Sparse matrix

```
#include <iostream>

#include <fstream>

using namespace std;


/***********************************NODE
CLASS************************************/

//a node class to make nodes for linked list

class node

{

public:

   int row;

   int col;

   int value;

   node *next;

   node(int r, int c, int v);

};


//parameterised constructor of node class

node::node(int r, int c, int v)

{

   this->row = r;

   this->col = c;

   this->value = v;

   this->next = NULL;

}


/***********************************SPARSE MATRIX
CLASS************************************/

//a class for performing various functions on the Sparse Matrix
```

```
class sparseMatrix
{
private:
   int **m;
   int r = 0;  //stores no. of rows in the matrix
   int c = 0;  //stores no. of cols in the matrix
   int nz = 0; //stores the no. non-zero elements in the mattrix

   //for array
   int **a;
   int **spaMat;

   //for linked list
   node *head = NULL;
   int count = 0;

public:
   sparseMatrix(){};  //constructor
   ~sparseMatrix(){}; //destructor

   void input();        //method to take input for matrix from the file
   void sparseToArray(); //converts sparse to array
   void arrayToSparse(); //converts array to sparse
   void sparseToLL();    //converts sparse to linked list
   void llToSparse();    //converts linked list to sparse
};

//takes the input for sparse matrix from the input file
void sparseMatrix::input()
{
   /*************************FILE HANDLING************************************/
```

```
//creating an object of ifstream class
ifstream inFile;

//opening the input file to take set data from it
inFile.open("SparseMatrix.txt");

//checking if the input file opens successfully or not
if (!inFile)
{
   cerr << "Error opening in file 1\n";
   exit(100);
}
/***********************************************************************************/

//taking input for rows and cols from the text file
inFile >> r;
inFile >> c;

//allocating memory to the 2d array of matrix dynamically
m = new int *[r];
for (int i = 0; i < r; i++)
{
   m[i] = new int[c];
}

//taking input for the matrix from the file
for (int i = 0; i < r; i++)
{
   for (int j = 0; j < c; j++)
   {
      inFile >> m[i][j];
```

```
        }
    }

    //printing the Matrix of the graph
    cout << ">>>Given Matrix is: " << endl;
    for (int i = 0; i < r; i++)
    {
        for (int j = 0; j < c; j++)
        {
            if (m[i][j] != 0)
                nz++;
            cout << m[i][j] << "\t";
        }
        cout << endl;
    }

    if (nz > ((r * c) - nz))
        cout << "*The given Matrix is not a Sparse Matrix" << endl;
    else
        cout << "*The given matrix is a Sparse Matrix" << endl;

    /***************************CLOSING FILES***********************************/
    inFile.close();
    //checking file closes or not
    if (inFile.fail())
    {
        cerr << "Error in closing file 2\n";
        exit(102);
    }
    /*************************************************************************/
}
```

```
//method to convert the sparse matrix into 2D array
void sparseMatrix::sparseToArray()
{
    int x = 1;

    a = new int *[3];
    for (int i = 0; i < 3; i++)
    {
        a[i] = new int[nz + 1];
    }

    //storing no. of rows, cols and non-zero elements in the 1st entry of the array rows
    a[0][0] = r;
    a[1][0] = c;
    a[2][0] = nz;

    for (int i = 0; i < r; i++)
    {
        for (int j = 0; j < c; j++)
        {
            if (m[i][j] != 0)
            {
                a[0][x] = i;
                a[1][x] = j;
                a[2][x] = m[i][j];
                x++;
            }
        }
    }
    //Printing the Array which is created from the Sparse Matrix
    cout << "\n***| Array created from Sparse Matrix: " << endl;
    for (int i = 0; i < 3; i++)
    {
        if (i == 0)
            cout << "Rows -> " << a[0][0] << " | ";
        else if (i == 1)
            cout << "Cols -> " << a[1][0] << " | ";
        else
            cout << "Vals -> " << a[2][0] << " | ";

        for (int j = 1; j < nz + 1; j++)
        {
            cout << a[i][j] << "\t";
        }
        cout << endl;
    }
}

//converts the array into the Sparse matrix
void sparseMatrix::arrayToSparse()
{
    int rows = a[0][0];
    int cols = a[1][0];
    int nonZ = a[2][0];

    //creating a matrix with dynamically allocating memory
    spaMat = new int *[rows];
    for (int i = 0; i < rows; i++)
    {
        spaMat[i] = new int[cols];
    }

    //filling the matrix with zeroes
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            spaMat[i][j] = 0;

    //filling the non-zero elements in the matrix
    for (int i = 1; i < nonZ + 1; i++)
    {
        spaMat[a[0][i]][a[1][i]] = a[2][i];
    }

    //Printing the Sparse Matrix which is created from the Array
    cout << "\n***Sparse Matrix created from Array: " << endl;
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
            cout << spaMat[i][j] << "\t";
        cout << endl;
    }
}

//method to convert Sparse Matrix into Linked list
void sparseMatrix::sparseToLL()
{
    //creating head node
    head = new node(r, c, nz);
    node *copy = head;
    count++;
```

