

ФАКУЛЬТЕТ: Информатика и системы управления

КАФЕДРА: Программное обеспечение ЭВМ и информационные технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

**«Обработка разреженных матриц»**

Студент: Зайцева Алена Андреевна

Группа ИУ7 – 32Б

*2020 г.*

Оглавление

Оглавление

[1. Описание условия задачи 3](#_Toc53602516)

[2. Описание ТЗ 4](#_Toc53602517)

[1) Описание исходных данных: 4](#_Toc53602518)

[2) Описание результатов: 4](#_Toc53602519)

[3) Описание задачи, реализуемой программой: 5](#_Toc53602520)

[4) Способ обращения к программе: 5](#_Toc53602521)

[5) Описание возможных аварийных ситуаций и ошибок пользователя. 5](#_Toc53602522)

[3. Описание внутренних структур данных. 6](#_Toc53602523)

[4. Описание алгоритма 6](#_Toc53602524)

[5. Набор тестов с указанием, что проверяется 6](#_Toc53602525)

[1) Тесты на вводимые значения 7](#_Toc53602526)

[2) Проверка заполнения матрицы в ручном режиме 8](#_Toc53602527)

[3) Проверка заполнения матрицы в автоматическом режиме 9](#_Toc53602528)

[4) Проверка сложения матриц 9](#_Toc53602529)

[6. Оценка эффективности 10](#_Toc53602530)

[7. Выводы по проделанной работе 14](#_Toc53602531)

[8. Ответы на вопросы 14](#_Toc53602532)

# Описание условия задачи

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

* вектор A содержит значения ненулевых элементов;
* вектор JA содержит номера столбцов для элементов вектора A;
* связный список IA, в элементе Nk которого находится номер компонент в A и JA, с которых начинается описание строки Nk матрицы A. (значение -1 означает, что строка Nk пуста)

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 различной степени разреженности матриц и различной размерности матриц.

* Матрицы хранятся и выводятся в форме трех объектов. Для небольших матриц можно дополнительно вывести матрицу в виде матрицы.

При разработке интерфейса программы следует предусмотреть:

* указание операции, производимой программой,
* указание формата и диапазона вводимых данных,
* указание формата выводимых данных
* наличие пояснений при выводе результата,
* возможность заполнения разреженных матриц вручную (даже при большой размерности, например, 1000\*1000) и автоматически с разным процентом разреженности.

При тестировании программы необходимо:

* проверить правильность ввода o проконтролировать правильность вывода данных (т.е. их соответствие требуемому формату);
* проверить правильность выполнения операций;
* обеспечить вывод сообщений при отсутствии входных данных («пустой ввод») или вывод нулевого результата при ненулевом входе;
* сравнить время выполнения стандартного алгоритма обработки матриц и алгоритма обработки разреженных матриц при различной заполненности матриц.
* сравнить объем требуемой памяти для реализации стандартного алгоритма обработки матриц и алгоритма обработки разреженных матриц при различном проценте заполнения матриц и при различном их размере.
* Необходимо тщательно следить за освобождением динамической памяти (если она используется) при окончании программы.

# Описание ТЗ

Внешняя спецификация:

## Описание исходных данных:

* n, m – количество строк и столбцов в обеих складываемых матрицах, соответственно.

(Ограничения: снизу – n, m >0, сверху – зависит от компьютера (память выделяется динамически))

* amount1, amount2 – количество ненулевых элементов в первой и второй матрицах, соответственно.

(Ограничения: 0<= amount1, amount<=n\*m)

* m1, m2 – первая и вторая слагаемые матрицы

(содержат действительные числа)

Матрицу можно ввести в автоматическом режиме (0) – она случайным образом заполнится необходимым количеством случайных чисел, или в ручном режиме (1) в координатном формате:

“i j value” - где

i – номер строки (0<=i<n, индексация начинается с нуля),

j – номер столбца (0<=j<m, индексация начинается с нуля),

value – значение элемента (неравное нулю с точностью EPS) -

по одному в строке.

Примечание: вычисления ведутся с точностью EPS = 1e-6

## Описание результатов:

1. Две исходные матрицы и матрица-результат сложения исходных матриц в форме 3-х объектов:

* вектор A содержит значения ненулевых элементов;
* вектор JA содержит номера столбцов для элементов вектора A;
* связный список IA, в элементе Nk которого находится номер компонент в A и JA, с которых начинается описание строки Nk матрицы A.

Если размерности матриц небольшие (n <= 20, m <= 20), то все три матрицы будут дополнительно выведены в классическом представлении

1. Сравнительная таблица, которая показывает отношение необходимых времени и памяти при использовании стандартной и описанной выше (далее – “разреженной”) формах хранения матриц для реализации операции сложения исходных матриц.

## Описание задачи, реализуемой программой:

Программа реализует операцию сложения двух матриц одинаковой размерности двумя способами (при хранении матриц в разреженной и классической формах) и сравнивает необходимое количество времени и памяти для данной операции при использовании каждого из подходов.

## Способ обращения к программе:

Программа запускается из папки, содержащей исполняемый файл, через терминал.

## Описание возможных аварийных ситуаций и ошибок пользователя.

Возможные ошибки пользователя:

1. Ввод нечислового значения, когда программа ожидает число.
2. Ввод нецелого значения, когда программа ожидает целое.
3. Ввод некорректного количества строк в матрицах (n<1).
4. Ввод некорректного количества столбцов в матрицах (m<1).
5. Ввод некорректного количества ненулевых элементов матрицы (amount< 0 или amount >n\*m).
6. Ввод некорректного номера строки (i<0 или i>=n)
7. Ввод некорректного номера столбца (j<0 или j>=m)
8. Ввод некорректного значения ненулевого элемента матрицы (value = 0 с точностью EPS).
9. Ввод нескольких ненулевых значений на одну позицию в матрице.

Возможные аварийные ситуации:

1. при вводе числового поля пользователь ничего не вводит – программа будет ждать ввода и не завершится, пока не встретит хоть один непробельный символ.
2. Компьютер не сможет выделить необходимое количество памяти – программа выведет сообщение об ошибке и завершится.

## Описание внутренних структур данных.

Для хранения матрицы в классическом представлении используются

* массив длины n, элементами которого являются массивы длины m, элементами которых являются действительные числа;
* массив указателей длины n на начала массивов длины m.

Для хранения матрицы в разреженном представлении используются

* массив действительных чисел длины amount, который содержит значения ненулевых элементов;
* массив целых чисел длины amount, который содержит номера столбцов для элементов первого массива;
* массив целых чисел длины n, в элементе Nk которого находится индекс компонент в первых двух массивах, с которого начинается описание строки Nk исходной матрицы.

## Описание алгоритма

1. Выводится общая информация о программе
2. Вводятся количество строк n и количество столбцов m в исходных матрицах
3. Вводится количество ненулевых элементов в первой матрице
4. Выбирается режим ввода первой матрицы
5. Первая матрица вводится в выбранном режиме
6. Вводится количество ненулевых элементов во второй матрице
7. Выбирается режим ввода второй матрицы
8. Вторая матрица вводится в выбранном режиме
9. Первая матрица преобразуется к разреженной форме.
10. Вторая матрица преобразуется к разреженной форме.
11. Исходные матрицы складываются двумя способами – при хранении в обычной и разреженной формах
12. Выводятся исходные и полученная матрицы, а также сравнительная таблица.

На каждом этапе контролируется успешность ввода и выделения памяти, в случае ошибки программа выводит соответствующее сообщение

Код программы содержит комментарии по используемым в программе подпрограммам.

# Набор тестов с указанием, что проверяется

## Тесты на вводимые значения

|  |  |  |  |
| --- | --- | --- | --- |
| Какой параметр вводим | Что проверяем | Ввод | Результат |
| Количество строк в матрицах - n | Слишком маленькое | 0 | Сообщение об ошибке и предложение ввести снова |
| дробное | 2.9 |
| нечисловое | а |
| На нижней границе | 1 | Программа продолжает выполнение |
| Общий случай | 1000 |
| Количество столбцов в матрицах - m | Слишком маленькое | -1 | Сообщение об ошибке и предложение ввести снова |
| дробное | .9 |
| нечисловое | rt |
| На нижней границе | 1 | Программа продолжает выполнение |
| Общий случай | 333 |
| Количество ненулевых элементов в матрице - amount | Слишком маленькое | -1 | Сообщение об ошибке и предложение ввести снова |
| Слишком большое (для n = 2, m = 3) | 7 |
| дробное | 0.2 |
| нечисловое | p |
| На нижней границе | 0 | Программа продолжает выполнение |
| На верхней границе (для n = 2, m = 3) | 6 |
| Общий случай  (для n = 2, m = 3) | 1 |
| Режим ввода mode | Слишком маленький | -1 | Сообщение об ошибке и предложение ввести снова |
| Слишком большой | 2 |
| дробный | 1.1 |
| нечисловой | first |
| Первый возможный | 0 | Программа продолжает выполнение |
| Второй возможный | 1 |
| Ввод ненулевых элементов при вводе в ручном режиме  (n=3, m=4, amount = 2) | | | |
| номер строки i | Слишком маленький | -1 1 1 | Сообщение об ошибке и предложение ввести данный элемент снова |
| Слишком большой | 3 1 1 |
| дробный | 1.1 1 1 |
| нечисловой | A 1 1 |
| На нижней границе | 0 1 1 | Программа продолжает выполнение |
| На верхней границе | 2 1 1 |
| Общий случай | 1 1 1 |
| номер столбца j | Слишком маленький | 1 -2 1 | Сообщение об ошибке и предложение ввести данный элемент снова |
| Слишком большой | 1 4 1 |
| дробный | 1 1.2 1 |
| нечисловой | 1 a 1 |
| На нижней границе | 1 0 1 | Программа продолжает выполнение |
| На верхней границе | 1 3 1 |
| Общий случай | 1 1 1 |
| Значение элемента value | нулевое | 1 1 0 | Сообщение об ошибке и предложение ввести данный элемент снова |
| Близкое к нулю (в сравнении с EPS | 1 1 0.00000001 |
| Общий случай | 1 1 -0.1 | Программа продолжает выполнение |
| Ввод элемента на уже занятую позицию |  | 1 1 1  1 1 0.1 | Сообщение об ошибке и предложение ввести элемент на другую позицию |
| Ввод бОльшего числа параметров |  | 1 1 1 1 | Сообщение об ошибке и предложение ввести предложение ввести данный элемент снова |

