NATIONAL INSTITUTE OF TECHNOLOGY, SILCHAR

Subject: CS211

Submitted By:

Subhojit Ghimire

1912160

B.Tech. IIIrd Sem

CSE-B

Q.1, Q.2, and Q.3 (Sparse Matrix, Addition of Matrices and Transpose of one of the matrices)

#include <stdio.h>

void AddMat(int matA[3][100], int matB[3][100], int k, int l);

void TranMat (int matA[3][100], int k);

int main()

{

int a[20][20], matA[3][100], n, m, count = 0, i, j, k = 0, b[20][20], matB[3][100], l = 0, p, q;

printf("Enter the size of the matrix: \n");

printf ("m: ");

scanf ("%d", &m);

printf("n: ");

scanf("%d", &n);

printf ("Enter Matrix A:\n");

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

{

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

{

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

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

{

matA[0][k] = i;

matA[1][k] = j;

matA[2][k] = a[i][j];

k++;

}

}

}

printf ("Enter Matrix B:\n");

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

{

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

{

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

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

{

matB[0][l] = i;

matB[1][l] = j;

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

l++;

}

}

}

printf("\nSparse matrix A is \n");

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

{

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

{

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

}

printf("\n");

}

printf("\nSparse matrix B is \n");

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

{

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

{

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

}

printf("\n");

}

printf ("### ADDITION ###\n"); AddMat(matA, matB, k, l);

printf ("### TRANSPOSE of matrix A ###\n"); TranMat (matA, k);

return 0;

}

void AddMat(int matA[3][100], int matB[3][100], int k, int l)

{

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

while (i < k && j < l)

{

if ((matA[0][i] == matB[0][j]) && (matB[1][j] == matA[1][i]))

{

printf("Entered this");

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

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

sparse[2][x] = matB[2][j] + matA[2][i];

x++;

i++;

j++;

}

else

{

if (matA[0][i] < matB[0][j])

{

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

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

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

x++;

i++;

}

else

{

if ((matA[0][i] == matB[0][j]) && (matA[1][i] < matB[1][j]))

{

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

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

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

x++;

i++;

}

else

{

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

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

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

x++;

j++;

}

}

}

}

while (i < k)

{

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

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

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

x++;

i++;

}

while (j < l)

{

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

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

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

x++;

j++;

}

printf("\nAddition of the Matrix is \n");

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

{

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

{

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

}

printf("\n");

}

}

void TranMat (int matA[3][100], int k)

{

int temp, i, j;

int transpose [3][k];

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

{

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

{

transpose [0][j] = matA [1][j];

transpose [1][j] = matA [0][j];

transpose [2][j] = matA [2][j];

}

printf ("Transpose of the matrix is\n");

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

{

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

{

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

}

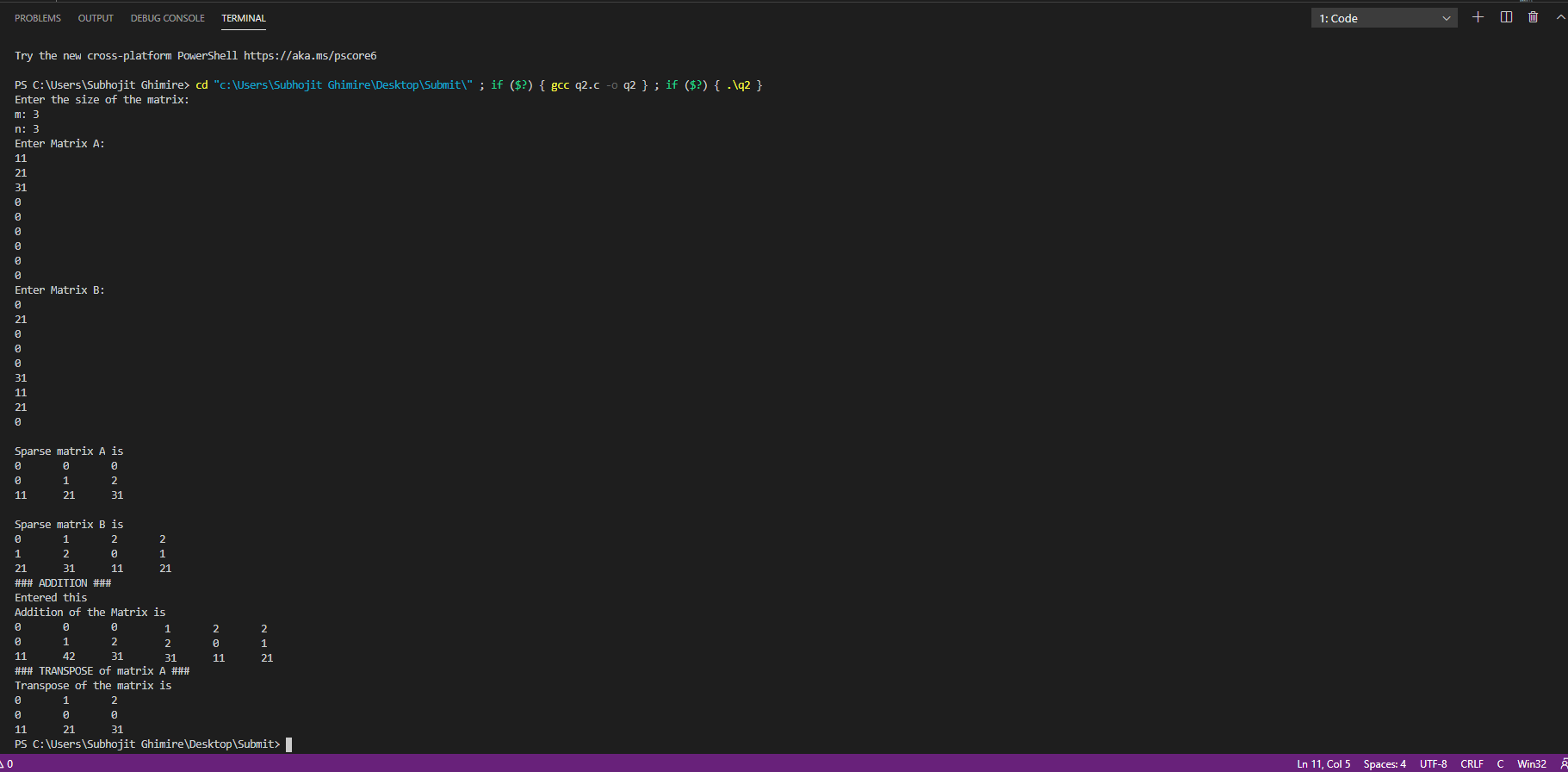
printf ("\n");

}

}

}

OUTPUT: (for 1, 2 ,3)



Q.4. Sparse Matrix Multiplication

#include <stdio.h>

void print(int k[3][100], int count)

{

int i, j;

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

{

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

{

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

}

printf("\n");

}

}

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

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

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 size of the Matrix B\n");

printf ("m: ");

scanf("%d", &r2);

printf ("n: ");

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 valid");

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++;

}

}

print(result, rcount);

}

int main()

{

int a[20][20], k[3][100], i, j, m, n, count = 0;

printf ("Enter size of the Matrix A\n");

printf("m: ");

scanf("%d", &m);

printf("n: ");

scanf("%d", &n);

printf ("Enter values in Matrix A: ");

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

{

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

{

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

if (a[i][j])

{

k[0][count] = i;

k[1][count] = j;

k[2][count] = a[i][j];

count++;

}

}

}

multiply(k, count, m, n);

}

OUTPUT:

