**Архитектура вычислительных систем.**

Вариант 215, задача №5, дополнительная функция №16

Выполнил: Манжула Олег Олегович

БПИ206

1. **Описание задания**

### **Общие для всех альтернатив** переменные**:**

Размерность – целое число

### **Общие для всех альтернатив функции:**

### Вычисление среднего арифметического (действительное число)

### **Обобщенный артефакт** - Квадратные матрицы с действительны ми числами

### **Базовые альтернативы**

1. Обычный двумерный массив

2. Диагональная (на основе одномерного массива)

3. Нижняя треугольная матрица (одномерный массив с формулой пересчета)

1. **Основные характеристики программы**

1) число интерфейсных модулей (заголовочных файлов) – 0

2) число модулей реализации (фалов с определением программных объектов) – 6

3) общий размер исходных текстов – 16 Кб

4)время выполнения программы для различных тестовых данных:

|  |  |
| --- | --- |
| **№ теста** | **Время выполнения**  **3 программа** |
| **Тест\_1** | 0.003 сек |
| **Тест\_2** | 0.00297 сек |
| **Тест\_3** | 0.002 сек |
| **Тест\_4** | 0.002 сек |
| **Тест\_5** | 0.002 сек |
| **Тест\_6** | 0.00303 сек |
| **Тест\_7** | 0.002 сек |
| **Тест\_8** | 0.00154 сек |
| **Тест\_9** | 0.002 сек |
| **Тест\_10** | 0.00172 сек |
| **Тест\_11** | 0.002 сек |
| **Тест\_12 (100 матриц)** | 0.00815 сек |
| **Тест\_13 (500 матриц)** | 0.05284 сек |
| **Тест\_14 (1000 матриц)** | 0.12647 сек |

После 11 теста – случайные данные.

1. **Ввод данных**

|  |
| --- |
| ***Пример возможного теста:***  1  7  1 2 3 4 0 14 15 |

Формат входных данных в файлах (определён автором работы).

В первой строке одно из трёх чисел: 0, 1 или 2. Определяет тип матрицы (0 - обычная, 1 - диагональная, 2 - нижнетреугольная).

Во второй строке натуральное число - размерность матрицы.

Если матрица обычная, то далее вводятся значения элементов, разделённые пробелом, строка за строкой.

Диагональная и нижнетреугольная матрицы задаются одной строкой.

Команда терминала должна иметь один из следующих видов:

* command -f <input file> <out file01> <out file02> command -r <matrix type>
* <matrix size> <output file name>

Файлы с наборами данных для тестирования расположены в директории «/AvsLab0/tests», корректные результаты находятся в ««/AvsLab0/results». В файле с названием «output\_n\_1.txt» лежат данные n-ого теста до сортировки, а в файле «output\_n\_2.txt» - после сортировки.

