****Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Курсовая работа**

**по курсу «Языки и методы программирования»**

**II Семестр**

**Задание 7**

**Разреженные матрицы**

Группа: М80 – 107Б-18

Студент: Син Денис Дмитриевич

Преподаватель: Ридли Александра Николаевна

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2018.

**Содержание**

1. Постановка задачи
2. Общие сведения о программе
3. Общий метод и алгоритм решения
4. Основные файлы программы
5. Демонстрация работы программы
6. Вывод

**Постановка задачи**.

Составить программу на языке Си с процедурами и/или функциями для обработки прямоугольных разреженных матриц с элементами целого, вещественного или комплексного типов, которая:

1. Вводит матрицы различного размера, представленные во входном текстовом файле в обычном формате, с одновременным размещением ненулевых элементов в разреженной матрице в соответствии с заданной схемой
2. Печатает введенные матрицы во внутреннем представлении согласно заданной схеме размещения и в обычном естественном виде.
3. Выполняет необходимые преобразования разреженных матриц (или вычисления над ними) путем обращения к соответствующим процедурам и/или функциям
4. Печатает результат преобразования согласно заданной схеме размещения и в обычном виде.

**Вариант схемы**: 1. Цепочка ненулевых элементов в векторе A со строчным индексированием.

**Вариант операции**: 3. Найти элемент матрицы, ближайший к заданному значению. Разделить на него элементы строки и столбца на пересечении которых он находится. Если таких элементов несколько, то обработать все.

**Общие сведения о программе**

Дерево проекта:

├── Makefile

├── main.c

├── sparse\_matrix

├── srcs

│   ├── create.c

│   ├── operation.c

│   ├── print.c

│   ├── read.c

│   ├── sparse\_matrix.h

│   └── supporting\_functions.c

├── test.py

├── test\_maps

│   ├── test1

│   ├── test2

│   ├── test3

│   ├── test4

│   ├── test5

│   ├── test6

│   ├── test7

│   └── test8

└── vector

├── vector.c

└── vector.h

3 directories, 20 files

1. Директория vector

Содержит в себе реализованную структуру данных vector для 4 типов данных: int, double, char\*, elem (структура для данного проекта, предназначена для реализации хранения разреженных матриц.

1. Директория srcs
   1. create.c – создание структуры для хранения разреженной матрицы.
   2. operation.c – реализация операции с разреженной матрицы
   3. print.c – печать разреженной матрицы
   4. read.c – считывание разреженной матрицы
   5. supporting\_function.c – дополнительные функции.
   6. sparse\_matrix.h – заголовочный файл для реализации структуры хранения разреженной матрицы и функций, для работы с данной структурой.
2. Директория test\_maps – тестовые файлы с разреженными матрицами
3. main.c – главный файл
4. Makefile – файл для сборки проекта
5. test.py – файл для запуска всех тестовых файлов из директории test\_maps
6. sparse\_matrix – исполняемый файл.

**Общий метод и алгоритм решения**.

Для реализации поставленной задачи необходимо:

1. Реализовать структуру, которая хранит в себе 2 вектора
2. При считывании данных из файла необходимо знать количество элементов в строке, чтобы отвалидировать файл.
3. Каждый элемент обработать функцией atof.
4. Использовать машинное эпсилон, для аккуратной работы с вещественными числами.
5. Последовательно записать данные в необходимый вектор.
6. Проитерироваться по созданной структуре и найти минимальную разницу между заданным числом и элементами матрицы.
7. Реализовать функцию, которая делает необходимое преобразование зная минимальную разницу, найденную на предыдущем шаге.
8. Применить эту функцию столько раз, сколько чисел найдется в матрице. Таким образом получится необходимое преобразование.
9. При печати нужно пройтись по вектору B, и зная ширину матрицы напечатать столько элементов, сколько необходимо.