(ввод для обеих матриц выполняется по одному алгоритму)

## Проверка заполнения матрицы в ручном режиме

Input number of rows n (int, >= 1): 3

Input number of columns m (int, >= 1): 4

How many nonnull elements will be in first matrix (from 0 to 12): 2

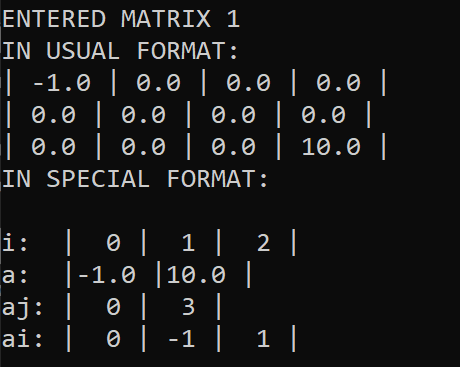
Choose input mode: 0 - for automatically, 1 - for manually: 1

Input element number 1:

0 0 -1

Input element number 2:

1. 3 10



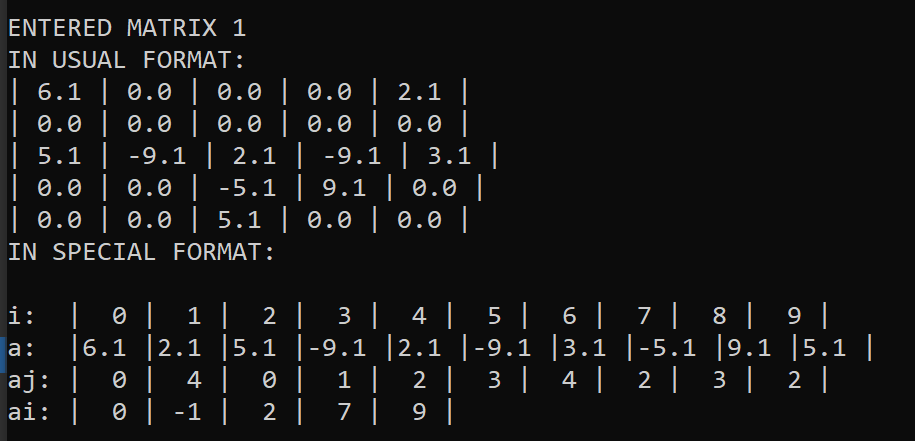
## Проверка заполнения матрицы в автоматическом режиме