1. **Схемы:**

**Отображение на память содержимого модуля main**

Изображение выглядит как стол

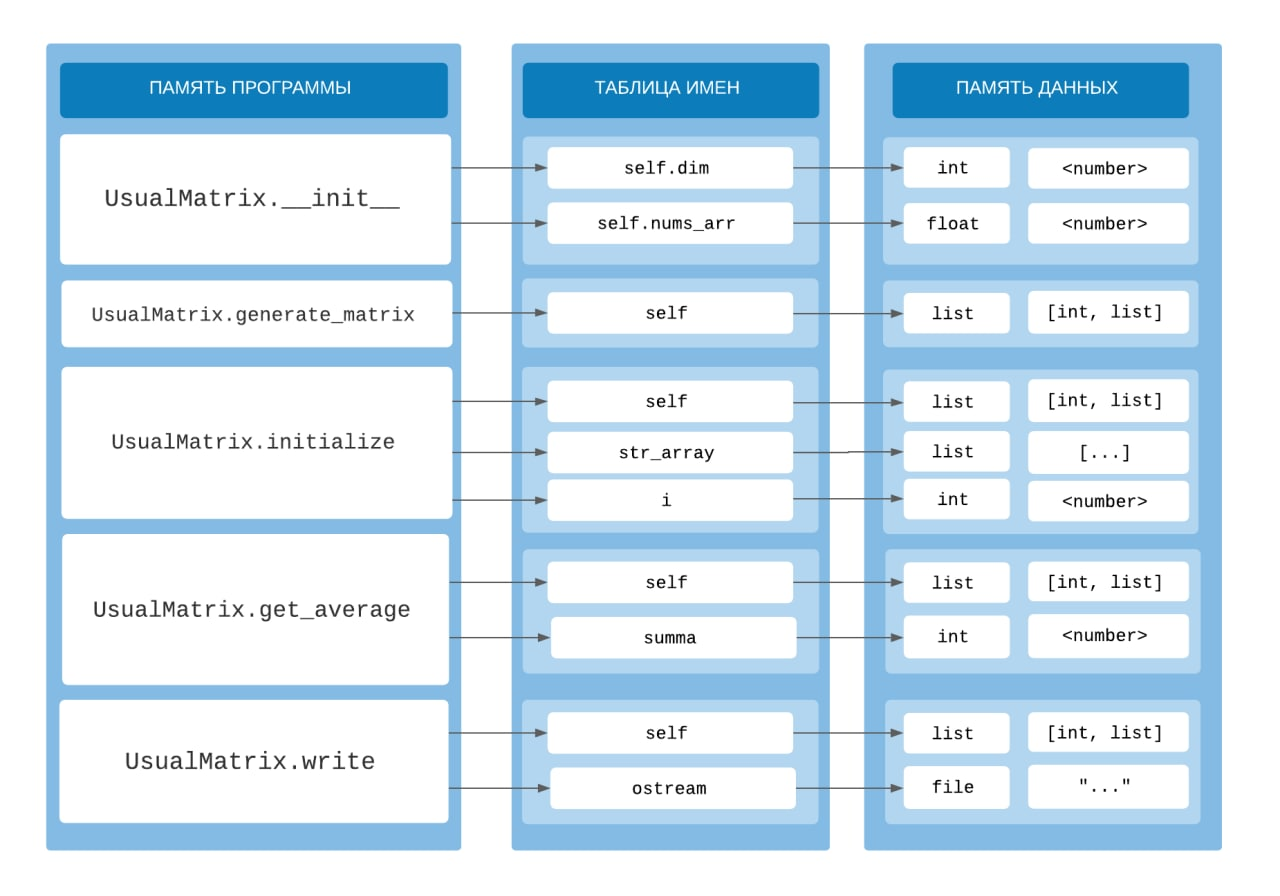