**Основные файлы программы**.

**srcs/read.c**

#include "sparse\_matrix.h"

static void s\_matrix\_read\_line(vector \*v, s\_matrix \*matrix)

{

    double eps;

    double value;

    int num\_line;

    int prev;

    int indent;

    s\_matrix\_elem \*tmp;

    s\_matrix\_elem m\_elem;

    eps = machine\_eps();

    num\_line = vec\_length(&matrix->A);

    prev = -1;

    indent = (int)vec\_length(&matrix->B);

    for (int i = 0; i < vec\_length(v); ++i) {

        value = \*(double \*)v->data[i].value;

        if (fabs(value) < eps)

            continue;

        if (vec\_length(&matrix->A) == num\_line)

            vec\_push\_back(&matrix->A, INTEGER, i);

        s\_matrix\_elem\_init(&m\_elem, i, value, 0);

        if (prev != -1) {

            tmp = (s\_matrix\_elem \*)matrix->B.data[prev + indent].value;

            tmp->index = i;

        }

        vec\_push\_back(&matrix->B, MATRIX\_ELEM, m\_elem);

        prev++;

    }

    if (vec\_length(&matrix->A) == num\_line)

        vec\_push\_back(&matrix->A, INTEGER, -1);

}

/\* Return -1 if error \*/

int s\_matrix\_read(FILE \*in, s\_matrix \*matrix)

{

double num;

    int state;

char line[BUF\_SIZE];

vector values;

    state = 0;

    bzero(line, BUF\_SIZE);

while (fgets(line, BUF\_SIZE, in) != NULL) {

        line[strlen(line) - 1] = '\0';

values = strsplit(line, ' ');

        if (state++ == 0)

            matrix->width = vec\_length(&values);

        else if (matrix->width != vec\_length(&values))

            return -1;

        vec\_str\_to\_double(&values);

        s\_matrix\_read\_line(&values, matrix);

        vec\_destroy(&values);

        matrix->height++;

}

    if (state == 0)

        return -1;

    return 1;

}

**srcs/create.c**

#include "sparse\_matrix.h"

s\_matrix s\_matrix\_create(void)

{

s\_matrix matrix;

matrix.A = vec\_create(10);

matrix.B = vec\_create(10);

    matrix.width = 0;

    matrix.height = 0;

return matrix;

}

void s\_matrix\_elem\_init(s\_matrix\_elem \*sm\_elem, size\_t column\_num, double value, size\_t index)

{

    sm\_elem->column\_num = column\_num;

    sm\_elem->value = value;

    sm\_elem->index = index;

}

**srcs/print.c**

#include "sparse\_matrix.h"

static void zero\_print(FILE \*out, int num)

{

    for (int i = 0; i < num; ++i)

        fprintf(out, "%10d", 0);

}

void s\_matrix\_scheme\_print(FILE \*out, s\_matrix \*matrix)

{

    vec\_print(out, &matrix->A);

    fprintf(out, "\n");

    vec\_print(out, &matrix->B);

    fprintf(out, "\n");

}

void s\_matrix\_print(FILE \*out, s\_matrix \*matrix)

{

    int iter;

    int delta;

    s\_matrix\_elem tmp;

    iter = 0;

    for (int i = 0; i < matrix->height; ++i) {

        if (\*(int \*)matrix->A.data[i].value == -1) {

            zero\_print(out, matrix->width);

            fprintf(out, "\n");

            continue;

        }

        zero\_print(out, \*(int \*)matrix->A.data[i].value);

        tmp = \*(s\_matrix\_elem \*)matrix->B.data[iter].value;

        iter++;

        fprintf(out, "%10f", tmp.value);

        while (tmp.index != 0) {

            delta = tmp.column\_num;

            tmp = \*(s\_matrix\_elem \*)matrix->B.data[iter].value;

            iter++;

            delta = tmp.column\_num - delta - 1;

            zero\_print(out, delta);

            fprintf(out, "%10f", tmp.value);

        }

        zero\_print(out, matrix->width - tmp.column\_num - 1);

        fprintf(out, "\n");

    }

}

**srcs/operation.c**

#include "sparse\_matrix.h"

static void divide(s\_matrix \*matrix, double value, int line, int column)

{

    s\_matrix\_elem \*tmp;

    int num\_line;

    num\_line = 0;

    for (int i = 0; i < vec\_length(&matrix->B); ++i) {

        tmp = (s\_matrix\_elem \*)matrix->B.data[i].value;

        if (tmp->column\_num == column || num\_line == line) {

            tmp->value /= value;

        }

        if (tmp->index == 0)

            num\_line++;

    }

}

static double get\_delta(s\_matrix \*matrix, double value)

{

    double delta;

    s\_matrix\_elem \*tmp;

    tmp = (s\_matrix\_elem \*)matrix->B.data[0].value;

    delta = fabs(value - tmp->value);

    for (int i = 1; i < vec\_length(&matrix->B); ++i) {

        tmp = (s\_matrix\_elem \*)matrix->B.data[i].value;

        if (fabs(value - tmp->value) < delta)

            delta = fabs(value - tmp->value);