Input number of rows n (int, >= 1): 5

Input number of columns m (int, >= 1): 5

How many nonnull elements will be in first matrix (from 0 to 25): 10

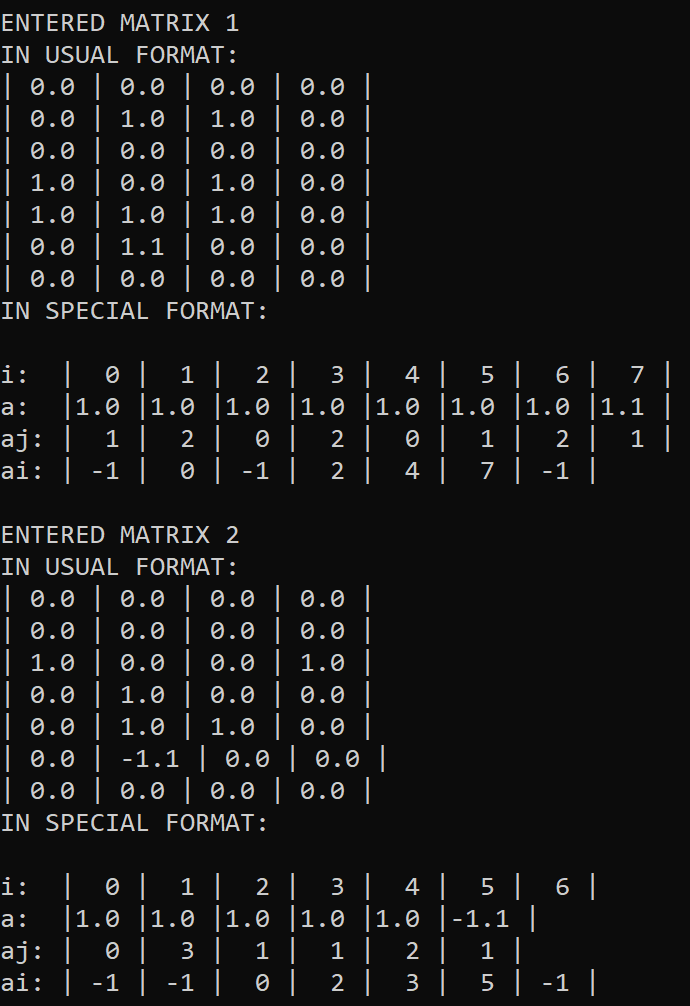
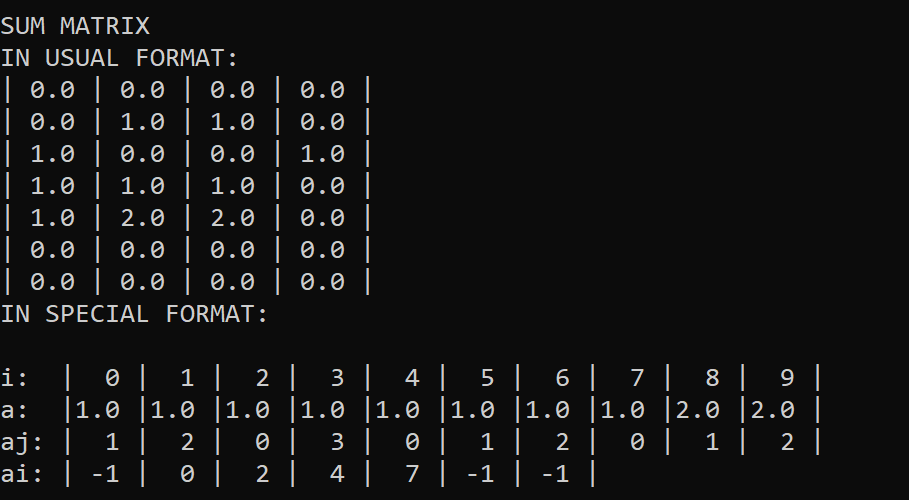
Choose input mode: 0 - for automatically, 1 - for manually: 0



## Проверка сложения матриц

Пояснения по строкам:

1. Была пустой в обеих матрицах и осталась пустой
2. Была непустой в первой матрице и пустой во второй – стала копией первой
3. Была пустой в первой матрице и непустой во второй – стала копией второй
4. Была непустой в обеих строках, но элементы не пересекались
5. Была непустой в обеих строках, некоторые элементы пересекались и произошло сложение
6. Была непустой в обеих строках, элементы пересекались, но в результате сложения обнулились
7. Была пустой в обеих матрицах и осталась пустой

# Оценка эффективности

Оценим эффективность (по памяти и по времени) различных подходов к представлению матриц (разреженный и классический) при различных размерностях и процентах заполенности матриц.

