Assignment 2(Sparse matrix):

Cpp code

Naive approach:

```cpp
#include <iostream>
#include <vector>

using namespace std;
void solution1(int sparse[4][4])
{
    vector<vector<int>> mat;
    int k = 0;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            if (sparse[i][j] != 0)
            {
                vector<int> v1;
                v1.push_back(i);
                v1.push_back(j);
                v1.push_back(sparse[i][j]);
                mat.push_back(v1);
            }
        }
    }

    cout<<"row"<<" column"<<"   value"<<"\n";
    for (auto i : mat)
    {
        for (auto j : i)
        {
            cout << j << "\t";
        }
        cout << "\n";
    }
}

void solution2(int sparse[4][4])
{
    int size = 0;
    for (int i = 0; i < 4; i++)
    {
```

```
        for (int j = 0; j < 4; j++)
        {
            if (sparse[i][j] != 0)
            {
                size++;
            }
        }
    }
    int mat[3][size];
    int k = 0;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            if (sparse[i][j] != 0)
            {
                mat[0][k] = i;
                mat[1][k] = j;
                mat[2][k] = sparse[i][j];
                k++;
            }
        }
    }
    string arr[3] = {"row", "column", "value"};
    for (int i = 0; i < 3; i++)
    {
        cout << arr[i] << " ";
        for (int j = 0; j < size; j++)
        {
            cout << mat[i][j] << " ";
        }
        cout << "\n";
    }
}

int main()
{
    int sparse[4][4] = {
        {0, 0, 3, 4},
        {5, 0, 0, 2},
        {0, 0, 0, 0},
        {1, 0, 3, 0}};
    cout<<"solution 1 using vector:\n";
```

```
    solution1(sparse);
    cout<<"\nsoultion 2 using array:\n";
    solution2(sparse);
    return 0;
}
```

2 pointers:

```
#include <iostream>
using namespace std;

class Node
{
public:
    int row;
    int column;
    int data;
    Node *next;
};

void newNode(Node **p, int i, int j, int value)
{
    Node *temp = *p;
    Node *r;

    if (temp == NULL)
    {
        temp = new Node();
        temp->row = i;
        temp->column = j;
        temp->data = value;
        temp->next = NULL;
        *p = temp;
    }
    else
    {
        while (temp->next != NULL)
        {
            temp = temp->next;
        }
        r = new Node();
        r->row = i;
        r->column = j;
```

```
        r->data = value;
        r->next = NULL;
        temp->next = r;
    }
}

void PrintLinkedList(Node **start)
{
    Node *ptr1 = *start;
    std::cout << "row  "
            << "column  "
            << "data  "
            << "\n";
    while (ptr1 != NULL)
    {
        std::cout << ptr1->row << "\t" << ptr1->column << "\t" <<
ptr1->data << "\n";
        ptr1 = ptr1->next;
    }
}

int main()
{

    int sparseMatrix[4][5] = {
        {0, 0, 3, 0, 4},
        {0, 0, 5, 7, 0},
        {0, 0, 0, 0, 0},
        {0, 2, 6, 0, 0}};

    Node *firstNode = NULL;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            if (sparseMatrix[i][j] != 0)
            {
                newNode(&firstNode, i, j, sparseMatrix[i][j]);
            }
        }
    }
    PrintLinkedList(&firstNode);
    return 0;
```

```
}
```

- Test cases passed
- Completed on 15/3/23

Q/A:

1. How long did you spend on this assignment?
    a. 1day
2. Based on your effort, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 10/10.
3. Based on your solution, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 9/10.
4. Provide a summary of what doesn't work in your solution, along with an explanation of how you attempted to solve the problem and where you feel you struggled?
    a. The sparse matrix array technique has three rows, the first row being the row values, the second being the column values, and the third being the value. By viewing all non-zero values and storing them in a 3*m matrix
    b. It is a Single linked list representation by storing non-zero values in a node with a row, a column, some data, and the next node values