Автоматически созданное описание

**Отображение классов**

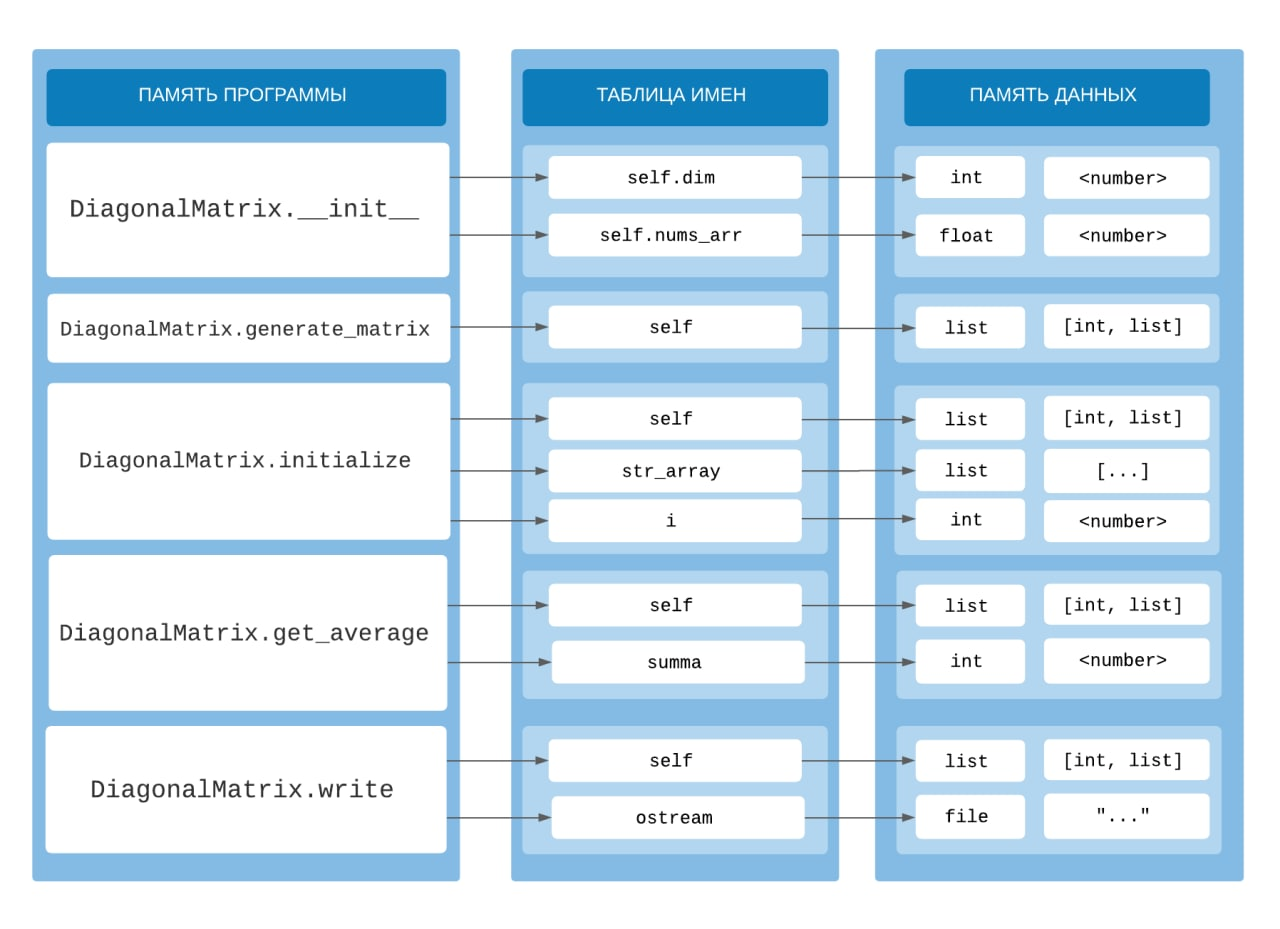
Изображение выглядит как стол