Размерности поварьируем так: 10\*10, 100\*100, 500\*500, 1000\*1000

Заполненность поварьируем так: от 1% до 79% с шагом в 3%

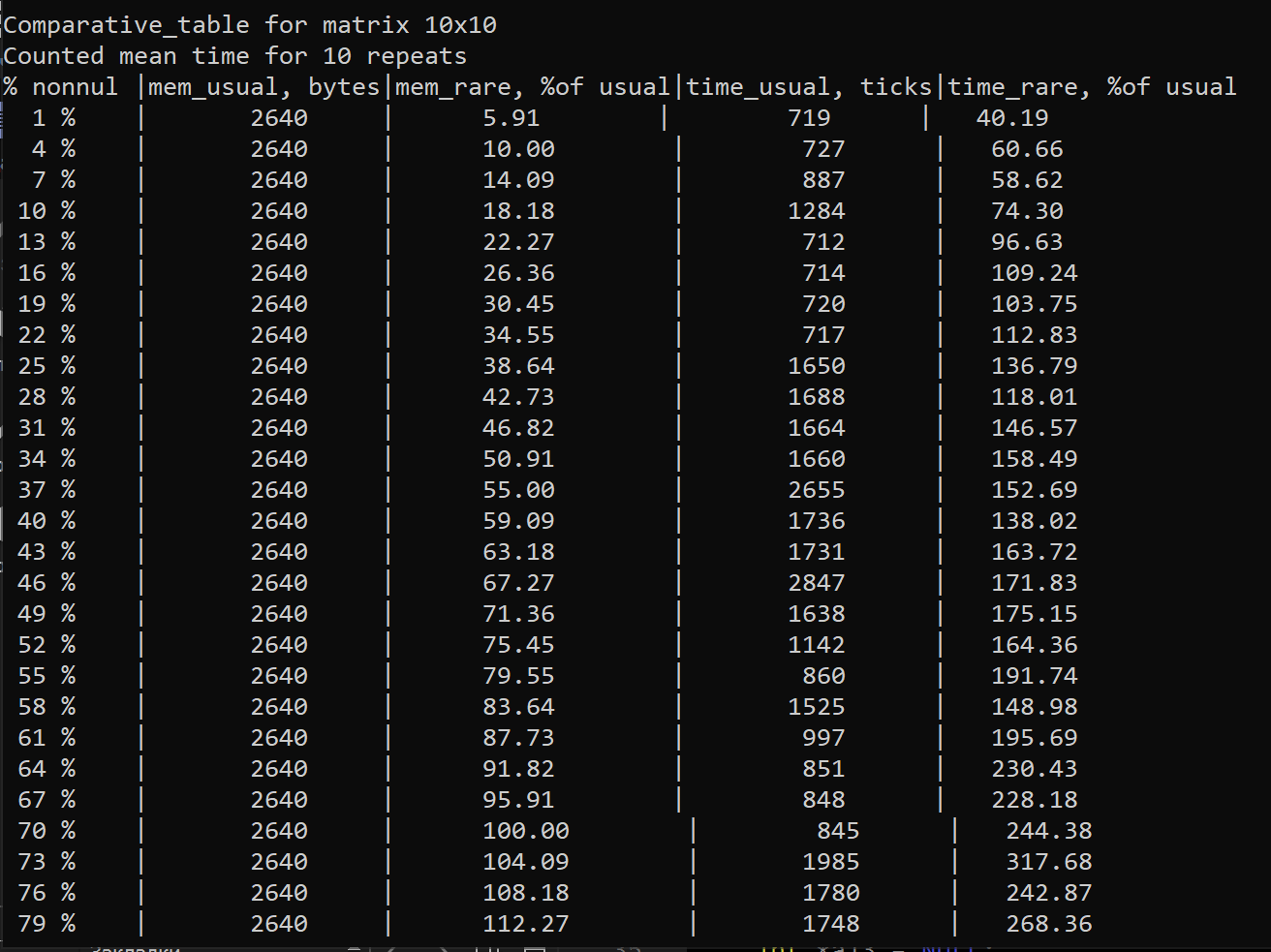
Оценивать будем в процентном отношении:

* по памяти: память, занимаемую матрицами в классическом представлении считаем за 100%.
* по времени: время, затраченное на сложение матриц в классическом представлении считаем за 100% (если оно равно 0, выводим время разреженного подхода в usec, а не в процентах).

