МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионально образования

“Московский авиационный институт

(национальный исследовательский университет)”

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРИКЛАДНОЙ МАТЕМАТИКИ**

**Кафедра вычислительной математики и программирования**

Курсовой проект по курсу:

“Основы информатики”

II семестр

Задание 7

“Разреженные матрицы”

Работу выполнил:

студент 1 курса

Группы М8О-107Б-2018

Гамов Павел Антонович

Проверила:

асп. каф. 806 Ридли Александра Николаевна.

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2019

Содержание

Введение и постановка задачи……………………………………………..……3

Решение и алгоритм действий………………………………………..…………3

addit.h……………………………………………………………………………...3

main.c……………………………………………………………………………...4

Пример работы и тесты…………………………………………………….…….7

Вывод…………………………………………………………………………...…8

Использованная литература………………………………………………….…..9

**Введение и постановка задачи**

Разреженные матрицы – матрицы содержащие в себе преимущественно нулевые элементы, порой, выделять полный размер для хранения нулевых значений является плохим решением, поэтому существуют различные способы записи таких матриц.

Задача сводится к созданию функций записи и хранения в памяти разреженной матрицы, а также осуществлению операций над ней.

Задача: необходимо реализовать запись матрицы комплексных чисел высокой точности в трех векторах. Найти наиболее приближенный элемент к введенному и разделить на него элементы, стоящие на пересечении горизонталей и вертикалей матрицы. Если таких элементов несколько, обработать все.

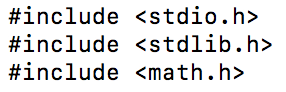
**Решение и алгоритм**

Ввод обычной матрицы, перевод ее в векторную запись, ввод числа, поиск ближайшего к нему посредством функции, считающей расстояние между точками на комплексной плоскости. Пробег по векторам с поиском подходящих элементов, их деление на число ближайшее к заданному.

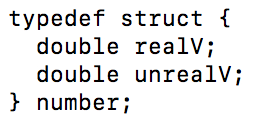
Программа состоит из двух файлов: main.c и addit.h. Рассмотрим их содержание и принцип работы.

**addit.h**

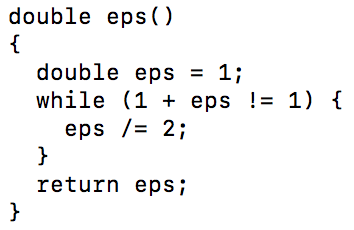
Рассмотрим данный файл, так как он сравнительно небольшой.



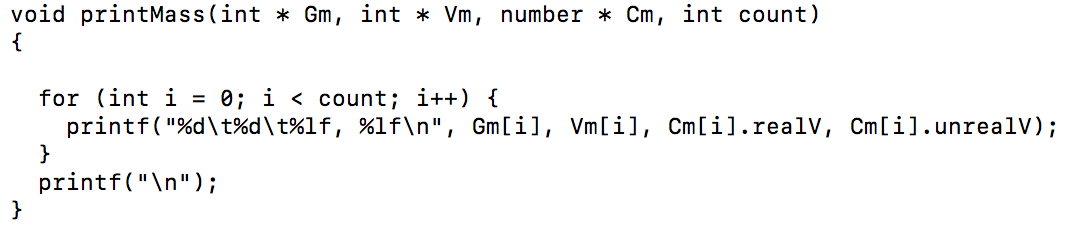
Подключаем math.h ради корней и степеней.



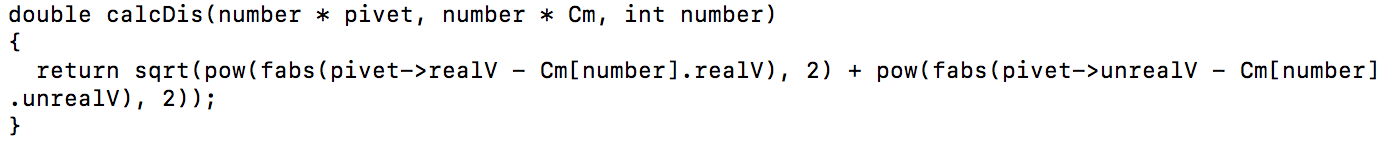
Объявляем структуру для записи комплексных чисел.



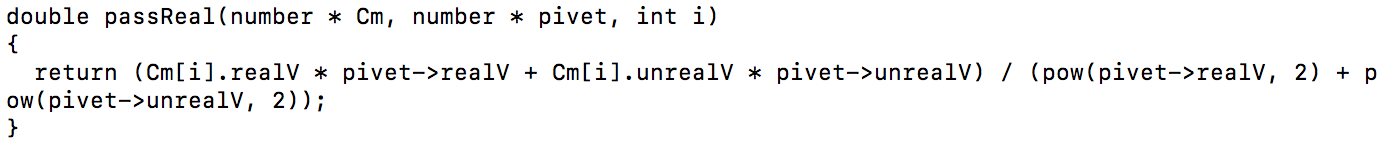
Функция eps() из первого семестра. Нужна для сравнения чисел высокой точности, которые нельзя нормально проверить на совпадение.

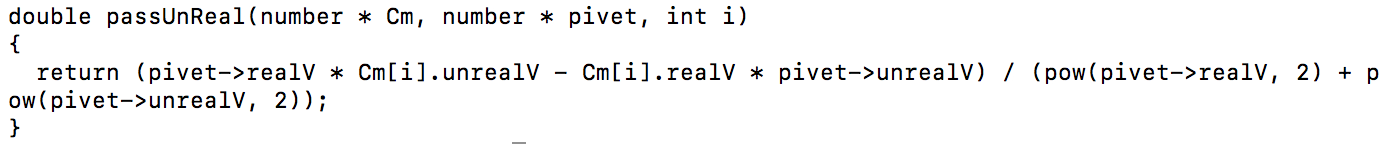


Так будет происходить вывод трех векторов, в которых будут записана наша матрица.



calcDis() считает расстояние от точки до точки на комплексной плоскости.



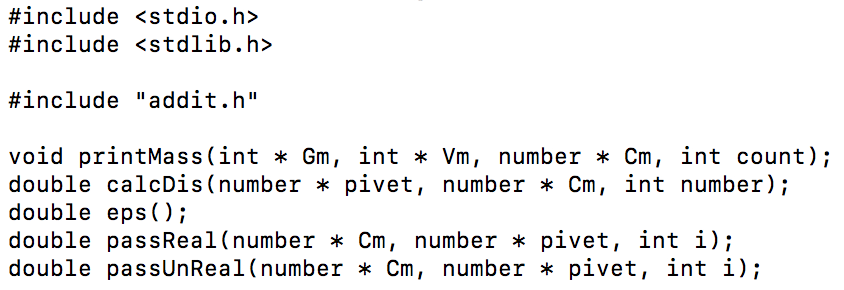


Данные функции принимают на вход массив структур number описанных выше.

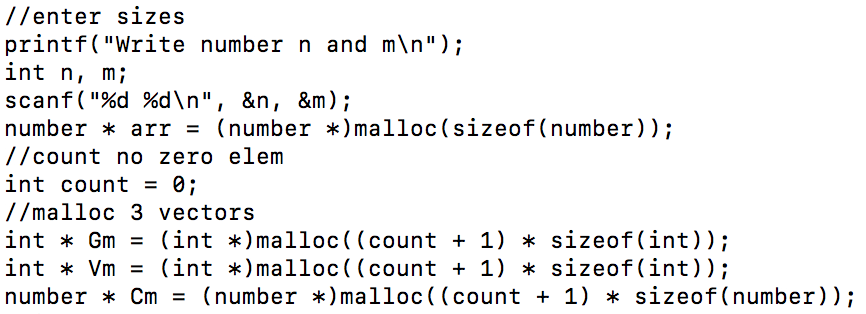
Сложность арифметики деления комплексных чисел подразумевает умножения на сопряженное знаменателя с дальнейшим раскрытием скобок, в итоге получаем в знаменателе действительное число.

**main.c**

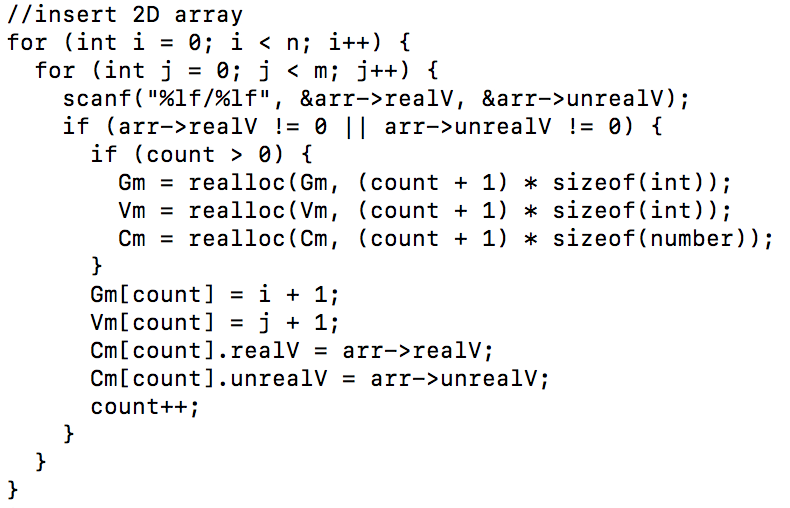
Последовательно рассмотрим содержимое главного файла.



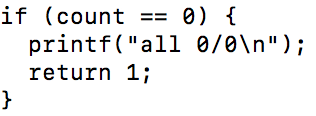
Подключение и определение функций из addit.h.



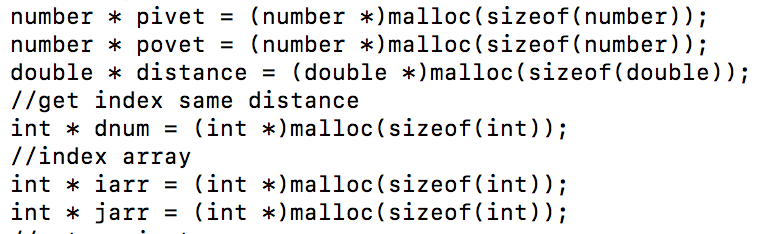
Вводим размеры массива, выделяем память под структуру arr, в которую будем записывать данные и тут же обрабатывать, решая, записывать их в векторы или нет. Ставим счетчик count на ноль и выделяем 3 вектора размера 1.



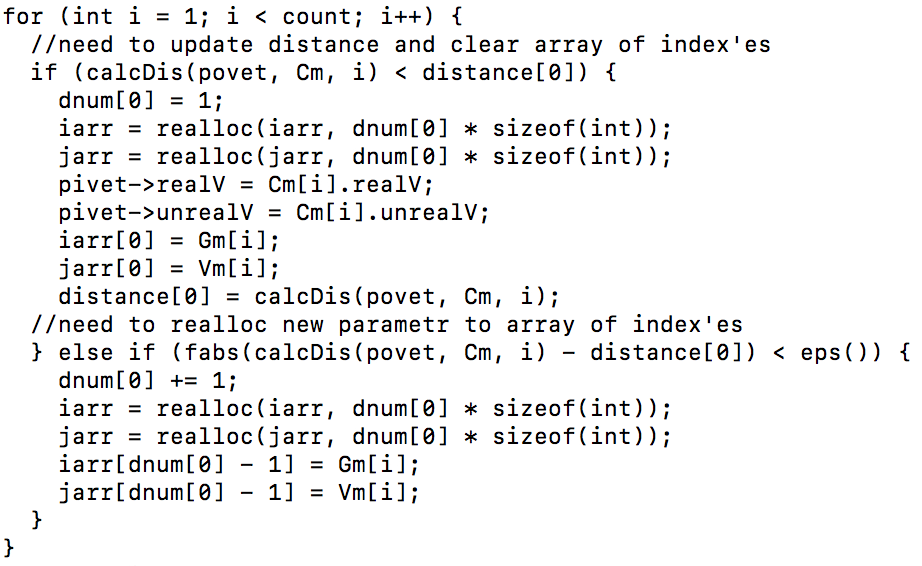
Заполняем arr, делаем проверку на нули, заполняем, если векторы переполнены, выделяем память, поднимаем счетчик.



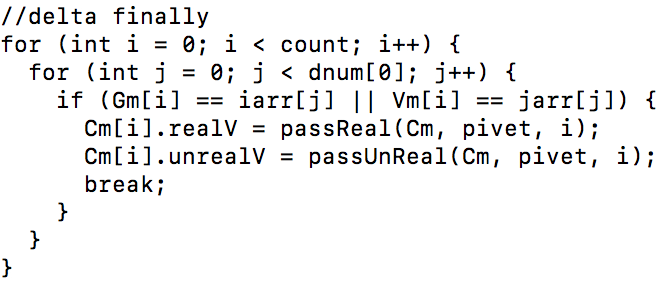
Ошибка при нулевой матрице.



pivet – элемент, который будет ближе к всего к введенному, либо равен ему. povet – элемент, ближайший к которому мы будем искать. iarr и jar – массивы интов, в которых будут находиться индексы элементов равных povet. Так как задание подразумевает найти все элементы ближайшие к данному и обработать все стоки и столбцы. dnum – размер массивов iarr и jarr.



