**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.

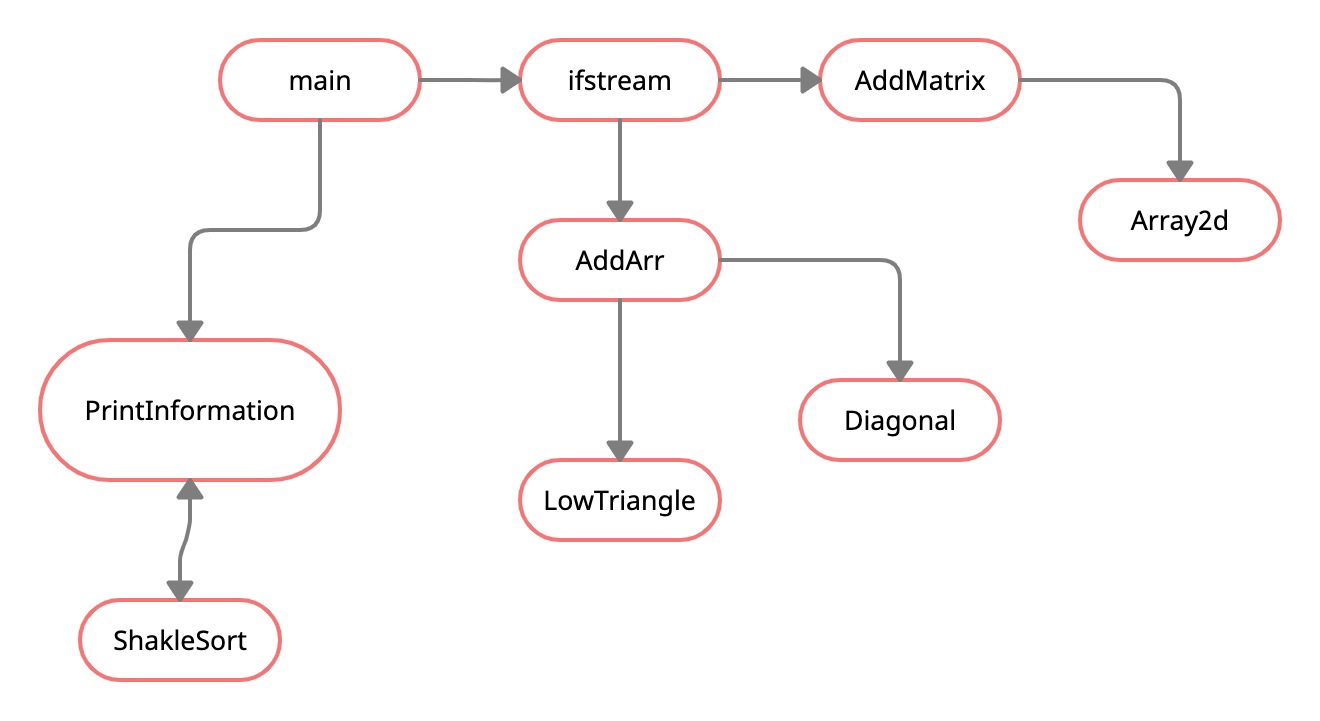
**СТРУКТУРНАЯ СХЕМА АРХИТЕКТУРЫ ВС С ПРОГРАММОЙ:**

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  vector<Array2d>  vector<Diagonal>  vector<LowMatrix> | 32 + 12\*(dimension^2)  4  4  8 +4\*(dimension^2)  8 +4\*(dimension^2)  8 +4\*(dimension^2) |
| Array2d  mid : double  vector<int> | 8 +4\*(dimension^2)  8  4\*(dimension^2) |
| Diagonal  mid : double  vector<int> | 8 +4\*(dimension^2)  8  4\*(dimension^2) |
| LowMatrix  mid : double  vector<int> | 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 AddArr в рамках архитектуры:**

|  |
| --- |
| **Stack** |
| typeOfMatrix |
| demension |
| curLen |
|  |
|  |
|  |

|  |
| --- |
| **Память программы** |
| void AddArr(std::vector<int> arr, int typeOfMatrix, int demension){ if (typeOfMatrix == 2)  {  matrixesDiagonal.push\_back(Diagonal(arr,  demension));  curLen++; }  else  { matrixesLowMatrix.push\_back(LowMatrix(  arr, demension)); curLen++; } } |
|  |

|  |
| --- |
| **Heap** |
| Vector<int> arr |
| Diagonal |
| LowMatrix |

**Описание работы для функции AddMatrix в рамках архитектуры:**

|  |
| --- |
| **Stack** |
| typeOfMatrix |
| curLen |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Память программы** |
| void AddMatrix(std::vector<std::vector<int» arr, int typeOfMatrix){ matrixesArr2d.push\_back(Array2d(arr, arr.size())); curLen++; } |
|  |

|  |
| --- |
| **Heap** |
| Vector< Vector<int>> arr |
| Array2d |

**Основные характеристики программы:**

Число заголовочных файлов – 5

Число модулей реализации – 6

Общий размер исходных текстов – 373 строки включай комментарии