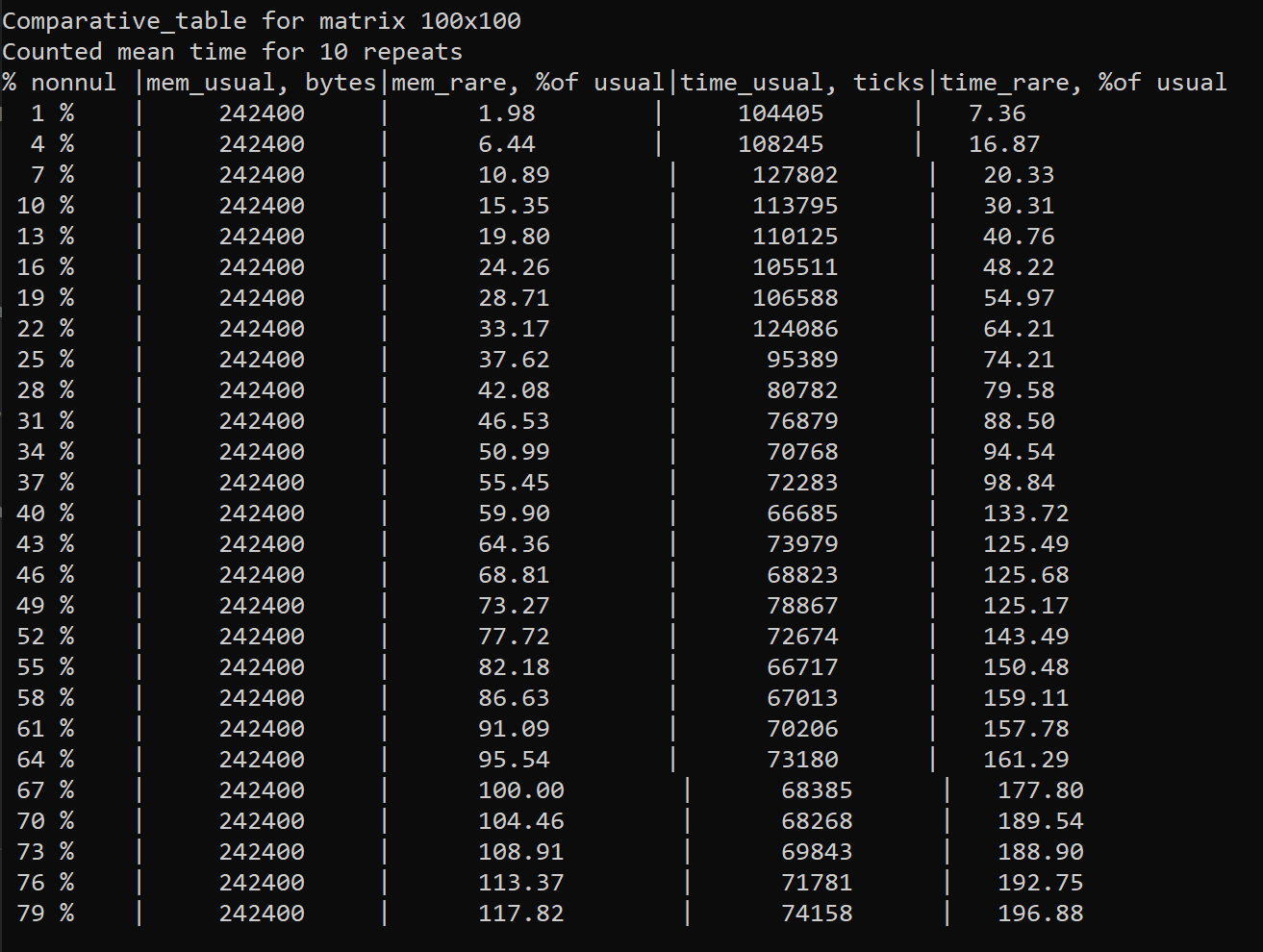
Примечания:

* время считается в тактах процессора
* время на выделение памяти не учитывается
* при подсчете памяти, занимаемой матрицей в классическом представлении, учитывается память для массива указателей на строки матрицы