    }

    return delta;

}

static vector get\_positions(s\_matrix \*matrix, double value)

{

    vector positions;

    s\_matrix\_elem \*tmp;

    double delta;

    double eps;

    int line;

    eps = machine\_eps();

    delta = get\_delta(matrix, value);

    positions = vec\_create(10);

    line = 0;

    for (int i = 0; i < vec\_length(&matrix->B); ++i) {

        tmp = (s\_matrix\_elem \*)matrix->B.data[i].value;

        if (fabs(fabs(value - tmp->value) - delta) < eps) {

            vec\_push\_back(&positions, INTEGER, line);

            vec\_push\_back(&positions, INTEGER, tmp->column\_num);

        }

        if (tmp->index == 0)

            line++;

    }

    return positions;

}

static vector get\_values(s\_matrix \*matrix, double value)

{

    vector values;

    double delta;

    double eps;

    s\_matrix\_elem \*tmp;

    eps = machine\_eps();

    delta = get\_delta(matrix, value);

    values = vec\_create(10);

    for (int i = 0; i < vec\_length(&matrix->B); ++i) {

        tmp = (s\_matrix\_elem\*)matrix->B.data[i].value;

        if (fabs(fabs(value - tmp->value) - delta) < eps)

            vec\_push\_back(&values, DOUBLE, tmp->value);

    }

    return values;

}

vector s\_matrix\_operation(s\_matrix \*matrix, double value)

{

    vector values;

    vector positions;

    s\_matrix\_elem \*tmp;

    int n;

    int m;

    double search\_value;

    values = get\_values(matrix, value);

    positions = get\_positions(matrix, value);

    for (int i = 0; i < vec\_length(&positions); i += 2) {

        search\_value = \*(double \*)values.data[i / 2].value;

        m = \*(int \*)positions.data[i].value;

        n = \*(int \*)positions.data[i + 1].value;

        divide(matrix, search\_value, m, n);

    }

    return values;

}

**srcs/supporting\_functions.c**

#include "sparse\_matrix.h"

double machine\_eps(void)

{

    double eps;

    eps = 1;

    while (1 + eps > 1)

        eps /= 2;

return eps;

}

/\* Split string cross delimetr \*/

vector strsplit(char \*str, char delim)

{

char buf[1024];

vector ans;

int j;

j = 0;

ans = vec\_create(10);

for (size\_t i = 0; i < strlen(str); ++i) {

if (str[i] == delim) {

if (j != 0) {

buf[i] = '\0';

vec\_push\_back(&ans, STRING, buf);

bzero(buf, 1024);

j = 0;

}

} else {

buf[j] = str[i];

j++;

}

    }

    vec\_push\_back(&ans, STRING, buf);

    return ans;

}

void vec\_str\_to\_double(vector \*v)

{

    double num;

    for (int i = 0; i < vec\_length(v); ++i) {

        num = atof((char \*)v->data[i].value);

        vec\_set\_value(v, i, DOUBLE, num);

    }

}

**srcs/sparse\_matrix.h**

#ifndef SPARSE\_MATRIX\_H

#define SPARSE\_MATRIX\_H

#include "../vector/vector.h"

#include <math.h>

#define BUF\_SIZE 10000

typedef struct {

    vector A;

    vector B;

    int width;

    int height;

} s\_matrix;

void        s\_matrix\_elem\_init(s\_matrix\_elem \*sm\_elem, size\_t column\_num, double value, size\_t index);

s\_matrix s\_matrix\_create(void);

int s\_matrix\_read(FILE \*in, s\_matrix \*matrix);

void s\_matrix\_print(FILE \*out, s\_matrix \*matrix);

void        s\_matrix\_scheme\_print(FILE \*out, s\_matrix \*matrix);

vector s\_matrix\_operation(s\_matrix \*matrix, double value);

/\*Supporting functions\*/

double      machine\_eps(void);

vector      strsplit(char \*str, char delim);

void        vec\_str\_to\_double(vector \*v);

#endif

**main.c**

#include "srcs/sparse\_matrix.h"

#include <stdio.h>

int main(int argc, char \*argv[])

{

    s\_matrix matrix;

    vector values;