Автоматически созданное описание

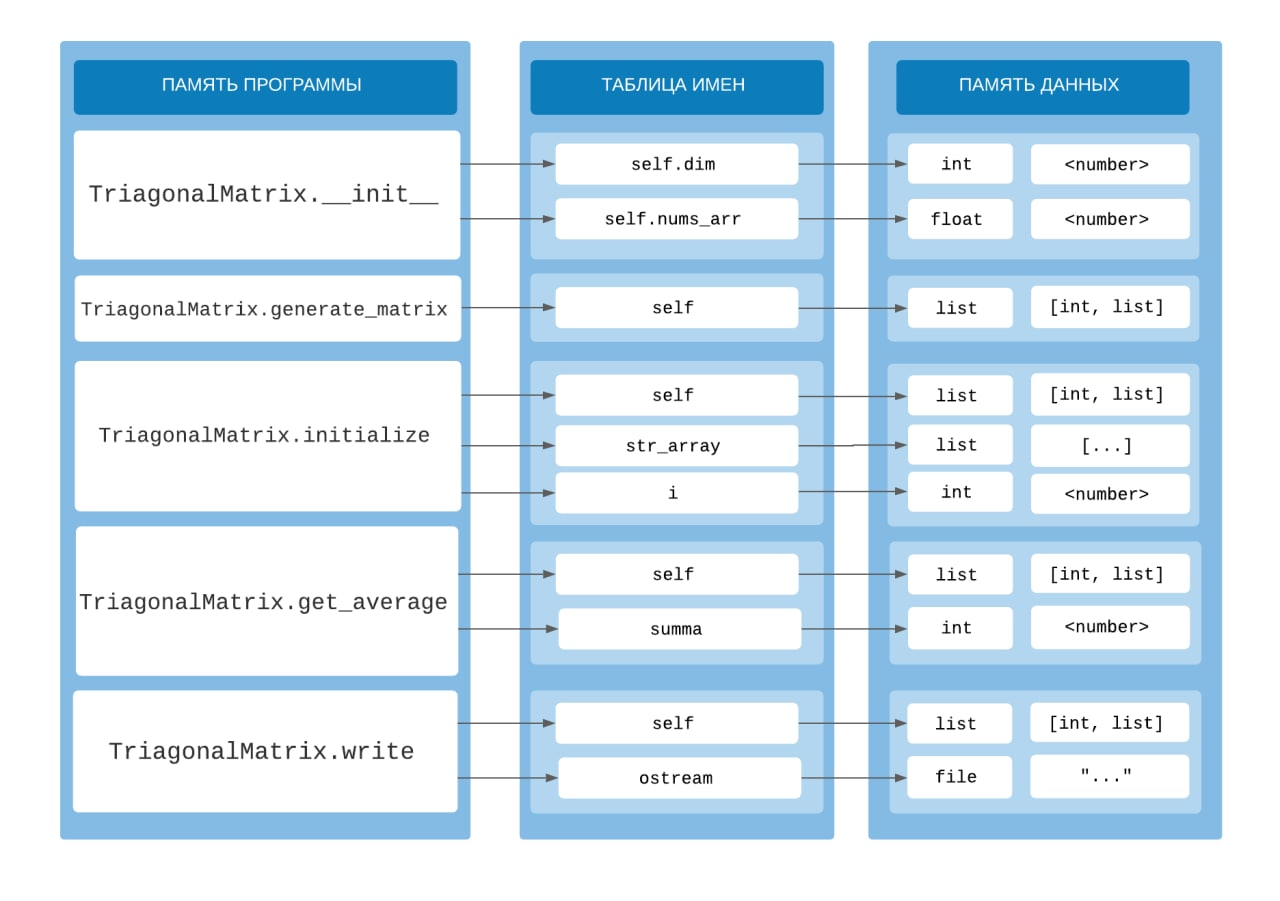
**Отображение на память методов класса UsualMatrix**



