Міністерство освіти і науки України
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Кафедра програмування

ЛАБОРАТОРНА РОБОТА № 6

Розріджена матриця

з курсу “Алгоритми та структури даних”

Виконав:

Студент групи ПМІ-16

Бевз Маркіян Юрійович

Львів – 2024

Мета: Ознайомлення зі структурою даних розрідженої матриці та реалізація основних операцій над нею.

Принцип роботи розрідженої матриці:

1. Кожен ненульовий елемент матриці буде представлений об'єктом структури **Node**, що містить інформацію про рядок (**row**), стовпець (**col**) та значення (**value**) елемента. Кожен вузол **Node** також містить посилання на наступний вузол у списку.
2. Всі нульові елементи матриці не зберігаються, тому що їх велика кількість у розріджених матрицях може займати багато пам'яті.

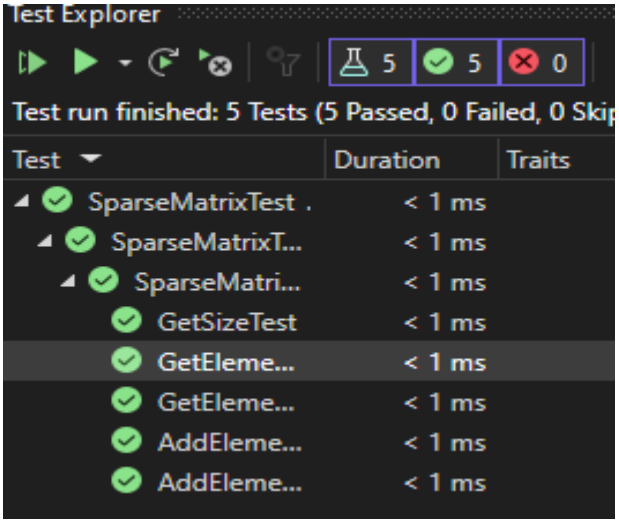
Опис структури даних:

- **Node**: Представляє вузол розрідженої матриці. Містить поля **row**, **col** і **value** для зберігання інформації про рядок, стовпець і значення елемента відповідно, а також посилання **next** на наступний вузол.
- **SparseMatrix**: Головний клас для роботи з розрідженою матрицею. Містить поля **rows** і **columns** для зберігання розмірів матриці, **capacity** для зберігання кількості ненульових елементів, і посилання **head** на перший елемент списку.

Опис операцій:

1. **SparseMatrix(int r, int c)**: Конструктор класу **SparseMatrix**, що ініціалізує розміри матриці і встановлює початкове значення полів.
2. **~SparseMatrix()**: Деструктор для класу **SparseMatrix**, який звільняє пам'ять, що виділена під вузли матриці.
3. **getSize() const**: Повертає загальну кількість елементів у матриці.
4. **addElement(int row, int col, int value)**: Додає новий елемент до матриці з вказаним значенням, рядком і стовпцем.
5. **print() const**: Виводить розріджену матрицю на екран. Якщо кількість ненульових елементів більше 70% від загальної кількості елементів, виводиться повна матриця; інакше виводяться тільки ненульові елементи.
6. **getElement(int row, int column) const**: Повертає значення елемента матриці за вказаними рядком і стовпцем.
7. **multiplyByScalar(const SparseMatrix& matrix, int scalar)**: Функція, що множить розріджену матрицю на скаляр.

