**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

# ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3 «Обработка разреженных матриц»

Студент Романов Семен Константинович

Группа ИУ7 – 35Б

## ***Описание задачи***

Разреженная (содержащая много нулей) матрица хранится в форме 3-х объектов:

- вектор A содержит значения ненулевых элементов;
- вектор IA содержит номера строк для элементов вектора A;
- связный список JA, в элементе Nk которого находится номер компонентв A и IA, с которых начинается описание столбца Nk матрицы A.

1. Смоделировать операцию сложения двух матриц, хранящихся в этой форме, с получением результата в той же форме.
2. Произвести операцию сложения, применяя стандартный алгоритм работы с матрицами.
3. Сравнить время выполнения операций и объем памяти при использовании этих 2-х алгоритмов при различном проценте заполнения матриц.

# Техническое задание:

### Входные данные:

1. **Целое число(номер комнады):** целое число от 0 до 8 (см. Функции программы)
2. **Числовые обозначения:** количество столбцов и строчек и значения этих элементов

### Выходные данные:

1. **Исходные и результирующие матрицы:** В исходном и разряженном виде.
2. **Сравнение способов сложения:** Затраченное время и память

**Способ обращения к программе:** запускается из терминала командой ./app.exe

## Функции программы:

1. Ввести матрицу с помощью файла
2. Сгенерировать случайную матрицу
3. Ввести матрицу с клавиатуры
4. Показать матрицу в стандартном виде
5. Привести матрицу к разряженному виду
6. Показать матрицу в разряженном виде
7. Просуммировать матрицы
8. Просуммировать матрицы в разряженном виде
0. Выход

## Аварийные ситуации:

1. Неверный тип данных при вводе числовых значений: Вывод сообщения “Invalid Input”
2. Ошибка при вводе файла: Вывод сообщения “Invalid Input”
3. Ошибка в матрице в файле: Вывод сообщения “Icorrect datafile”
4. Ошибка при разных размерах матрицы: Вывод сообщения “Incorrect matrix size”
5. Ошибка при вводе, когда размер не нулевых элементов больше размера матрицы: Вывод сообщения “Invalid Input”

## Структуры данных:

```
typedef struct
{
    int **matrix;
    int rows;
    int columns;
    short allocated;
} matrix;
```

1. int **matrix – массив указателей на строки матрицы
2. int rows – количество строк в матрице
3. int columns - количество столбцов в матрице
4. short allocated – флаг, говорящий о том, были ли присвоены данной структуре значения

```
typedef struct
{
    int *elements;
    int *index_row;
    int *cols_en;
    int e_amount;
    int columns;
    short allocated;
} parse;
```

1. Int *elements – массив с ненулевыми элементами матрицы, заполняемые в ходе обхода матрицы по столбцам
2. Int *index_row – массив с номерами строк для соответствующих значений elements
3. Int cols_en – массив, каждый элемент которого указывает на индекс элемента из elements, которого начинается соответствующий элемент

4. Int e_amount – количество элементов в elements и index_row
5. Int columns – количество столбцов в матрице
6. short allocated – флаг, говорящий о том, были ли присвоены данной структуре значения

## Алгоритм:

1. На вход подается команда от 0 до 9
2. При вводе матрицы, оная хранится лишь в стандартном виде, для того, чтобы провести сложение в разложенном виде, то необходимо вызвать необходимую команду, чтобы их создать(разложение происходит по столбцам)
3. При стандартном сложении матриц, выполняется сложение элемента к элементу
4. В разложенном виде, действия проводятся следующим образом:
    a. Сравнивается каждый столбец как массив построчных вхождений
    b. Если нет повторяющихся элементов, то они записываются по возрастанию
    c. Если совпадения есть, то элементы складываются и записываются
5. Пользователь выполняет действия с таблицей, пока не введет 0 (Выход)

| N | Test | Input | Output |
|---|---|---|---|
| 1 | Некорректная комманда | 11 | “Unknown command” |
| 2 | Несуществующий файл | Abc.txt(файл не существует) | “Invalid Input” |
| 3 | Некорректное количество строк | -1 | “Invalid input” |

| | | | |
|---|---|---|---|
| 4 | Некорректная позиция числа в матрице | (При матрице 5x5): 6 3 10 | “Invalid Input” |
| 5 | Подсчет суммы матриц без их инициализации | 7(матрицы не инициализированы) | “Matrixes not allocated” |
| 6 | Некорректный ввод процента заполненности | 101 | “Invalid input” |
| 7 | Некорректный ввод матрицы | 4 e 6<br>5 2 7 | “Invalid Input” |