    FILE \*fin;

    matrix = s\_matrix\_create();

    fprintf(stdout, "%s\n", argv[1]);

    if (argc != 3) {

        fprintf(stdout, "Usage: ./sparse\_matrix filename value\n");

        exit(1);

    }

   fin = fopen(argv[1], "r");

    if (fin == NULL) {

        fprintf(stdout, "File does not exist\n");

        exit(1);

    }

    if (s\_matrix\_read(fin, &matrix) == -1) {

        fprintf(stdout, "Not valid matrix\n");

        exit(1);

    }

    fprintf(stdout, "Original matrix\n");

    s\_matrix\_print(stdout, &matrix);

    fprintf(stdout, "\n");

    fprintf(stdout, "Original Scheme of matrix\n");

    s\_matrix\_scheme\_print(stdout, &matrix);

    fprintf(stdout, "\n");

    fprintf(stdout, "Processed matrix\n");

    values = s\_matrix\_operation(&matrix, atof(argv[2]));

    s\_matrix\_print(stdout, &matrix);

    fprintf(stdout, "\n");

    fprintf(stdout, "Processed scheme of matrix\n");

    s\_matrix\_scheme\_print(stdout, &matrix);

    fprintf(stdout, "\n");

    fprintf(stdout, "Processed values:\n");

    vec\_print(stdout, &values);

    fprintf(stdout, "\n\n");

    return 0;

}

**Makefile**

CC = gcc

NAME = sparse\_matrix

FLAGS = -Wall -Wextra -Werror

VECTOR = vector/vector.c

SRC\_DIR = srcs

SRC = create.c\

     read.c\

     print.c\

     operation.c\

     supporting\_functions.c\

SRCS = $(addprefix $(SRC\_DIR)/, $(SRC))

all:

    @$(CC) $(KEYS) main.c $(SRCS) $(VECTOR) -o $(NAME)

clean:

    @rm -rf $(NAME)

fclean: clean

re: fclean all

**test.py**

import os

test\_dir = "test\_maps/"

value = 3.4

files = os.listdir(test\_dir)

for file in files:

os.system("./sparse\_matrix {} {}".format(test\_dir + file, value))

**Демонстрация работы программы.**

➜ KP2 ls

Makefile main.c sparse\_matrix srcs test.py test\_maps vector

➜ KP2 rm -rf sparse\_matrix

➜ KP2 make

➜ KP2 ls

Makefile main.c sparse\_matrix srcs test.py test\_maps vector

➜ KP2 ./sparse\_matrix hello

hello

Usage: ./sparse\_matrix filename value

➜ KP2 ./sparse\_matrix fds 23

fds

File does not exist

➜ KP2 cd test\_maps

➜ test\_maps cat test1

1.2 0 0 0 0 1 0 0 3

1.3 0 0 1 0 1 0 0 2

1.4 0 0 0 0 1 0 0 6

1.4 0 0 3 2 0 0 0 3

➜ test\_maps cat test2

0 0 0 3.4 0 0 5 0 6 0 0 0 8

0 0 0 0 0 0 0 0 0 0 4 0 0% ➜ test\_maps cat test3

12 32 43

34 4 2 3

➜ test\_maps cat test4

➜ test\_maps cat test5

1

0

0

0

0

0

1

0

0

3

4

5

➜ test\_maps cat test6

1 0 0 0 2 3 -4.5 0 -3 -0.4 0 0 0 0 0 1 0 0 0 2 3 -4.5 0 -3 0

➜ test\_maps cat test7

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

0 0 0 0 3.4 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 1.5 0 0 0

23.43 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

0 0 0 732.45 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 3.4 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

1.2 0 0 0 0 0 0 0 0 0 0 0 1.3 1.4 0 0 0 0 0 -1.5 0 0 0

➜ test\_maps cat test8

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 5 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0

➜ test\_maps cd ..

