**Linked List Implementation**

**Sajid Iqbal Ishaq Mohammed**

Concordia University

November 14, 2022

CSC- 543

**Self-Assessment**

**1**. It took me just under a week to complete the assignment.

**2**. I put my entire effort into coding the programs. In this assignment, I will indeed expect an A Grade (100 on 100).

**3**. Every coding solution is correct. As a result, I would anticipate an A grade.

**4**. Learning the fundamentals of C++ produced coding simple. The main issue I encountered was in running the code. As I complete all of the weekly assignments, I am becoming more precise in detecting the errors and executing the program. The overall experience was excellent and will be best.

**Linked List Implementation**

**A LinkedList Implementation for Sparse Matrices as described above:**

#include <iostream>

#include <cstdlib>

#include <cmath>

using namespace std;

**// A structure to represent a node in the Linked List**

struct Node

{

int row, col, val;

struct Node\* next;

};

**// A structure to represent a sparse matrix**

struct SparseMatrix

{

int m, n, num; // Dimensions of the sparse matrix

struct Node\* head; // Linked list of non-zero elements

};

**// Function to create a new Node in the Linked List**

struct Node\* newNode(int row, int col, int val)

{

struct Node\* temp = new Node;

temp->row = row;

temp->col = col;

temp->val = val;

temp->next = NULL;

return temp;

}

**/ Function to create a Sparse Matrix of size m x n**

struct SparseMatrix createSparseMatrix(int m, int n)

{

struct SparseMatrix sparseMatrix;

sparseMatrix.m = m;

sparseMatrix.n = n;

sparseMatrix.num = 0;

sparseMatrix.head = NULL;

return sparseMatrix;

}

**// Function to insert a new element in the Sparse Matrix**

void insert(struct SparseMatrix\* sparseMatrix, int row, int col, int val)

{

**// Check if the element is already present**

if (val != 0)

{

struct Node\* temp, \* last;

temp = sparseMatrix->head;

**// Go to the end of the linked list**

while (temp != NULL)

{

if (temp->row == row && temp->col == col)

{

**// Add the new element here**

temp->val = val;

return;

}

else

{

last = temp;

temp = temp->next;

}

}

**Final answer**

**// If row number is greater, insert from list 2**

else if (temp1->row > temp2->row)

{

insert(&sparseMatrix3, temp2->row, temp2->col, temp2->val);

temp2 = temp2->next;

}

**// If column number is less, insert from list 1**

else if (temp1->col < temp2->col)

{

insert(&sparseMatrix3, temp1->row, temp1->col, temp1->val);

temp1 = temp1->next;

}

**// If column number is greater, insert from list 2**

else if (temp1->col > temp2->col)

{

insert(&sparseMatrix3, temp2->row, temp2->col, temp2->val);

temp2 = temp2->next;

}

}

**// Insert the remaining elements of list 1**

while (temp1 != NULL)

{

insert(&sparseMatrix3, temp1->row, temp1->col, temp1->val);

temp1 = temp1->next;

}

**// Insert the remaining elements of list 2**

while (temp2 != NULL)

{

insert(&sparseMatrix3, temp2->row, temp2->col, temp2->val);

temp2 = temp2->next;

}

return sparseMatrix3;

}

**// Driver program to test above functions**

int main()

{

**// Creating two sparse matrices**

struct SparseMatrix sparseMatrix1, sparseMatrix2, sparseMatrix3;

sparseMatrix1 = createSparseMatrix(5, 5);

sparseMatrix2 = createSparseMatrix(5, 5);

**// Adding elements in first sparse matrix**

insert(&sparseMatrix1, 0, 0, 8);

insert(&sparseMatrix1, 0, 3, 9);

insert(&sparseMatrix1, 1, 1, 10);

insert(&sparseMatrix1, 1, 2, 11);

insert(&sparseMatrix1, 2, 2, 12);

**// Adding elements in second sparse matrix**

insert(&sparseMatrix2, 0, 0, 8);

insert(&sparseMatrix2, 0, 3, 9);

insert(&sparseMatrix2, 1, 1, 10);

insert(&sparseMatrix2, 1, 2, 11);

insert(&sparseMatrix2, 2, 2, 12);

**// Adding the two sparse matrices**

sparseMatrix3 = add(sparseMatrix1, sparseMatrix2);

**// Displaying sum of sparse matrices**

cout << "Sparse Matrix 1" << endl;

display(sparseMatrix1);

cout << "\nSparse Matrix 2" << endl;

display(sparseMatrix2);

cout << "\nSum of two sparse matrices" << endl;

display(sparseMatrix3);

return 0;

}

**Link to my GitHub repository:**