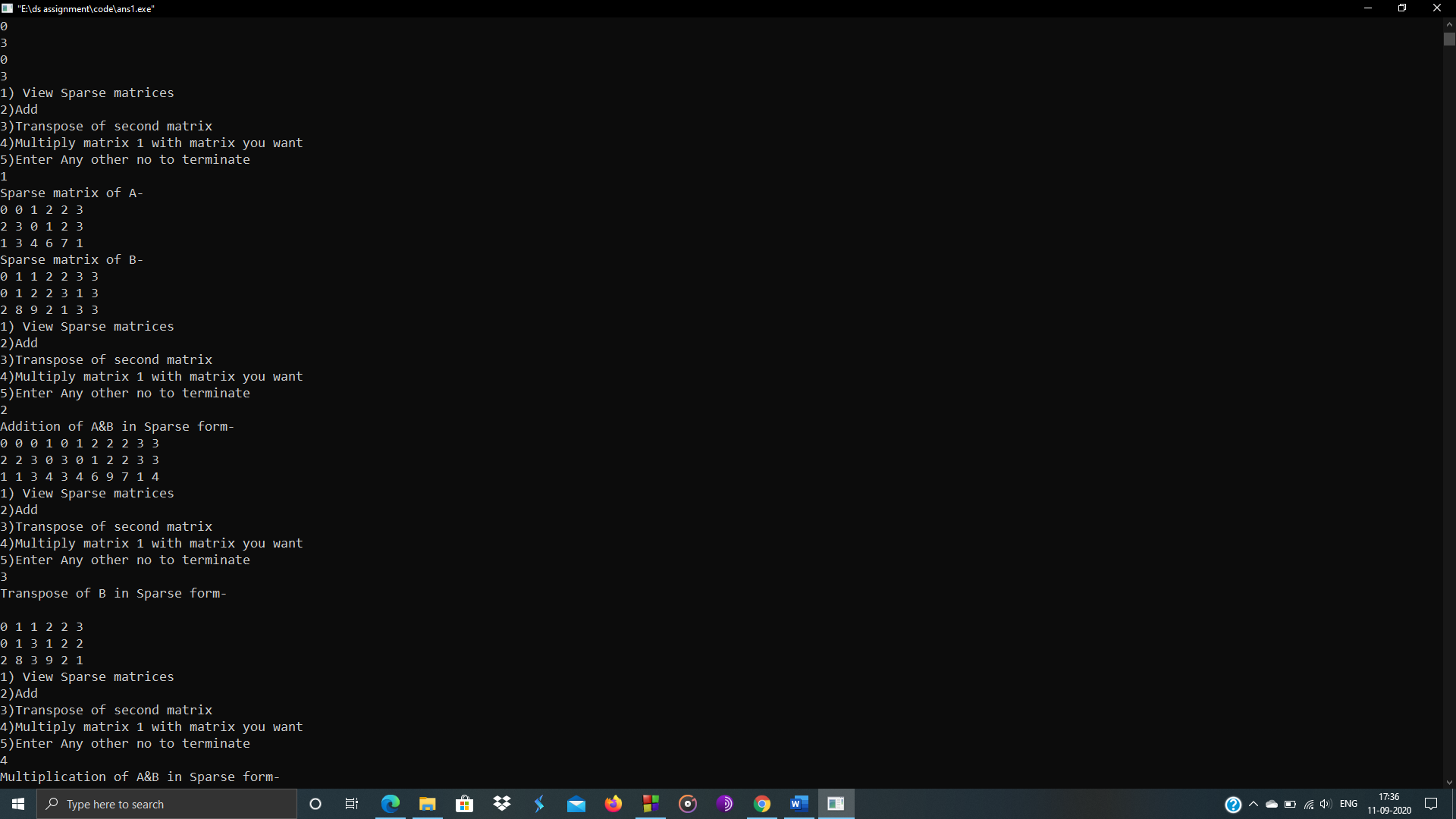
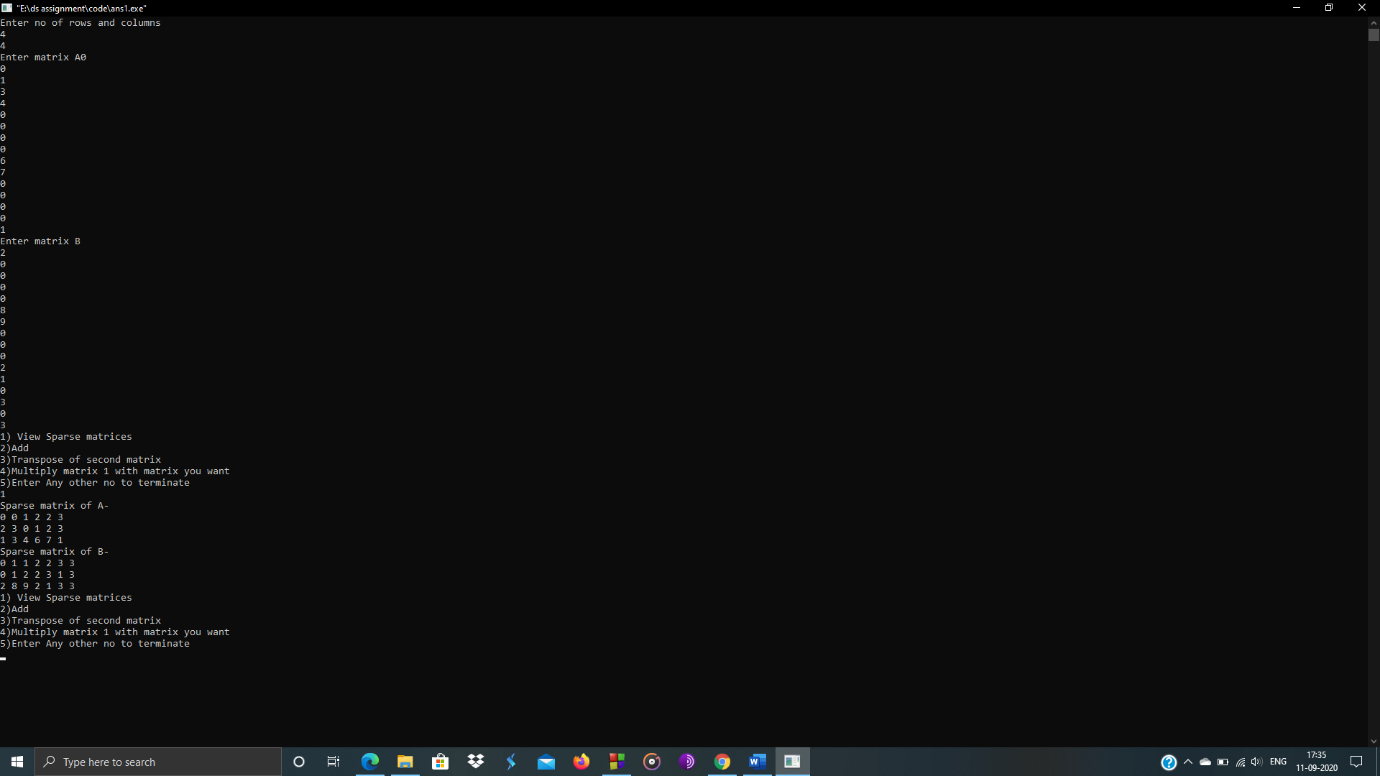
}

printf("\n");

}

}

Screen shot :



The end