**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе № 5**

**по дисциплине «Программирование»**

Тема: Использование файлов для ввода-вывода данных. Программирование функций пользователя

|  |  |  |
| --- | --- | --- |
| Студент гр. 2309 |  | Савин П.А. |
| Преподаватель |  | Калмычков В.А. |

Санкт-Петербург

2022

**Формулировка задания.**

Дана матрица A n x m. На ее основе сформировать вектор B, элементы которого равны суммам элементов столбцов матрицы.

**Анализ задания.**

Для уточнения, речь идет про вектор-строку (размер 1 x m). Для разнообразия, в рамках прошлой лабораторной работы происходила обработка целых чисел, теперь же возьмем вещественные числа.

**Математическая** **постановка.**

Дано:

Найти:

Решение:

Задать формулу очередного элемента вектора B:

Пройти по каждому столбцу, просуммировать элементы с каждого ряда и записать их сумму в очередной элемент вектора.

**Контрольный пример.**

Для контрольного примера была взята следующая матрица 4 x 3:

Матрица постоянных из математики, физики и химии образует следующий вектор:

**Особенности реализации задания на компьютере.**

Возможная матрица хранится в файле. Почему возможная? Из гарантированного в формате файла только первая строка – размер матрицы строки-столбцы в целых числах. Далее файл имеет хаотичную структуру, но, гарантированно, только из следующих символов: табуляция (\t), переход на новую строку (\n), пробел (\_) и неразрывная последовательность, образующая вещественное число.

Появляется новая задача – рано полагаться на считанные в 1 строке числа, нужно вручную прогнать файл и определить, получается ли вообще матрица (оба измерения матрицы должны быть строго положительными целыми числами), и если получается, то какие у нее будут размеры. Поскольку упор идет на желание пользователя, определение размеров будет происходить по следующим правилам:

1. В первую очередь добиваемся выполнения «квоты» по строкам

2. Пустые строки в файле, т.е. в которых не обнаруживается ни одной последовательности числа, игнорируются

3. Как только обнаруживается строка, количество элементов в которой меньше заданного числа столбцов, заданное число столбцов подстраивается под него

4. Строки, выходящие за «квоту», игнорируются

5. Если строк недостает, заданное число строк подстраивается под новое значение

6. Если файл совсем пустой или он состоит из пустых строк, объявляется о невозможности создания матрицы и бессмысленности дальнейшей работы программы

За этой задачей следует еще одна – основная часть кода должна быть разбита на функции, которые объявляются в основной. Следовательно, все функции будут обрабатывать уже существующие в main объекты, параллельно накладывая свои внутренние. Передать объекты можно разными способами: по значению, по ссылке и по указателю. Для удобства будем передавать переменные по значениям (в случае двумерного массива, т.к. взаимодействие с ним в этом случае не меняется) и по ссылкам (во всех остальных случаях).

**Разработка интерфейса пользователя.**

Как и в прошлый раз, программа работает в рамках двух файлов, поэтому пользователю вновь не предоставлен никакой интерфейс.

**Описание используемых данных.**

В скобках указана функция, которой принадлежит переменная.

|  |  |  |
| --- | --- | --- |
| Тип | Имя | Назначение |
| int | rows (main) | хранение исходного значения строк |
| cols (main) | хранение исходного значения столбцов |
| arr\_start (defsize) | хранение позиции в файле, с которой начинается последовательность строк |
| line\_start (inarr) | хранение позиции в файле, с которой начинается текущая строка |
| line\_end (inarr) | хранение позиции в файле, на которой заканчивается текущая строка |
| row (process) | указатель на динамически выделяемое в памяти число строк (заглушка в функцию outrow) |
| unsigned int | row\_count (defsize) | подсчет строк, годящихся для запись в массив |
| col\_count (defsize) | подсчет столбцов для каждой строки |
| char\_count (defsize) | подсчет непустых символов в строке (контроль за наполненностью строки; сбрасывается, как только обнаружено очередное число) |
| chars\_count (inarr) | подсчет непустых символов в строке (контроль за наполненностью строки; накапливается на всю строку) |
| rows\_count (inarr) | подсчет строк |
| cols\_count (inarr) | подсчет столбцов (установка крайней точки для записи в массив) |
| bool | flag (main) | флаг состояния программы (останавливает работу, если был установлен) |
| cols\_changed (defsize) | флаг изменения кол-ва столбцов в двумерном массиве (производится логирование в файл, если был установлен) |
| char | tmp (defsize) | хранение считанного символа при подсчете |
| tmpc (inarr) | хранение считанного символа в посимвольном режиме считывания |
| float | A (main) | указатель на исходный двумерный массив |
| tmpf (inarr) | хранение считанного числа в режиме пропуска разделителей |
| p (process) | указатель на вектор |
| summary (process) | подсчет суммы элементов очередного столбца |