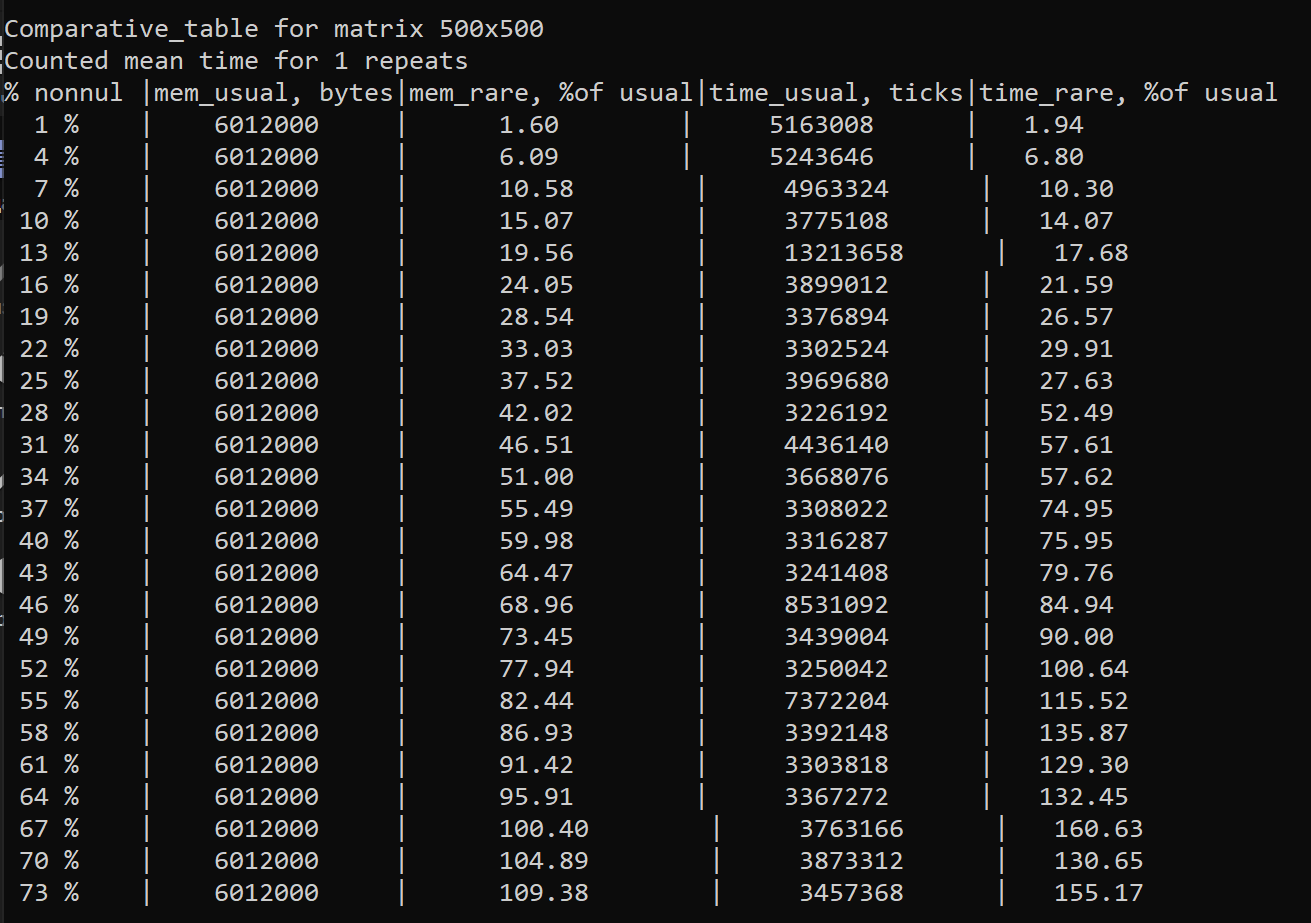
**10\*10**



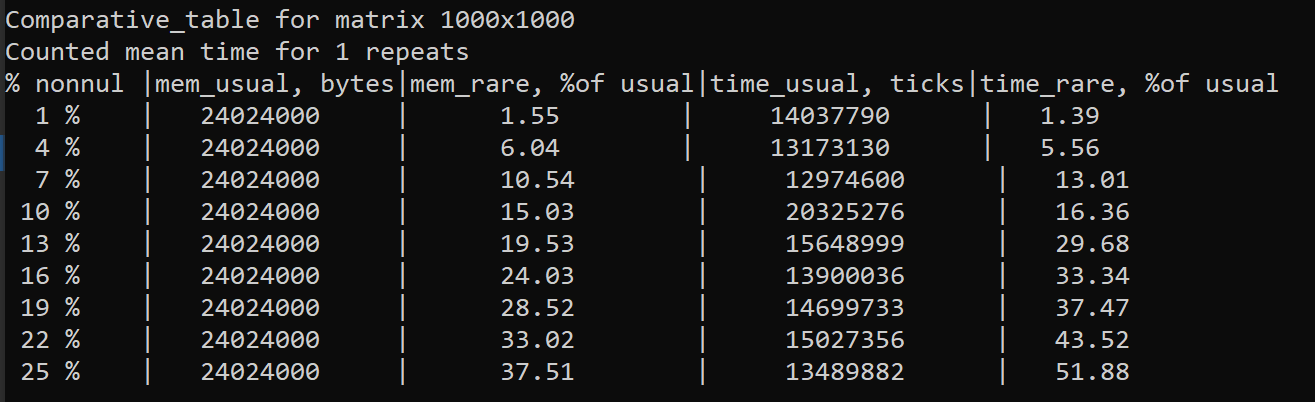
**100\*100**

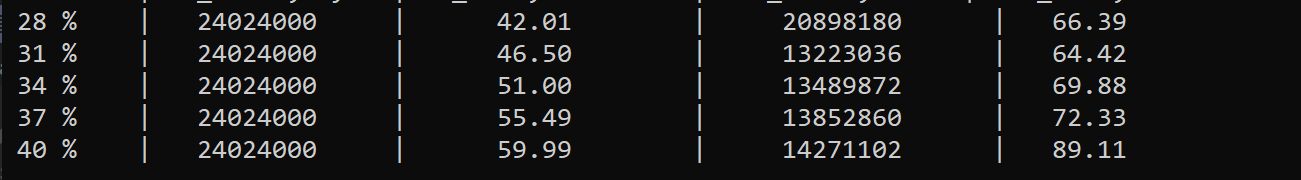


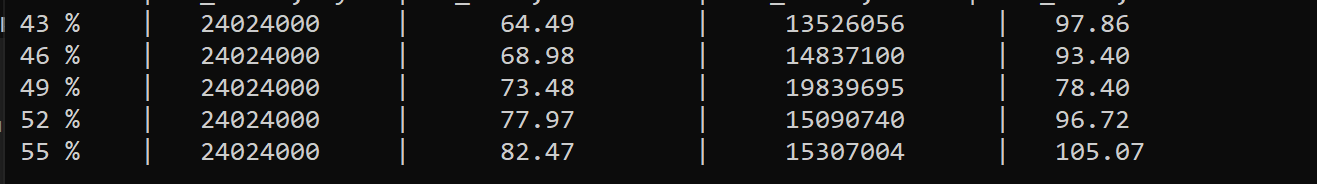
**500\*500**

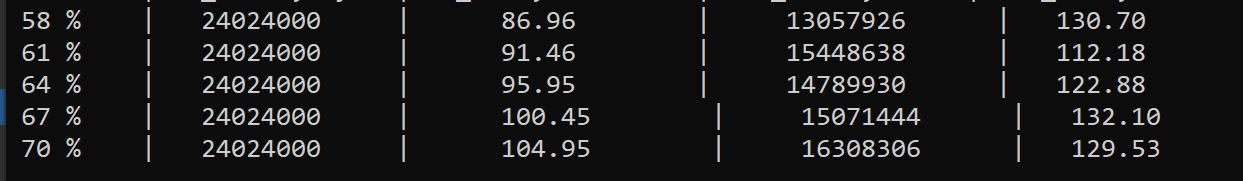
****

**1000\*1000**











|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Размерность | 10\*10 | 100\*100 | 500\*500 | 1000\*1000 |
| С какого % заполненности разреженная форма перестала выигрывать *по памяти* | 70 | 67 | 67 | 67 |
| С какого % заполненности разреженная форма перестала выигрывать *по времени* | 16 | 40 | 52 | 55 |

# Выводы по проделанной работе

При такой реализации хранения матрицы в классическом и разреженном видах:

* Сравнительная эффективность по памяти не зависит от размерности матриц, а зависит только от их разреженности. В среднем, для любой размерности – когда процент заполненности начинает превышать 67%, разреженное представление занимает больше места в памяти, чем классическое,
* Сравнительная эффективность по времени (в тактах процессора) зависит как от размерности матриц, так и от их разреженности: чем больше размерность матрицы, тем больше процент заполнения, при котором алгоритм разрежённой матрицы становится менее эффективным, чем классический.

Таким образом, при выборе формы представления разреженной матрицы, следует учитывать следующие факторы:

* Размерность матрицы
* Процент ее заполненности
* Что в конкретной задаче для нас важнее – эффективность по времени или эффективность по памяти

# Ответы на вопросы

1. Что такое разреженная матрица, какие схемы хранения таких матриц Вы знаете?

Матрица порядка n называется разреженной, если число ненулевых элементов в ней находится примерно в интервале от n^(1.2) до n^(1.4).

