**Topic – Transpose of Sparse Matrix with algorithm O(n+c) with Extra Space**

* **Problem Statement**

**Part-b**

b) Write another program to  transpose a sparse matrix in O( n + c) times with extra space, where n and c are the number of non-zero values and number of columns in the matrix. Note that, the transposed matrix must have its rows and columns within rows in ascending order.

**Input example:**

Enter the number of rows in sparse matrix

7

Enter the number of columns in sparse matrix

6

Enter the number of nonzero elements in sparse matrix

8

Enter the row index for 1th element in sparse matrix

0

Enter the column index for 1th element in sparse matrix

4

Enter the 1th element in sparse matrix

15

Enter the row index for 2th element in sparse matrix

1

Enter the column index for 2th element in sparse matrix

2

Enter the 2th element in sparse matrix

67

Enter the row index for 3th element in sparse matrix

3

Enter the column index for 3th element in sparse matrix

5

Enter the 3th element in sparse matrix

45

Enter the row index for 4th element in sparse matrix

3

Enter the column index for 4th element in sparse matrix

1

Enter the 4th element in sparse matrix

23

Enter the row index for 5th element in sparse matrix

4

Enter the column index for 5th element in sparse matrix

4

Enter the 5th element in sparse matrix

21

Enter the row index for 6th element in sparse matrix

5

Enter the column index for 6th element in sparse matrix

0

Enter the 6th element in sparse matrix

35

Enter the row index for 7th element in sparse matrix

5

Enter the column index for 7th element in sparse matrix

2

Enter the 7th element in sparse matrix

19

Enter the row index for 8th element in sparse matrix

6

Enter the column index for 8th element in sparse matrix

5

Enter the 8th element in sparse matrix

29

**Output example:**

The sparse matrix taken from the user is:

Row Column Nonzero values

7 6 8

0 4 15

1 2 67

3 5 45

3 1 23

4 4 21

5 0 35

5 2 19

6 5 29

The transpose of the sparse matrix is:

Row Column Nonzero values

6 7 8

0 5 35

1 3 23

2 1 67

2 5 19

4 0 15

4 4 21

5 3 45

5 6 29

* **Proposed C Code**

**/\* ---------- sparse-part-b.c--------------- \*/**

**#include<stdio.h>**

**#include<stdlib.h>**

**typedef struct sparse\_matrix**

**{**

**int row;**

**int col;**

**int nonzero;**

**}sparse; /\* structure for representing rows,colulmns and nonzero values in sparse matrix \*/**

**int main()**

**{**

**int row\_count,col\_count,nonzero\_count;**

**/\* the counts of row,col,non zero elements are taken from user \*/**

**printf("Enter the number of rows in sparse matrix\n");**

**scanf("%d",&row\_count);**

**printf("Enter the number of columns in sparse matrix\n");**

**scanf("%d",&col\_count);**

**printf("Enter the number of nonzero elements in sparse matrix\n");**

**scanf("%d",&nonzero\_count);**

**/\* Array arr of structure sparse is taken to represent sparse matrix \*/**

**sparse\* arr = (sparse\*)malloc((nonzero\_count+1)\*sizeof(sparse));**

**/\* Array arr of structure transpose is taken to represent the transpose of sparse matrix \*/**

**sparse\* transpose = (sparse\*)malloc((nonzero\_count+1)\*sizeof(sparse));**

**/\* Array for frequency of non zero elements in each column \*/**

**int\* frequency = (int\*)calloc(col\_count,sizeof(int));**

**/\* Array startingPosition for the starting positions of columns where non zero elements are present \*/**

**int\* startingPosition = (int\*)calloc(col\_count,sizeof(int));**

**arr[0].row = row\_count;**

**arr[0].col = col\_count;**

**arr[0].nonzero = nonzero\_count;**

**/\* Taking input data of sparse matrix with increasing order of rows and columns \*/**

**for ( int i = 1 ; i <= nonzero\_count ; i++ )**

**{**

**printf("Enter the row index for %dth element in sparse matrix\n",i);**

**scanf("%d",&arr[i].row);**

**printf("Enter the column index for %dth element in sparse matrix\n",i);**

**scanf("%d",&arr[i].col);**

**frequency[arr[i].col]++;**

**/\* calculating the frequency of non zero elements in each column \*/**

**printf("Enter the %dth element in sparse matrix\n",i);**

**scanf("%d",&arr[i].nonzero);**

**}**

**printf("The sparse matrix taken from the user is: \n");/\* printing the representation of sparse matrix \*/**

**printf("Row\tColumn\tNonzero values\n");**

**for ( int i = 0 ; i <= nonzero\_count ; i++ )**

**{**

**printf("%3d\t%3d\t%3d\n",arr[i].row,arr[i].col,arr[i].nonzero);**

**}**

**startingPosition[0] = 1;/\* Initial starting position of column 0 is 1 in transpose matrix \*/**

**for ( int i = 1 ; i < col\_count ; i++ )**

**{**

**startingPosition[i] = startingPosition[i-1] + frequency[i-1];**

**/\* starting position of ith column comes after all the elements of i-1 th column so frequency[i-1] is also added \*/**

**}**

**/\* The col\_count of arr and row\_count of transpose are same and vice-versa \*/**

**transpose[0].row = col\_count;**

**transpose[0].col = row\_count;**

**transpose[0].nonzero = nonzero\_count;**

**for ( int i = 1 ; i <= nonzero\_count ; i++ )/\* Transposing \*/**

**{**

**transpose[startingPosition[arr[i].col]].row = arr[i].col;**

**transpose[startingPosition[arr[i].col]].col = arr[i].row;**

**transpose[startingPosition[arr[i].col]].nonzero = arr[i].nonzero;**

**startingPosition[arr[i].col]++;**

**/\* Increasing the startingPosition for a respective column as the same column if encountered,it will come exactly after it \*/**

**}**

**printf("The transpose of the sparse matrix is: \n");/\* Printing the transpose of sparse matrix \*/**

**printf("Row\tColumn\tNonzero values\n");**

**for ( int i = 0 ; i <= nonzero\_count ; i++ )**

**{**

**printf("%3d\t%3d\t%3d\n",transpose[i].row,transpose[i].col,transpose[i].nonzero);**

**}**

**/\* freeing the memory \*/**

**free(arr);**

**free(transpose);**

**free(frequency);**

**free(startingPosition);**

**return 0;**

**}**

**/\*------------------------------------------------------------------------------------------------------------------------- \*/**

* **Conclusion**

**The proposed algorithm of the transpose of sparse matrix problem part-b has a runtime of O(n+c) where n is the number of nonzero elements and c is the number of columns of the sparse matrix.**

* **Limitations : This program takes time complexity of O(n+c) i.e linear time complexity but it requires extra space of O(n).**
* **Assumptions: The user should give the rows and colums of the non zero elements of the sparse matrix such that they are in increasing order.**