**Описание используемых функций.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Назначение | Параметры | | | |
| входные | выходные | модифицируемые | транзитные |
| defsize | определение реальных размеров двумерного массива |  |  | in, out, est\_rows, est\_cols, fail |  |
| inarr | запись элементов файла в двумерный массив | max\_rows, max\_cols |  | a, in |  |
| outarr | вывод элементов двумерного массива | a, rows, cols |  | out | Переходят в outrow:  a (исходный массив), i (текущая строка), cols (кол-во столбцов в массиве), out (выходной файл) |
| outrow | вывод строки в массиве | a, row, cols |  | out |  |
| process | создание вектора и его вывод в файл | a, rows, cols |  | out | Переходят в outrow:  p (созданный вектор), row (заглушка для outrow), cols (кол-во столбцов в векторе), out (выходной файл) |

**Организация ввода-вывода.**

|  |  |  |  |
| --- | --- | --- | --- |
| Библиотека | Имя | Команда | Назначение |
| ifstream | infile | >> | считывание с файла через поток |
| is\_open | проверка на корректное открытие файла |
| eof | проверка на конец файла (EOF) |
| clear | очистка всех установленных флагов ошибок |
| close | закрытие файла |
| setf | установка флагов, связанных со считыванием |
| unsetf | снятие флагов, связанных со считыванием |
| tellg | возврат текущей позиции курсора |
| seekg | перемещение курсора на указанную позицию |
| ofstream | outfile | << | запись в файл через поток |
| close | закрытие файла |

**Представление алгоритма решения задачи.**

main:

1. Подключение заголовочных файлов с функциями
2. Заход в main, начало программы
3. Открытие входного и выходного файлов
4. Входной файл открылся?
   1. Да:
      1. Объявление переменных, считывание первой строки, логирование в файл
      2. Число строк или число столбцов неположительные?
         1. Да: логирование в файл, завершение программы
         2. Нет:
            1. Запуск defsize
            2. Флаг установлен?

Да: логирование в файл, завершение программы

Нет:

Выделение памяти под двумерный массив

Запуск inarr

Запуск outarr

Запуск process

Освобождение памяти, выделенной под массив

* 1. Нет: логирование в выходной файл

1. Логирование в выходной файл, закрытие файлов, выход с main, завершение программы

defsize:

1. Переключение в режим посимвольного считывания, объявление переменных
2. Считывание в заглушку первого \n символа
3. Определение размеров по правилам
4. Число строк изменилось?
   1. Да: если число корректное, логировать в файл; присваивание нового значения
   2. Нет: логирование в файл
5. Число столбцов изменилось?
   1. Да: если число корректное, логировать в файл
   2. Нет: логирование в файл
6. Размеры корректные?
   1. Да: установка флага ошибки
7. Снятие флагов ошибок с файла
8. Переход на начало набора, завершение функции

inarr:

1. Объявление переменных
2. Прогон по строкам набора следующим образом:
   1. Считывание строки в посимвольном режиме и определение кол-ва элементов в ней
   2. Кол-во элементов = кол-во столбцов?
      1. Да:
         1. Переход на начало строки, перевод в режим пропуска разделителей
         2. Считывание чисел в очередную строку массива
         3. Переход на начало новой строки
3. Повторять, пока не заполнится весь массив

outrow:

1. Поэлементное логирование в файл одной строки двумерного массива

outarr:

1. Построчный вызов функции outrow

process:

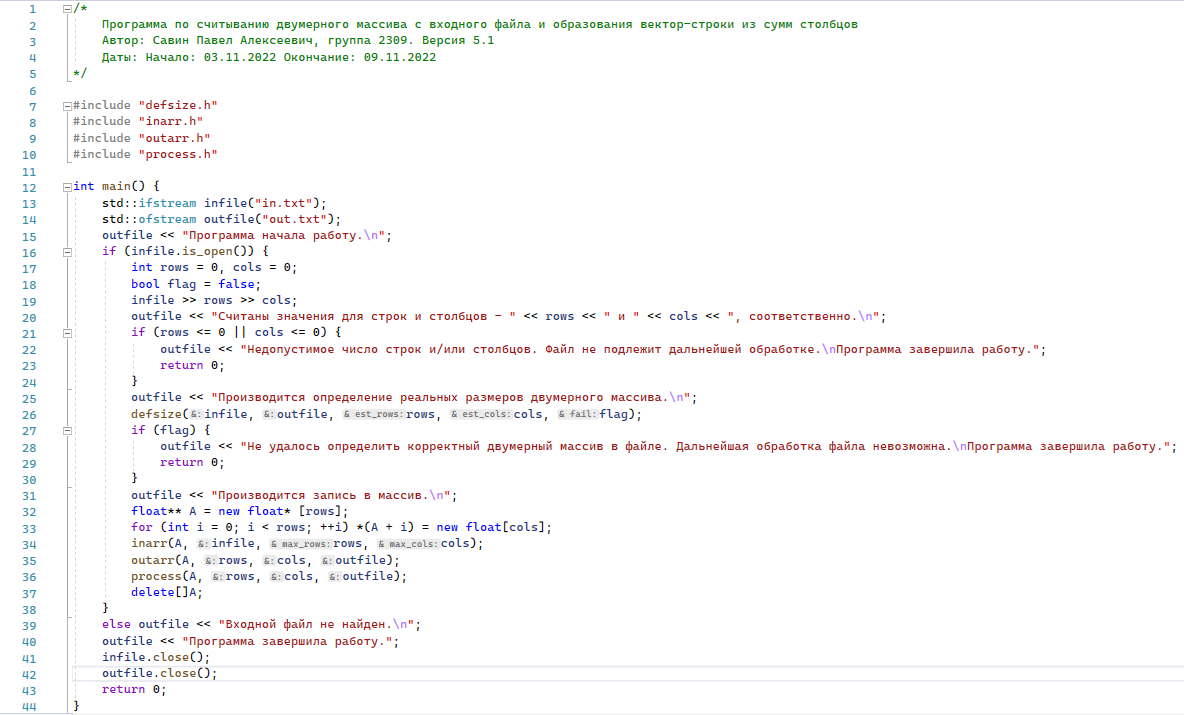
1. Выделение памяти под вектор
2. Постолбцовое суммирование элементов
3. Запись значения в очередной столбец вектора
4. Выделение памяти для заглушки
5. Вызов функции outrow
6. Освобождение памяти, выделенной под заглушку и вектор