Некоторые схемы хранения:

* линейный связный список (последовательность ячеек, связанных в определенном порядке, каждая ячейка списка содержит элемент списка и указатель на положение следующей ячейки),
* кольцевой связный список,
* двунаправленные стеки,
* очереди,
* диагональная схема хранения симметричных матриц. Наиболее широко используемая схема хранения разреженных матриц - это схема, предложенная Чангом и Густавсоном, называемая "разреженным строчным форматом".

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

* В классическом представлении: n\*m\*sizeof(elem\_type). Память выделяется сразу под все элементы
* В разреженном – в зависимости от выбранной схемы хранения. Например, для описанной в задаче формы – amount \* sizeof(elem\_type) (память под ненулевые элементы матрицы) + amount\*sizeof(index\_type) (память под номера столбцов) + n\*sizeof(index\_type) (память под индексы, с которых начинаются описания строк).

1. Каков принцип обработки разреженной матрицы?

Он предполагает обработку только ненулевых элементов, а определение (в случае необходимости) их позиций в матрице зависит от выбранной формы хранения разреженной матрицы.

1. В каком случае для матриц эффективнее применять стандартные алгоритмы обработки матриц? От чего это зависит?

* Эффективность по памяти зависит от процента разреженности матрицы: чем больше в ней ненулевых элементов, тем более эффективно применять стандартные алгоритмы обработки матриц.
* Эффективность по времени зависит от процента разреженности матрицы и от ее размерности. Чем больше размерность матрицы, тем больше процент заполнения, при котором эффективнее применять стандартные алгоритмы обработки матриц.