## Оценка эффективности:

## Время сложения:

| | | | |
|---|---|---|---|
| 10% заполнения | Size | Standart | Sparse |
| | 10x10 | 5 | 19 |
| | 50x50 | 78 | 68 |
| | 500x500 | 4816 | 20731 |
| 20% заполнения | 10x10 | 5 | 26 |
| | 50x50 | 82 | 162 |
| | 500x500 | 4816 | 67566 |
| 50% заполнения | 10x10 | 7 | 37 |
| | 50x50 | 78 | 695 |
| | 500x500 | 4816 | 232358 |
| 75% заполнения | 10x10 | 8 | 34 |
| | 50x50 | 88 | 1428 |
| | 500x500 | 5169 | 519257 |
| 100% заполнения | 10x10 | 14 | 48 |
| | 50x50 | 96 | 2155 |
| | 500x500 | 5448 | 912292 |

**Занимаемая память:**

| | | | |
|---|---|---|---|
| 10% заполнения | Size | Standart | Sparse |
| | 10x10 | 1344 | 520 |
| | 50x50 | 30144 | 6760 |
| | 500x500 | 3000144 | 606160 |
| 20% заполнения | 10x10 | 1344 | 760 |
| | 50x50 | 30144 | 12760 |
| | 500x500 | 3000144 | 1206160 |
| 50% заполнения | 10x10 | 1344 | 1480 |
| | 50x50 | 30144 | 30760 |
| | 500x500 | 3000144 | 3006160 |
| 75% заполнения | 10x10 | 1344 | 2080 |
| | 50x50 | 30144 | 45760 |
| | 500x500 | 3000144 | 4506160 |
| 100% заполнения | 10x10 | 1344 | 2680 |
| | 50x50 | 30144 | 60760 |
| | 500x500 | 3000144 | 6006160 |

**Контрольные вопросы:**

1. **Что такое разреженная матрица, какие способы хранения вы знаете?**

Разреженная матрица – это матрица, содержащая большое количество нулей. Способы хранения – строчный формат, столбцевой формат, линейный связанный список, связная схема хранения, кольцевой связной список, двунаправленые стеки.

2. **Каким образом и сколько памяти выделяется под хранение разреженной и обычной матрицы?**

Под обычную матрицу выделяется N * M ячеек памяти, где N и M – строки и столбцы. Для разреженного это 3 * P ячеек, где P – количество ненулевых элементов.

### 3. Каков принцип обработки разреженной матрицы?

Принцип работы с разреженными матрицами таков:
Необходимые вычисления проводятся только с ненулевыми элементами, количество операций же будет пропорционально количеству ненулевых элементов.

### 4. В каком случае для матриц эффективнее применить стандартные алгоритмы обработки матриц? Отчего это зависит?

Стандартные алгоритмы применять целесообразнее при заполненности матрицы от 40% - 50%, поскольку количество занимаемой памяти стремится к стандартному представления, и в какой-то момент её даже превысит. Также в случае если важна скорость обработки, то следует воспользоваться стандартным сложением матриц

## Вывод

При больших размерах матриц и малой её заполненности, эффективность использования разреженных матриц видна невооруженным взглядом: Значительный выигрыш по памяти при относительно небольшом проигрыше по времени. Однако при заполненности от ~40% становится ясно, что способ представления в столбцевом формате является неэффективным,

поскольку не дает значительного выигрыша по памяти, при этом крайне сильно страдает производительность, время работы порой превышает стандартный способ в 15 раз. Также если нам важна производительность, а не количество занимаемой памяти, то следует использовать стандартное представление.