Описание задания (задание 5, дополнительная функция 5):

Разработать программный продукт с использованием объектно-ориентированного подхода и статической типизацией. Программа должна содержать следующие структуры:

Обобщенный артефакт, используемый в задании:

Обобщенный артефакт, используемый в задании	Базовые альтернативы (уникальные параметры, задающие отличительные признаки альтернатив)	Общие для всех альтернатив переменные	Общие для всех альтернатив функции
Квадратные матрицы с действительными числами	1. Обычный двумерный массив 2. Диагональная (на основе одномерного массива) 3. Нижняя треугольная матрица (одномерный массив с формулой пересчета)	Размерность – целое число	Вычисление среднего арифметического (действительное число)

Для всех альтернатив общей переменной является размерность (целое число). Оно может принимать значения от 1 до 50.
Общей функция всех альтернатив выступает вычисление среднего арифметического (действительное
число). В качестве дополнительной функции необходимо отсортировать средние значения всех матриц с помощью Shell Sort.
Также нужно: разработать тестовые входные данные и провести тестирование и отладку программы на этих данных (при необходимости, программа должна правильно обрабатывать переполнение по данным); описать структуру используемой ВС с наложением на нее обобщенной схемы разработанной программы; зафиксировать количество заголовочных файлов, программных файлов, общий размер исходных текстов, полученный размер исполняемого кода (если он формируется), время выполнения программы для различных тестовых наборов данных.

СТРУКТУРНАЯ СХЕМА АРХИТЕКТУРЫ ВС С ПРОГРАММОЙ:

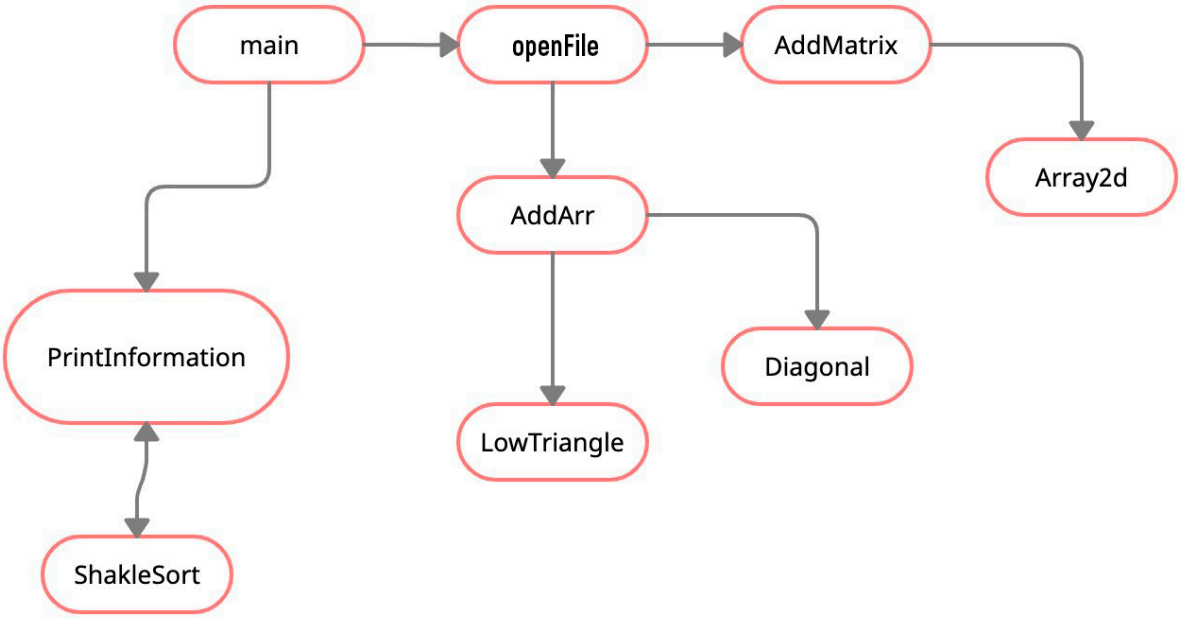
Программа разработана в 64 битной системе (дистрибутив Ubuntu на ярде Linux).