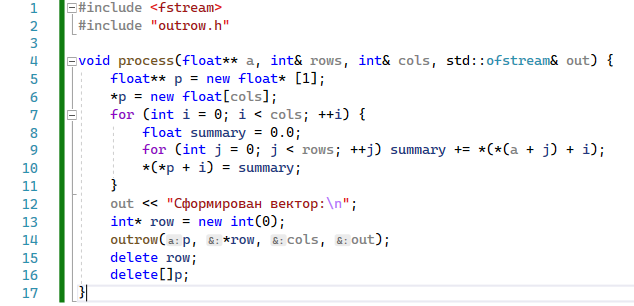
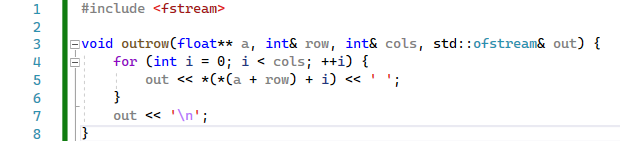
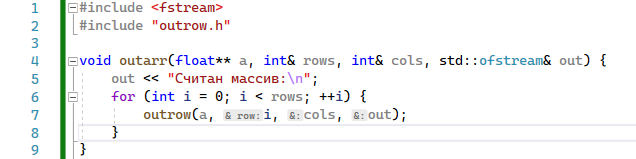
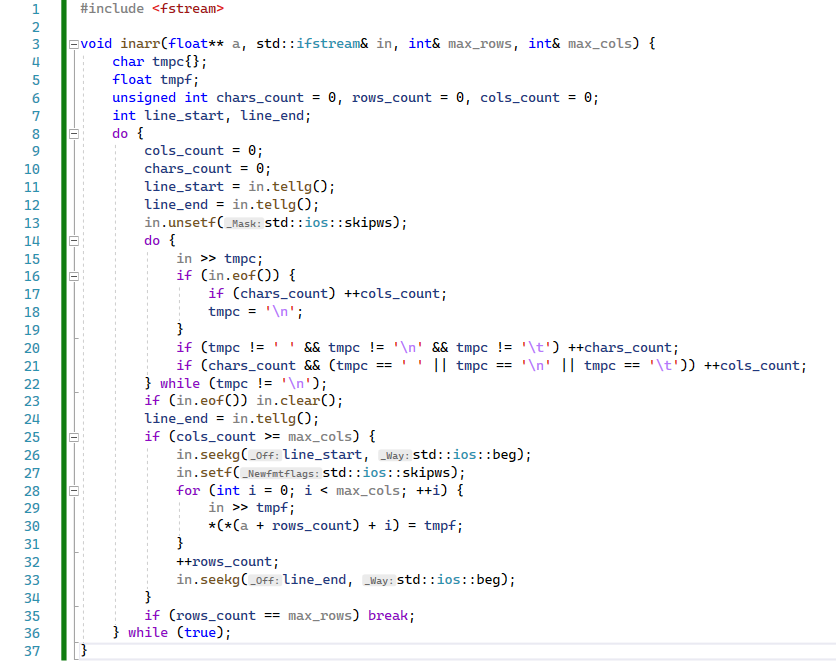
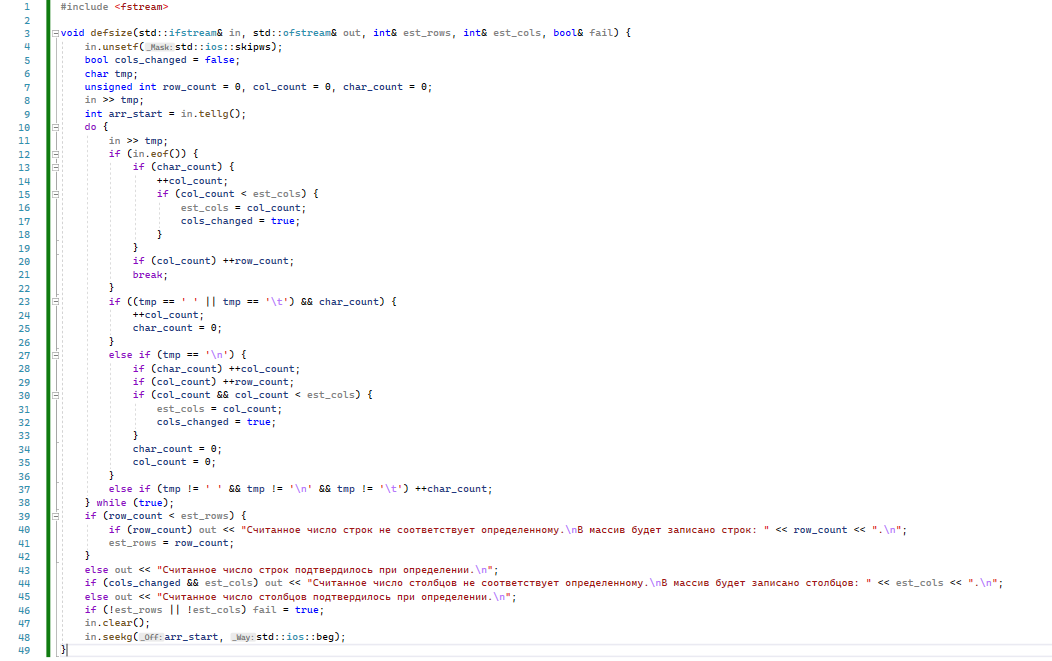
**Блок-схема.**

|  |  |  |
| --- | --- | --- |
| main | defsize | inarr |
| C:\Users\DOM\Desktop\main.png | C:\Users\DOM\Desktop\defsize.png | C:\Users\DOM\Desktop\inarr.png |
| outrow | outarr | process |
| C:\Users\DOM\Desktop\outrow.png | C:\Users\DOM\Desktop\outarr.png | C:\Users\DOM\Desktop\process.png |

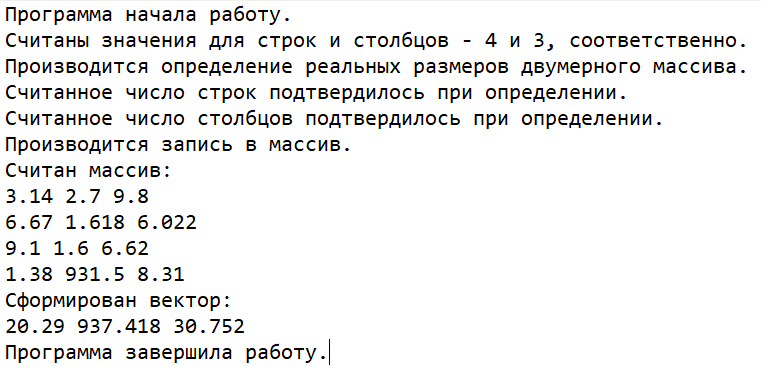
**Текст программы.**

Файлы представлены в порядке вызова (main, defsize, inarr, outarr, outrow, process):

****

****

**Результаты работы программы.**

****

**Вывод о проделанной работе.**

Изучены основные методики при работе с функциями в С++, в том числе и различные способы передачи параметров в функцию.