**Institute of Computer Technology**

**B. Tech. Computer Science and Engineering**

**Sub: DS**

**Course Code: 2CSE302**

**Practical – 12**

**Name: Jaymin Gondaliya**

**Enrollment No: 23162171007**

**Sem - 3**

**Branch: CS**

**Class: A**

**Batch: 32**

**Problem Definition-1:** You are given two integers m and n, which represent the dimensions of a matrix. You are also given the head of a linked list of integers. Generate an m x n matrix that contains the integers in the linked list presented in spiral order (clockwise), starting from the top-left of the matrix. If there are remaining empty spaces, fill them with -1. In return you need to print the matrix as shown in the example.

**Code:**

*#include* <stdio.h>

*#include* <stdlib.h>

*//* **Structure for a linked list node**

struct Node {

    int data;

    struct Node\* next;

};

*//* **Function to create a new linked list node**

struct Node\* newNode(int *data*) {

    struct Node\* node = (struct Node\*)malloc(sizeof(struct Node));

    node->data = *data*;

    node->next = NULL;

*return* node;

}

*//* **Function to create a linked list with a given number of elements**

struct Node\* createLinkedList(int *num*) {

    struct Node\* head = NULL;

    struct Node\* temp = NULL;

*for* (int i = 1; i <= *num*; i++) {

        struct Node\* new\_node = newNode(i);

*if* (head == NULL) {

            head = new\_node;

            temp = head;

        } *else* {

            temp->next = new\_node;

            temp = temp->next;

        }

    }

*return* head;

}

*//* **Function to fill the matrix in spiral order**

void fillSpiral(int *m*, int *n*, struct Node\* *head*) {

    int mat[*m*][*n*];

    int top = 0, bottom = *m* - 1, left = 0, right = *n* - 1;

    struct Node\* curr = *head*;

*while* (top <= bottom && left <= right) {

*//* **Fill top row**

*for* (int i = left; i <= right; i++) {

*if* (curr != NULL) {

                mat[top][i] = curr->data;

                curr = curr->next;

            } *else* {

                mat[top][i] = -1;

            }

        }

        top++;

*//* **Fill right column**

*for* (int i = top; i <= bottom; i++) {

*if* (curr != NULL) {

                mat[i][right] = curr->data;

                curr = curr->next;

            } *else* {

                mat[i][right] = -1;

            }

        }

        right--;

*//* **Fill bottom row**

*if* (top <= bottom) {

*for* (int i = right; i >= left; i--) {

*if* (curr != NULL) {

                    mat[bottom][i] = curr->data;

                    curr = curr->next;

                } *else* {

                    mat[bottom][i] = -1;

                }

            }

            bottom--;

        }

*//* **Fill left column**

*if* (left <= right) {

*for* (int i = bottom; i >= top; i--) {

*if* (curr != NULL) {

                    mat[i][left] = curr->data;

                    curr = curr->next;

                } *else* {

                    mat[i][left] = -1;

                }

            }

            left++;

        }

    }

*//* **Print the matrix**

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

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

            printf("%d ", mat[i][j]);

        }

        printf("\n");

    }

}

int main() {

    int m = 4, n = 3;

    int numElements = 13;*//* **Number of elements in the linked list**

*//* **Create a linked list with 'numElements' nodes**

    struct Node\* head = createLinkedList(numElements);

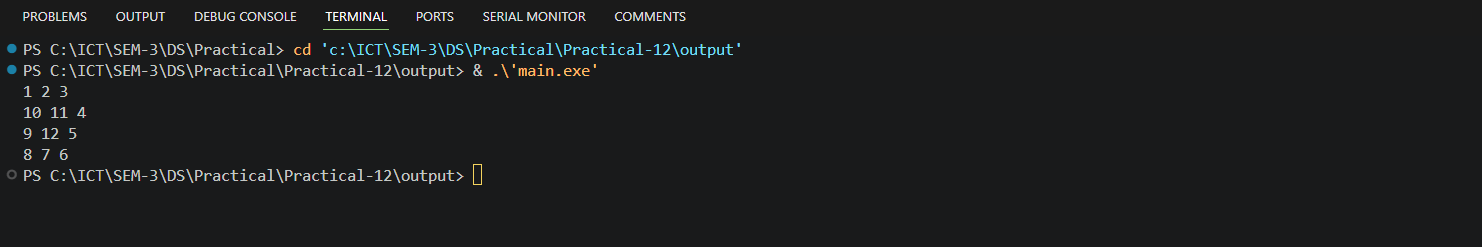
*//* **Fill the matrix and print it**

    fillSpiral(m, n, head);

*return* 0;

}

**Output:**

****