Таблица типов

Название	Размер
int	4
double	8
Class Matrix middle : double dimension : int	4*(dimension^2) 8 4
Class Container: maxLen : int curLen : int Array2d[] Diagonal[] LowMatrix[]	32 + 12*(dimension^2) 4 4 8 +4*(dimension^2) 8 +4*(dimension^2) 8 +4*(dimension^2)
Array2d mid : double array[][]	8 +4*(dimension^2) 8 4*(dimension^2)

Diagonal mid : double array[][]	8 +4*(dimension^2) 8 4*(dimension^2)
LowMatrix mid : double array[][]	8 +4*(dimension^2) 8 4*(dimension^2)

Блок схема возможного стека, в результате работы функции main (с глубиной в 1 шаг):



Описание работы для функции add_arr в рамках архитектуры:

Stack
typeOfMatrix
demension
curLen

Память программы

`def add_arr(self, arr, typeOfMatrix, demension):`
`  if typeOfMatrix == 2:`
`  self.matrixesDiagonal.append(Diagonal(arr, demension))`
`  self.curLen += 1`
`  else:`
`  self.matrixesLowMatrix.append(LowMatrix(arr, demension))`
`  self.curLen += 1`

Heap
Arr[]
Diagonal
LowMatrix

Описание работы для функции add_matrix в рамках архитектуры:

Stack
typeOfMatrix
curLen

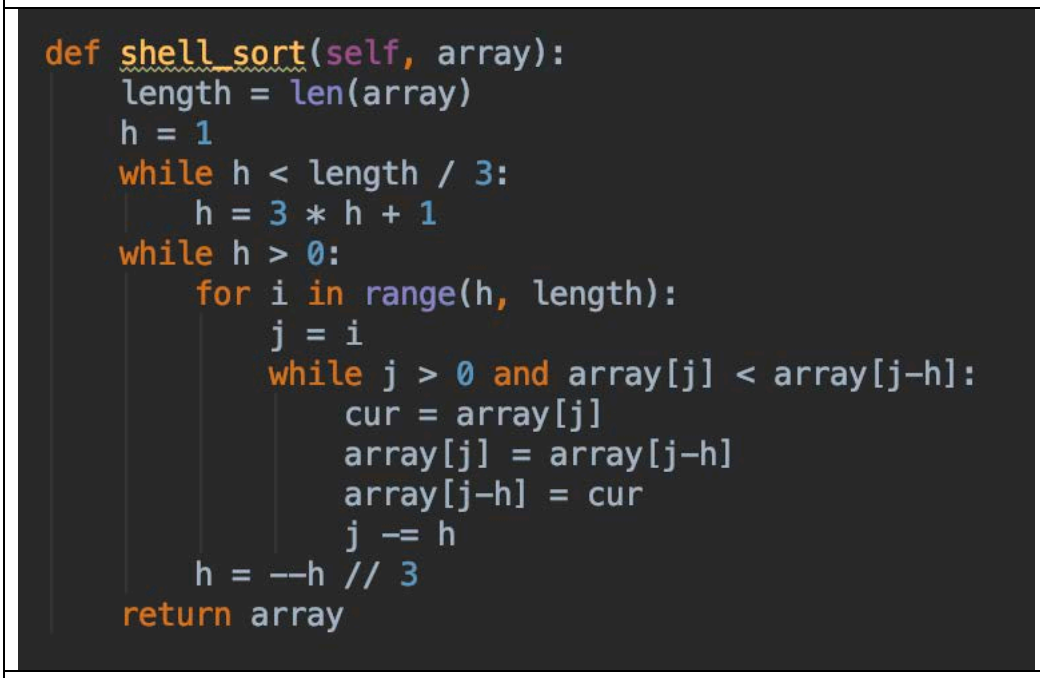
Память программы


```
def add_matrix(self, arr, typeOfMatrix):
    self.matrixesArr2d.append(Array2d(arr, len(arr)))
    self.curLen += 1
```

Heap
Arr[]
Array2d

Описание работы для функции shell_sort в рамках архитектуры:

Stack
length
h

Память программы


```
def shell_sort(self, array):
    length = len(array)
    h = 1
    while h < length / 3:
        h = 3 * h + 1
    while h > 0:
        for i in range(h, length):
            j = i
            while j > 0 and array[j] < array[j-h]:
                cur = array[j]
                array[j] = array[j-h]
                array[j-h] = cur
                j -= h
        h = --h // 3
    return array
```

Heap
Array[]

Основные характеристики программы:

Число заголовочных файлов – 6

Число модулей реализации – 6

Общий размер исходных текстов – 278 строки включай комментарии (на 95 строк меньше, по сравнению с C++)

Время работы программы:

1 тест - 0.012513875961303711 секунды

2 тест - 0.011850833892822266 секунды

3 тест - 0.011329889297485352 секунды

4 тест - 0.010828971862792969 секунды

5 тест - 0.052462100982666016 секунды

Размер кода: 76 килобайт