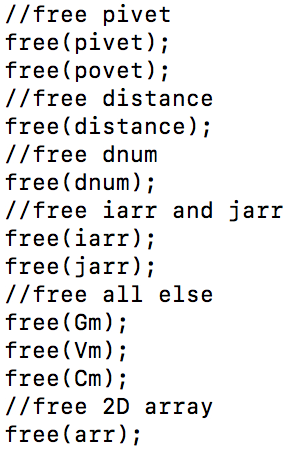
Для длины ненулевых элементов бежим по векторам, считаем расстояние от элемента до povet, если расстояние меньше – записываем индексы в iarr и jarr. Если так вышло что мы нашли меньшее расстояние, а в массивах индексов уже были элементы, те стираются до нуля и начинается перезапись новыми числами. В результате мы получаем 2 вектора с индексом/индексами элемента/элементов, который ближе всего к введенному.



Бежим по векторам матрицы сравнивая индексы элементов в векторах с теми, которые мы записали, если происходит совпадение по хотя бы одному их значений – мы попали на столбец или строку пересечения с нужным нам элементом. Запускаем функцию деления, передавая нужные индексы векторов в функцию.

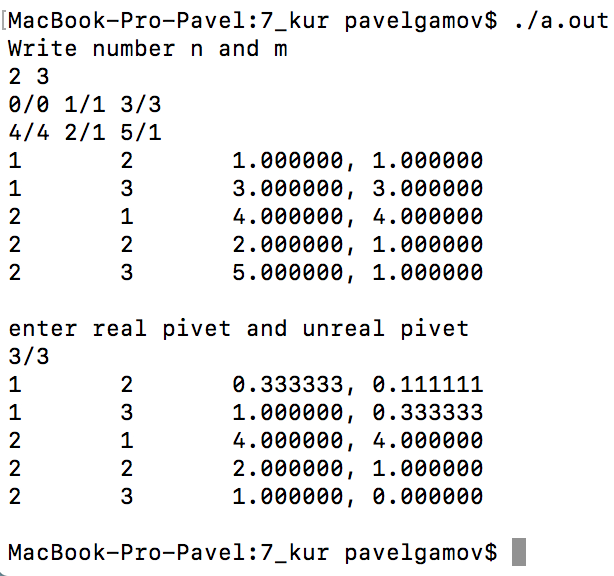


Окончательный вывод массива.



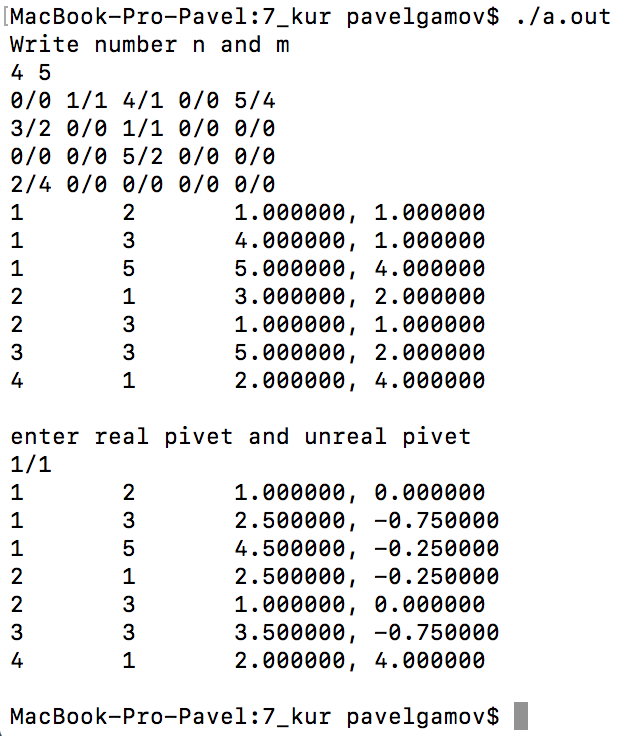
Освобождения памяти выделенной на все динамические структуры и объекты.

**Примеры и тесты**



Ввод чисел происходит через слеш для удобства записи.

После ввода программа печатает массив, после работы также выводит его для того, чтобы увидеть результат.



Пример с обработкой двух одинаково близких к введенному элементов.

Как видно 2/4 под индексами 4 и 1 не изменился, тк не лежит на пересечении горизонталей и вертикалей содержащих povet.

**Заключение**

Способы записи разреженных матриц существуют в разных вариациях и типах, их объединяет существенное сокращение размеров выделяемой памяти при стремлении количества ненулевых элементов к нулю. Также стоит сказать про динамическое выделение памяти в Си. Динамическая память привнесла в язык способность обрабатывать и изменять структуры на ходу программы, что существенно облегчает работу с памятью при использовании различных массивов и структур.

**Список использованной литературы**

Динамическое выделение памяти в Си – <https://prog-cpp.ru/c-alloc/>

Динамическое выделение памяти – <https://learnc.info/c/memory_allocation.html>

Разреженные матрицы в Си – <https://dxdy.ru/topic18364-15.html>