➜ KP2 ls

Makefile main.c sparse\_matrix srcs test.py test\_maps vector

➜ KP2 ./sparse\_matrix test\_maps/test1

test\_maps/test1

Usage: ./sparse\_matrix filename value

➜ KP2 ./sparse\_matrix test\_maps/test1 0.4

test\_maps/test1

Original matrix

1.200000 0 0 0 0 1.000000 0 0 3.000000

1.300000 0 0 1.000000 0 1.000000 0 0 2.000000

1.400000 0 0 0 0 1.000000 0 0 6.000000

1.400000 0 0 3.000000 2.000000 0 0 0 3.000000

Original Scheme of matrix

0 0 0 0

0:1.200000:5 5:1.000000:8 8:3.000000:0 0:1.300000:3 3:1.000000:5 5:1.000000:8 8:2.000000:0 0:1.400000:5 5:1.000000:8 8:6.000000:0 0:1.400000:3 3:3.000000:4 4:2.000000:8 8:3.000000:0

Processed matrix

1.200000 0 0 0 0 1.000000 0 0 3.000000

1.300000 0 0 1.000000 0 1.000000 0 0 2.000000

1.400000 0 0 0 0 1.000000 0 0 6.000000

1.400000 0 0 3.000000 2.000000 0 0 0 3.000000

Processed scheme of matrix

0 0 0 0

0:1.200000:5 5:1.000000:8 8:3.000000:0 0:1.300000:3 3:1.000000:5 5:1.000000:8 8:2.000000:0 0:1.400000:5 5:1.000000:8 8:6.000000:0 0:1.400000:3 3:3.000000:4 4:2.000000:8 8:3.000000:0

Processed values:

1.000000 1.000000 1.000000 1.000000

➜ KP2 ./sparse\_matrix test\_maps/test1 1.8

test\_maps/test1

Original matrix

1.200000 0 0 0 0 1.000000 0 0 3.000000

1.300000 0 0 1.000000 0 1.000000 0 0 2.000000

1.400000 0 0 0 0 1.000000 0 0 6.000000

1.400000 0 0 3.000000 2.000000 0 0 0 3.000000

Original Scheme of matrix

0 0 0 0

0:1.200000:5 5:1.000000:8 8:3.000000:0 0:1.300000:3 3:1.000000:5 5:1.000000:8 8:2.000000:0 0:1.400000:5 5:1.000000:8 8:6.000000:0 0:1.400000:3 3:3.000000:4 4:2.000000:8 8:3.000000:0

Processed matrix

1.200000 0 0 0 0 1.000000 0 0 1.500000

0.650000 0 0 0.500000 0 0.500000 0 0 1.000000

1.400000 0 0 0 0 1.000000 0 0 3.000000

0.700000 0 0 1.500000 1.000000 0 0 0 0.750000

Processed scheme of matrix

0 0 0 0

0:1.200000:5 5:1.000000:8 8:1.500000:0 0:0.650000:3 3:0.500000:5 5:0.500000:8 8:1.000000:0 0:1.400000:5 5:1.000000:8 8:3.000000:0 0:0.700000:3 3:1.500000:4 4:1.000000:8 8:0.750000:0

Processed values:

2.000000 2.000000

➜ KP2 ./sparse\_matrix test\_maps/test1 3.5

test\_maps/test1

Original matrix

1.200000 0 0 0 0 1.000000 0 0 3.000000

1.300000 0 0 1.000000 0 1.000000 0 0 2.000000

1.400000 0 0 0 0 1.000000 0 0 6.000000

1.400000 0 0 3.000000 2.000000 0 0 0 3.000000

Original Scheme of matrix

0 0 0 0

0:1.200000:5 5:1.000000:8 8:3.000000:0 0:1.300000:3 3:1.000000:5 5:1.000000:8 8:2.000000:0 0:1.400000:5 5:1.000000:8 8:6.000000:0 0:1.400000:3 3:3.000000:4 4:2.000000:8 8:3.000000:0

Processed matrix

0.400000 0 0 0 0 0.333333 0 0 0.333333

1.300000 0 0 0.333333 0 1.000000 0 0 0.222222

1.400000 0 0 0 0 1.000000 0 0 0.666667

0.155556 0 0 0.333333 0.222222 0 0 0 0.111111

Processed scheme of matrix

0 0 0 0

0:0.400000:5 5:0.333333:8 8:0.333333:0 0:1.300000:3 3:0.333333:5 5:1.000000:8 8:0.222222:0 0:1.400000:5 5:1.000000:8 8:0.666667:0 0:0.155556:3 3:0.333333:4 4:0.222222:8 8:0.111111:0

Processed values:

3.000000 3.000000 3.000000

➜ KP2 ./sparse\_matrix test\_maps/test2 3

test\_maps/test2

Original matrix

0 0 0 3.400000 0 0 5.000000 0 6.000000 0 0 0 8.000000

0 0 0 0 0 0 0 0 0 0 4.000000 0 0

Original Scheme of matrix

3 10

3:3.400000:6 6:5.000000:8 8:6.000000:12 12:8.000000:0 10:4.000000:0

Processed matrix

0 0 0 1.000000 0 0 1.470588 0 1.764706 0 0 0 2.352941

0 0 0 0 0 0 0 0 0 0 4.000000 0 0

Processed scheme of matrix

3 10

3:1.000000:6 6:1.470588:8 8:1.764706:12 12:2.352941:0 10:4.000000:0

Processed values:

3.400000

➜ KP2 ./sparse\_matrix test\_maps/test4 3.4

test\_maps/test4

Not valid matrix

➜ KP2 ./sparse\_matrix test\_maps/test5 3.4

test\_maps/test5

Original matrix

1.000000

0

0

0

0

0

1.000000

0

0

3.000000

4.000000

5.000000

Original Scheme of matrix

0 -1 -1 -1 -1 -1 0 -1 -1 0 0 0

0:1.000000:0 0:1.000000:0 0:3.000000:0 0:4.000000:0 0:5.000000:0

Processed matrix

0.333333

0

0

0

0

0

0.333333

0

0

1.000000

1.333333

1.666667

Processed scheme of matrix

0 -1 -1 -1 -1 -1 0 -1 -1 0 0 0

0:0.333333:0 0:0.333333:0 0:1.000000:0 0:1.333333:0 0:1.666667:0

Processed values:

3.000000

➜ KP2 ./sparse\_matrix test\_maps/test6 3.4

test\_maps/test6

Original matrix

1.000000 0 0 0 2.000000 3.000000 -4.500000 0 -3.000000 -0.400000 0 0 0 0 0 1.000000 0 0 0 2.000000 3.000000 -4.500000 0 -3.000000 0

Original Scheme of matrix

0

0:1.000000:4 4:2.000000:5 5:3.000000:6 6:-4.500000:8 8:-3.000000:9 9:-0.400000:15 15:1.000000:19 19:2.000000:20 20:3.000000:21 21:-4.500000:23 23:-3.000000:0

Processed matrix

0.111111 0 0 0 0.222222 0.333333 -0.500000 0 -0.333333 -0.044444 0 0 0 0 0 0.111111 0 0 0 0.222222 0.333333 -0.500000 0 -0.333333 0

Processed scheme of matrix

0

0:0.111111:4 4:0.222222:5 5:0.333333:6 6:-0.500000:8 8:-0.333333:9 9:-0.044444:15 15:0.111111:19 19:0.222222:20 20:0.333333:21 21:-0.500000:23 23:-0.333333:0

Processed values:

3.000000 3.000000

➜ KP2 ./sparse\_matrix test\_maps/test7 3.4

test\_maps/test7

Original matrix

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 3.400000 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 1.500000 0 0 0

23.430000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0732.450000 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 3.400000 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

Original Scheme of matrix

0 12 0 4 0 0 0 12 0 3 0 0 12 0 0

0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 4:3.400000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:1.500000:0 0:23.430000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 3:732.450000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:4 4:3.400000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0

Processed matrix

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0.294118 0 0 0 0 0 0 0 0.382353 0.411765 0 0 0 0 0 -0.441176 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 1.500000 0 0 0

23.430000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0732.450000 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0.352941 0 0 0 0.294118 0 0 0 0 0 0 0 0.382353 0.411765 0 0 0 0 0 -0.441176 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

0 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

1.200000 0 0 0 0 0 0 0 0 0 0 0 1.300000 1.400000 0 0 0 0 0 -1.500000 0 0 0

Processed scheme of matrix

0 12 0 4 0 0 0 12 0 3 0 0 12 0 0

0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 4:0.294118:12 12:0.382353:13 13:0.411765:19 19:-0.441176:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:1.500000:0 0:23.430000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 3:732.450000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:0.352941:4 4:0.294118:12 12:0.382353:13 13:0.411765:19 19:-0.441176:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0 0:1.200000:12 12:1.300000:13 13:1.400000:19 19:-1.500000:0

Processed values:

3.400000 3.400000

**Вывод**

В данной курсовой работе я реализовал структуру для хранения разреженных матриц. Полезно пользоваться такой структурой, для работы с большими массивами данных, где необходимо работать с разреженными матрицами, так как хранить в памяти большое количество нулей не имеет смысла. В итоге получилась полностью работающая программа, которая может обрабатывать разреженные матрицы.