|  |  |
| --- | --- |
| Программа для расшифровки текста | |
| Внутренняя спецификация | |
| Студент | Быков А.А. |
| Преподаватель | преп. каф. ПОАС Матюшечкин Д.С. |
| Сдано |  |
| Лабораторная работа №2 |  |

# 1 Общие сведения

Наименование программы – «Программа для распаковки строки».

Для функционирования программы необходима операционная система

Windows 7 или выше.

Программа написана на языке C++.

# 2 Описание логической структуры программы

2.1 Алгоритм программы

1. Считать входные данные из входного файла.
2. Определить размерность матрицы
3. Проверить корректность входных данных.
4. Считать операцию над матрицой
5. Получить результат операции
6. Вывести в результат операции

## 2.2 Декомпозиция программы

Выделенные подпрограммы (функции) описаны в приложении А.

Иерархия вызовов подпрограмм представлена в приложении Б.

Диаграмма потоков данных представлена в приложении В.

# 

# Приложение А

Описание функций

Функция: **int main(const int argc, char\*\* argv)**

Обеспечить считывание из файла, вывод в консоль ошибок, если они есть, запись ответа в выходной файл, вызов главной функции, решающей задачу.

Алгоритм работы функции (псевдокод):

Выдать ошибку в консоль, если выходной файл невозможно открыть

Выдать ошибку в консоль, если выходной файл имеет неправильное расширение

Записать ошибку в выходной файл, если входной файл невозможно открыть

Записать ошибку в выходной файл, если входной файл имеет неправильное расширение

Вызвать функцию определяющую размер матрицы

Вызвать функцию проверяющую размер матриц

Если размер матрицы больше ограничения

{

Записать ошибку в выходной файл

Завершить работу программы

}

Иначе

{

Заполнить матрицы нулями

}

Считать матрицы из файлов

Если в файле есть некорректные данные

{

Записать ошибку в выходной файл

Завершить работу программы

}

Считать операцию над матрицой

Выполнить операцию над матрицей

Записать результат в выходной файл

Функция:

**bool getMatrixFromFile(int arr[][MAX\_SIZE], ifstream& file, string path)**

Алгоритм работы функции (псевдокод):

Открыть файл

Пока не конец файла

{

Получить строку

Вызвать функцию получения массива из строки

}

Функция:

**bool getArrFromRow(int arr[][MAX\_SIZE], int row, string line)** Алгоритм работы функции (псевдокод):

Выделить первую часть строки со значением

Пока в строке есть данные

{

Выделять очередную часть строки со значением

Передавать это значение в функцию получения числа из символов

}

Функция:

**bool getNumberFromRow(char \* value,int arr[][MAX\_SIZE], int row, int &col)**

Алгоритм работы функции (псевдокод):

Для всех элементов массива выделенной части

{

Если выделенная часть не число и не минус

{

Вернуть флаг некорректных данных

}

Иначе

Преобразовать выделенную часть в число

Записать число в массив

}

Функция:

**int getNumberOfRows(ifstream& file, string path)**

Алгоритм работы функции (псевдокод):

Открыть файл

Пока не конец файла{

Получить строку

Увеличить счетчик строк

}

Вернуть счетчик строк

Функция:

**int getNumberOfColumns(ifstream& file, string path)**

Алгоритм работы функции (псевдокод):

Открыть файл

Пока не конец файла{

Получить строку

Разбить строку на подстроки по разделителям

Пока есть очередная подстрока

{

Увеличить счетчик столбцов

}

Занести полученное значение в массив значений столбцов для каждой строки

}

Вызвать функцию получения наибольшего значения столбцов

Вернуть набольший размер столбцов

Функция:

**int getMaxRowSize(int arr[], int size)**

Алгоритм работы функции (псевдокод):

Записать первый элемент массива в наибольший значение столбцов

Для всех элементов массива

{

Если наибольший элемент больше очередного

Занести очередной элемент в наибольшее значение

}

Вернуть наибольшее значение столбцов

Функция:

**void getSizeLimitOfMatrix(int x, int y)**

Алгоритм работы функции (псевдокод):

Если первый размер больше второго

Вернуть первый размер

Иначе

Вернуть второй размер

Функция:

**bool isSizeCorrect(int size[])**

Алгоритм работы функции (псевдокод):

Проверить, что полученные размеры лежат в диапазоне 0 до 20

Функция:

**void setZeroMatrix(Matrix\* matr, int size[])**

Алгоритм работы функции (псевдокод):

Для всех элементов матрицы

Присвоить 0

Функция:

**void matrixAdd(int matr1[][MAX\_SIZE], int matr2[][MAX\_SIZE], int result[][MAX\_SIZE], int size[])**

Алгоритм работы функции (псевдокод):

Для всех элементов матрицы

Сложить элемент первой матрицы с элементов второй матрицы

Функция:

**void matrixSub(int matr1[][MAX\_SIZE], int matr2[][MAX\_SIZE], int result[][MAX\_SIZE], int size[])**

Алгоритм работы функции (псевдокод):

Для всех элементов матрицы

Вычесть из элемента первой матрицы элемент второй матрицы

Функция:

**void matrixSub(int matr1[][MAX\_SIZE], int matr2[][MAX\_SIZE], int result[][MAX\_SIZE], int size[])**

Алгоритм работы функции (псевдокод):