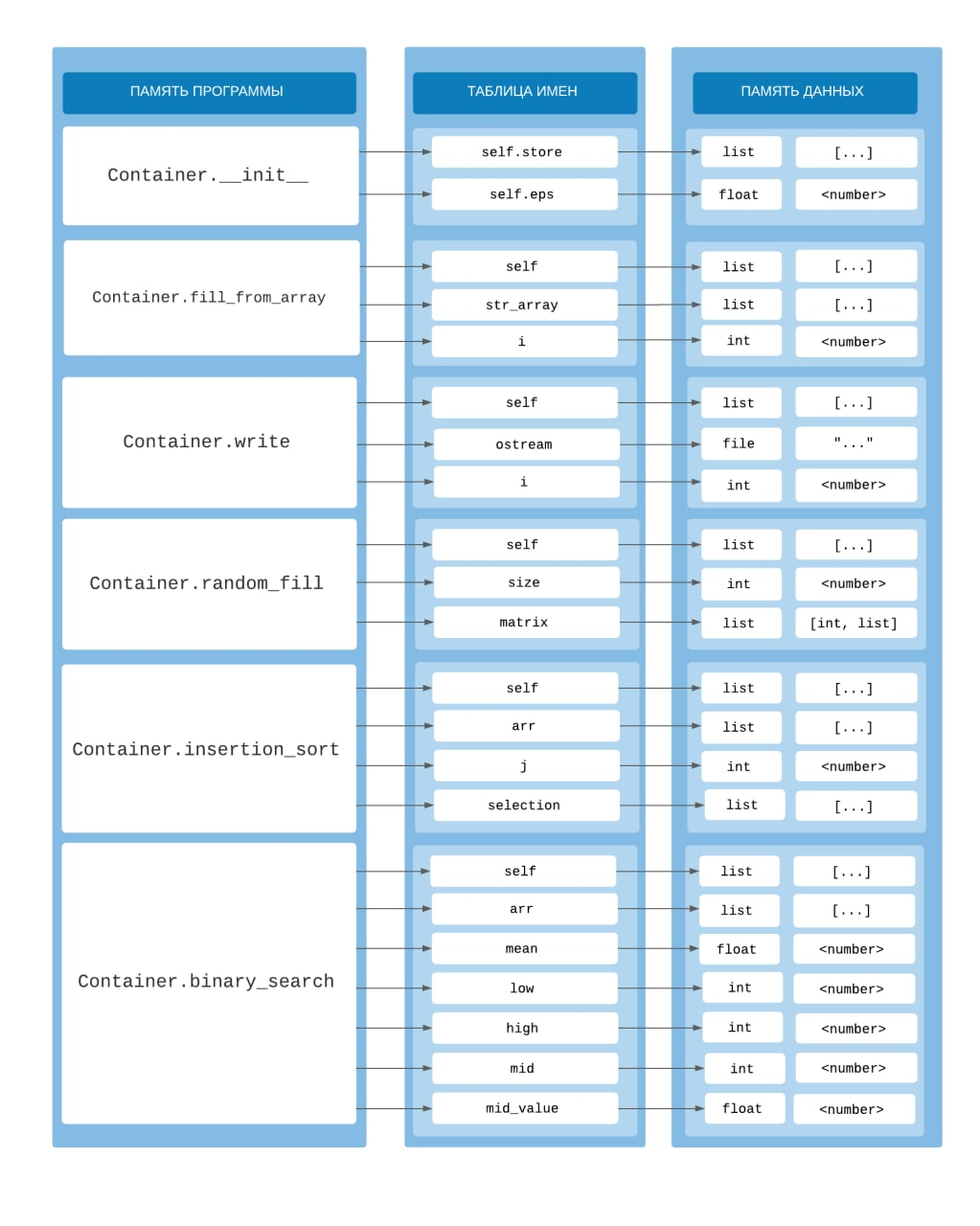
**Отображение на память методов класса DiagonalMatrix**



**Отображение на память методов класса TriagonalMatrix**



**Отображение на память методов класса Container**



1. **Результаты сравнительного анализа полученных характеристик с теми, которые были получены для предыдущей программы**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Время выполнения**  **1 программа** | **Время выполнения**  **2 программа** | **Время выполнения**  **3 программа** |
| **Тест\_1** | 0.015000 сек | 0.016000 сек | 0.003 сек |
| **Тест\_2** | 0.015000 сек | 0.015000 сек | 0.00297 сек |
| **Тест\_3** | 0.015000 сек | 0.015000 сек | 0.002 сек |
| **Тест\_4** | 0.000000 сек | 0.015000 сек | 0.002 сек |
| **Тест\_5** | 0.000000 сек | 0.016000 сек | 0.002 сек |
| **Тест\_6** | 0.015000 сек | 0.016000 сек | 0.00303 сек |
| **Тест\_7** | 0.016000 сек | 0.015000 сек | 0.002 сек |
| **Тест\_8** | 0.015000 сек | 0.015000 сек | 0.00154 сек |
| **Тест\_9** | 0.218000 сек | 0.219000 сек | 0.002 сек |
| **Тест\_10** | 0.062000 сек | 0.078000 сек | 0.00172 сек |
| **Тест\_11** | 0.109000 сек | 0.063000 сек | 0.002 сек |

* *Начиная с 7 теста, время помечено серым цветом, поскольку в программу были внесены некоторые изменения и данные тесты (8-11) не пригодны для заданий 1 и 2*

**Вывод:**

1. Не требуется в динамической типизации использовать интерфейсные модули
2. Динамическая типизация занимает больше времени, чем программы со статической типизацией(то есть, в предыдущих программах).
3. Чтение данных в динамически типизированном языке проще, чем в статической типизации
4. Динамическая типизация также обладает компактностью кода