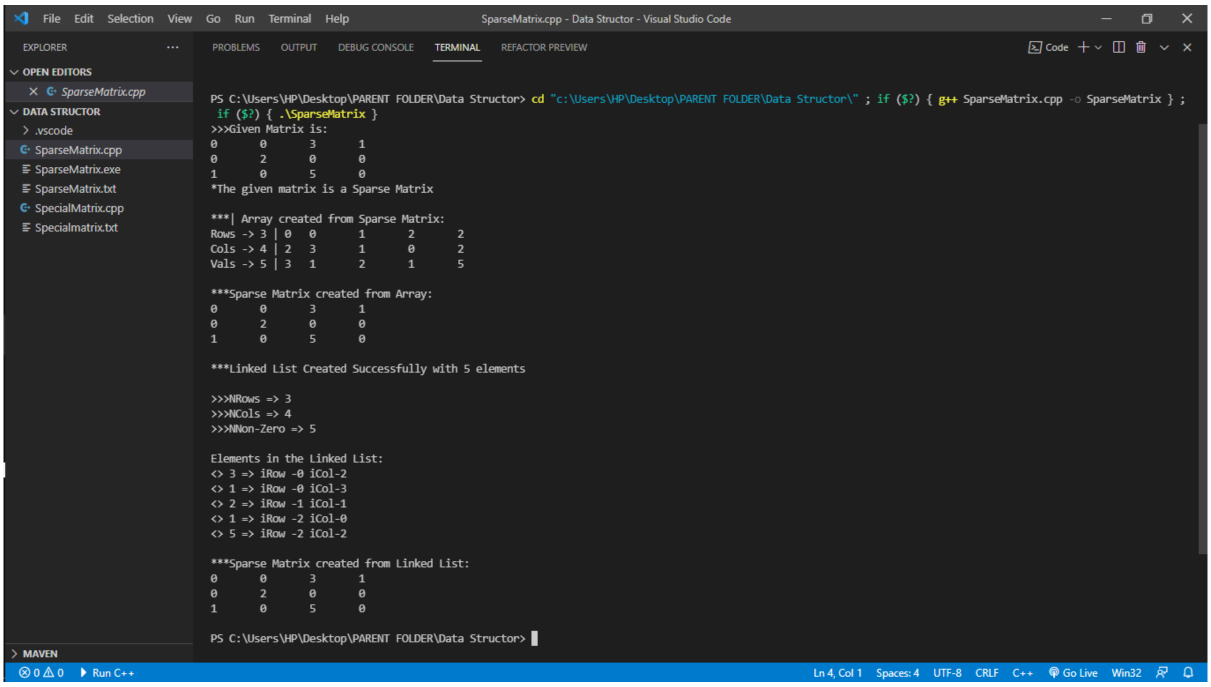
```
//creating nodes for non-zero elements and adding into linked list
for (int i = 0; i < r; i++)
{
    for (int j = 0; j < c; j++)
    {
        if (m[i][j] != 0)
        {
            node *tmp = new node(i, j, m[i][j]);
            copy->next = tmp;
            copy = tmp;
            count++;
        }
    }
}
cout << "\n***Linked List Created Successfully with " << count - 1 << " elements" << endl;

//printing the nodes of the linked list after creating
node *t = head;
while (t != NULL)
{
    if (t == head)
    {
        cout << "\n>>>NRows => " << t->row << endl;
        cout << ">>>NCols => " << t->col << endl;
        cout << ">>>NNon-Zero => " << t->value << endl;
        cout << "\nElements in the Linked List: " << endl;
    }
    else
    {
        cout << "<> " << t->value << " => iRow -" << t->row << " iCol-" << t->col << endl;
    }
```

```
        t = t->next;
    }
}

//method to create linked list to Sparse Matrix
void sparseMatrix::llToSparse()
{
    int rows = head->row;
    int cols = head->col;
    int nonZ = head->value;

    //creating matrix by dynamically allocating memory
    spaMat = new int *[rows];
    for (int i = 0; i < rows; i++)
    {
        spaMat[i] = new int[cols];
    }

    //filling he matrix with zeroes
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            spaMat[i][j] = 0;

    //filling non-zero elements in the matrix
    node *t = head->next;
    while (t != NULL)
    {
        spaMat[t->row][t->col] = t->value;
        t = t->next;
    }
    //Printing the Sparse Matrix which is created from the Linked List
    cout << "\n***Sparse Matrix created from Linked List: " << endl;
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < cols; j++)
            cout << spaMat[i][j] << "\t";
        cout << endl;
    }
    cout << endl;
}

int main()
{
    sparseMatrix obj;

    obj.input();
    obj.sparseToArray();
    obj.arrayToSparse();
    obj.sparseToLL();
    obj.llToSparse();

    return 0;
}
```

SparseMatrix.txt

```
3 4
0 0 3 1
0 2 0 0
1 0 5 0
```

Output



```
PS C:\Users\HP\Desktop\PARENT FOLDER\Data Structor> cd "c:\Users\HP\Desktop\PARENT FOLDER\Data Structor\" ; if ($?) { g++ SparseMatrix.cpp -o SparseMatrix } ;
 if ($?) { .\SparseMatrix }
>>>Given Matrix is:
0       0       3       1
0       2       0       0
1       0       5       0
*The given matrix is a Sparse Matrix

***| Array created from Sparse Matrix:
Rows -> 3 | 0   0       1       2       2
Cols -> 4 | 2   3       1       0       2
Vals -> 5 | 3   1       2       1       5

***Sparse Matrix created from Array:
0       0       3       1
0       2       0       0
1       0       5       0

***Linked List Created Successfully with 5 elements

>>>NRows => 3
>>>NCols => 4
>>>NNon-Zero => 5

Elements in the Linked List:
<> 3 => iRow -0 iCol-2
<> 1 => iRow -0 iCol-3
<> 2 => iRow -1 iCol-1
<> 1 => iRow -2 iCol-0
<> 5 => iRow -2 iCol-2

***Sparse Matrix created from Linked List:
0       0       3       1
0       2       0       0
1       0       5       0

PS C:\Users\HP\Desktop\PARENT FOLDER\Data Structor>
```