Для всех элементов матрицы

Зануляем значение элемента результирующего массива

Для всех элементов строки первой матрицы и столбца второй матрицы

Прибавляем результат умножения элемента строки первой матрицы на элемент столбца второй матрицы

# 

# 

# Приложение Б

Иерархия вызовов подпрограмм

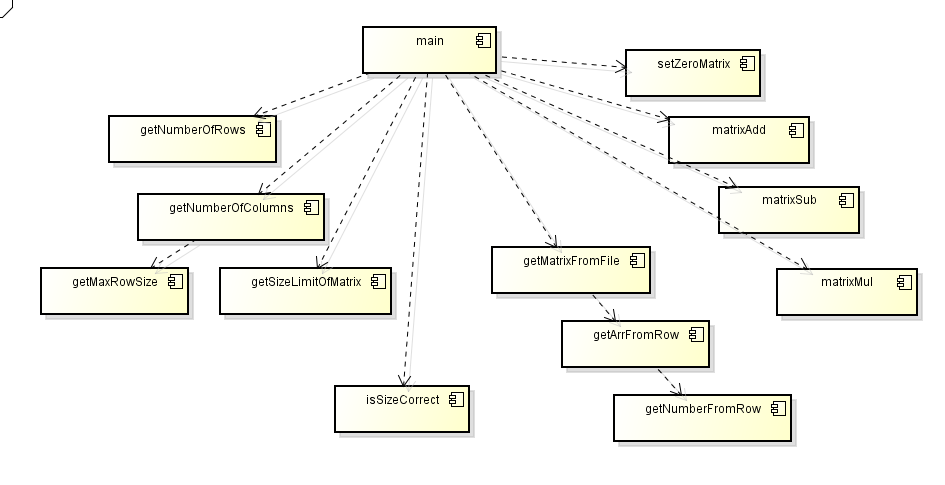


Рис. 1. Иерархия вызовов функций.

# Приложение В

Диаграмма потоков данных

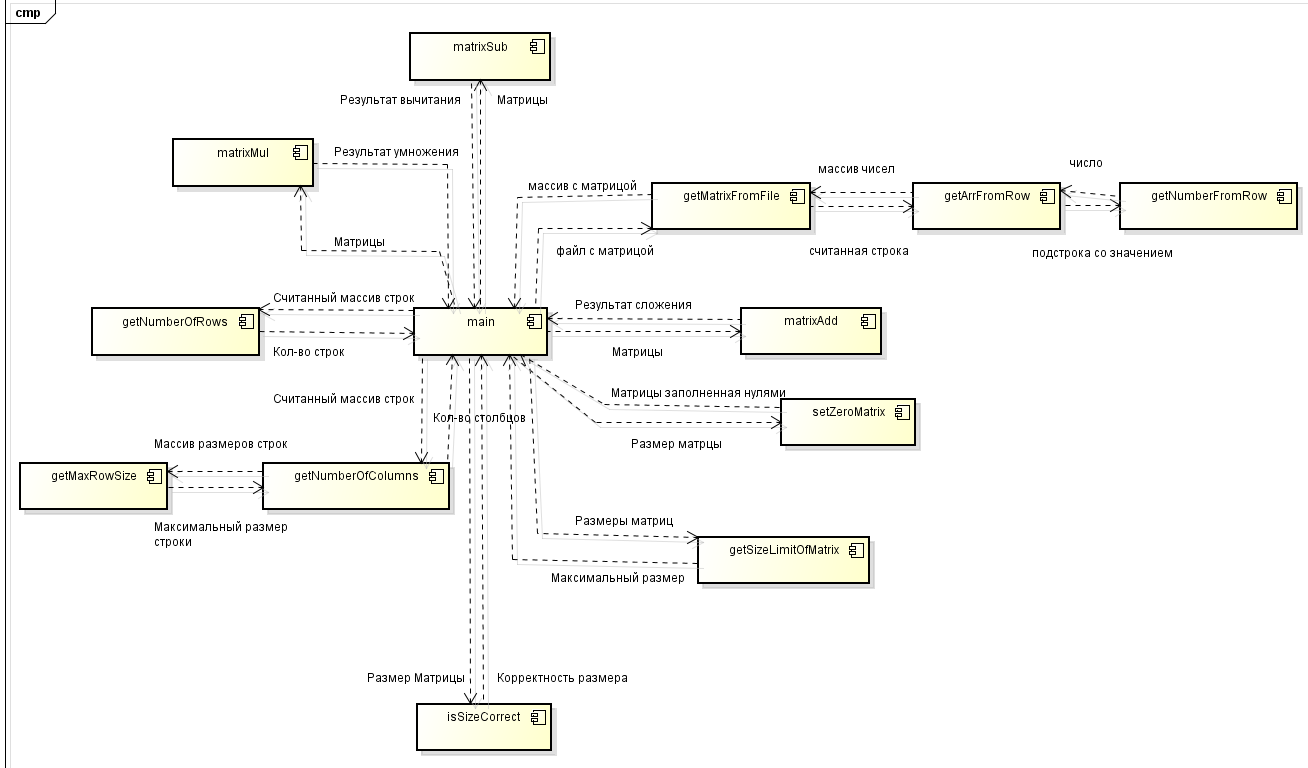


Рис. 2. Диаграмма потоков данных