Хід роботи: Після вивчення принципу роботи розрідженої матриці я реалізував усі необхідні методи й сам клас. Також було створено 5 тестів для перевірки правильності роботи цих методів. Нижче буде прикріплено результат виконання програми, тестів, та їх код.




```
TEST_CLASS(SparseMatrixTest)
{
public:

    TEST_METHOD(AddElementTest)
    {
        SparseMatrix matrix(3, 3);
        matrix.addElement(0, 0, 1);
        matrix.addElement(1, 1, 2);
        matrix.addElement(2, 2, 3);

        Assert::AreEqual(matrix.getElement(0, 0), 1);
        Assert::AreEqual(matrix.getElement(1, 1), 2);
        Assert::AreEqual(matrix.getElement(2, 2), 3);
        Assert::AreEqual(matrix.getElement(0, 1), 0);
    }
    TEST_METHOD(AddElementTest2)
    {
        SparseMatrix matrix(3, 3);

        matrix.addElement(1, 1, 5);
        matrix.addElement(2, 2, 10);

        Assert::AreEqual(matrix.getElement(1, 1), 5);
        Assert::AreEqual(matrix.getElement(2, 2), 10);
        Assert::AreEqual(matrix.getElement(0, 0), 0);
    }
```

```
TEST_METHOD(GetElementTest)
{
    SparseMatrix matrix(3, 3);
    matrix.addElement(0, 0, 1);
    matrix.addElement(1, 1, 2);
    matrix.addElement(2, 2, 3);

    Assert::AreEqual(matrix.getElement(0, 0), 1);
    Assert::AreEqual(matrix.getElement(1, 1), 2);
    Assert::AreEqual(matrix.getElement(2, 2), 3);
    Assert::AreEqual(matrix.getElement(0, 1), 0);
}
TEST_METHOD(GetElementTest1)
{
    // Arrange
    SparseMatrix matrix(3, 3);
    matrix.addElement(1, 1, 5);

    // Act
    int element = matrix.getElement(1, 1);

    // Assert
    Assert::AreEqual(element, 5);
}
TEST_METHOD(GetSizeTest)
{
    SparseMatrix matrix(3, 4);
    Assert::AreEqual(matrix.getSize(), 12);

    matrix.addElement(0, 0, 1);
    Assert::AreEqual(matrix.getSize(), 12);
}
```

```
Enter the number of rows: 3
Enter the number of columns: 3
Enter row, column, and value (or 'stop' to end):  12 12 2
Error: Attempting to add element outside matrix boundaries.
Enter row, column, and value (or 'stop' to end): 0 0 1
Enter row, column, and value (or 'stop' to end): 0 1 3
Enter row, column, and value (or 'stop' to end): 1 0 1
Enter row, column, and value (or 'stop' to end): 1 2 3
Enter row, column, and value (or 'stop' to end): 2 1 2
Enter row, column, and value (or 'stop' to end): 2 0 1
Enter row, column, and value (or 'stop' to end): 1 1 4
Enter row, column, and value (or 'stop' to end): stop
1 3 0
1 4 3
1 2 0
Enter multiplyer for your matrix:6
6 18 0
6 24 18
6 12 0
```

```
Enter the number of rows: 300
Enter the number of columns: 500
Enter row, column, and value (or 'stop' to end): 120 120 12
Enter row, column, and value (or 'stop' to end): 200 300 40
Enter row, column, and value (or 'stop' to end): 1 12 12
Enter row, column, and value (or 'stop' to end): 2 2 3
Enter row, column, and value (or 'stop' to end): 1 0 2
Enter row, column, and value (or 'stop' to end): stop
Row: 1 Col: 0 Val: 2
Row: 1 Col: 12 Val: 12
Row: 2 Col: 2 Val: 3
Row: 120 Col: 120 Val: 12
Row: 200 Col: 300 Val: 40
Enter multiplyer for your matrix: 7
Row: 1 Col: 0 Val: 14
Row: 1 Col: 12 Val: 84
Row: 2 Col: 2 Val: 21
Row: 120 Col: 120 Val: 84
Row: 200 Col: 300 Val: 280
```

Висновок: У ході виконання лабораторної роботи було успішно реалізовано структуру даних розрідженої матриці та основні операції над нею. Робота з розрідженими матрицями є важливою для оптимізації пам'яті та швидкості обчислень у великих чисельних задачах.