Московский авиационный институт
(Национальный исследовательский университет)

Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Задание VIII по курсовому проекту

«Линейные списки»

Руководитель: Никулин С.П.

_____
(оценка)

_____ _____
(подпись) (дата)

Студент группы 8О-104Б: Дюсекеев Алишер

Москва 2018

Задание:

Составить программу на языке Си с процедурами и функциями для обработки прямоугольных разреженных матриц с элементами вещественного типа, которая:

1. Вводит матрицы различного размера с одновременным размещением ненулевых элементов в разреженной матрице в соответствии с заданной схемой.

2. Печатает введенные матрицы во внутреннем представлении и в обычном виде;

3. Выполняет необходимые преобразования разреженных матриц (или вычисления над ними) путем обращения к соответствующим функциям;

4. Печатает результат преобразования во внутреннем представлении и в обычном виде.

В процедурах и функциях предусмотреть проверки и печать сообщений в случаях ошибок в задании параметров. Для отладки использовать матрицы, содержащие 5-10% ненулевых элементов.

Вариант схемы размещения: 2. Один вектор

Вариант физического представления: 1. Отображение на массив.

Таблица переменных и констант:

Название переменной	**Тип переменной**	**Назначение переменной**
str	Int	Количество строк матрицы
Col	Int	Количество столбцов матрицы
I, j, r, z, c, m, k	Int	Счётчики циклов (в т.ч. вложенных)
*arr	Float	Динамический массив внутреннего представления матрицы
X	Float	Чтение элементов матрицы из файла
Mtx	FILE	Текстовый файл с матрицей
n	Int	Размер массива необходимый для хранения матрицы
sym	int	Проверка на кососимметричность (0 – нет, >0 – да)

Дополнительные функции:

```
   1) void printarr(float *arr, int k);
   Вывод массива внутреннего представления матрицы на экран.
   2) void printmatrix(float *arr, int str, int col);
Вывод матрицы на экран.
   3) void conj(float *arr, int str, int col);
Транспонирование матрицы.
```

Входные данные

Входные данные в программу поступают из текстовых файлов, содержащих прямоугольные матрицы с указанием их размера. Элементы таких матриц разделены табуляциями, а строки – символом переноса строки. Имена файлов поступают в качестве аргументов командной строки. В моём случае это matrix1.txt, matrix2.txt и matrix3.txt.

Выходные данные

После обработки полученной матрицы, программа выводит на экран массив, представляющий во внутренней памяти исходную матрицу, затем саму исходную матрицу, затем массив представления преобразованной матрицы, затем саму преобразованную матрицу. В конце программа выводит результат проверки на кососимметричность (skew symmetry).

Программа на Си:

Main.c

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>

void printarr(float *arr, int k);
void printmatrix(float *arr, int str, int col);
void conj(float *arr, int str, int col);
void main(int argc, char *argv[]) {
	int str, col, n;
	int i = 0, j = 0;
	float *arr, x;
	if (argc != 2)
		printf("usage: matrix filename \n");
	FILE *mtx = fopen(argv[1], "r");
	fscanf(mtx, "%d\t", &str);
	fscanf(mtx, "%d\t", &col);
	n = str*col; // приблизительная формула расчёта необходимой памяти ( [2*(K+N+1) ]
)
	arr = (float *)malloc(n * sizeof(float));
	int k = 0; // счётчик для массива
	for (i = 0; i < str; i++) {
```

```
                arr[k++] = 0.00;
                arr[k++] = (float)(i + 1);
                for (j = 1; j <= col; j++) {
                        fscanf(mtx, "%f\t", &x);
                        if (x != 0.00) {
                                arr[k++] = (float)j;
                                arr[k++] = x;
                        }
                }
        }
        arr[k++] = 0.00;
        arr[k++] = 0.00;
        printf("Array of source matrix \n \n");
        printarr(arr, k);
        printf("\n \n Source matrix \n");
        printmatrix(arr, str, col);
        printf("\n");
        conj(arr, str, col);
        free(arr);
        fclose(mtx);
}

void printarr(float *arr, int k) {
        for (int m = 0; m < k; m++)
                printf("%6.2f", arr[m]);
}

void printmatrix(float *arr, int str, int col) {
        int r, z, c, m;
        int k = 0;
        for (z = 0; z < str; z++) {
                for (r = 0; r < col;) {
                        c = arr[k + 2] - arr[k] - 1;
                        k += 2;
                        if (c > 0)
                                for (m = 0; m < c; m++) {
                                        printf("%6.2f\t", 0.00);
                                        r++;
                                }
                        if (arr[k] != 0.00) {
                                printf("%6.2f\t", arr[k + 1]);
                                r++;
                        }
                        if (arr[k + 2] == 0.00 && arr[k] < col) {
                                c = col - arr[k];
                                for (m = 0; m < c; m++) {
                                        printf("%6.2f\t", 0.00);
                                        r++;
                                }
                        }


                }
                printf("\n");

        }
}

void conj(float *arr, int str, int col) {
        int m = 0, sym = 0;
        int i, k, r;
        int n = str * col;
        float *newarr;
        newarr = (float *)malloc(n * sizeof(float));
        for (i = 1; i <= col; i++) {
```

```
                newarr[m++] = 0.00;
                newarr[m++] = (float)i;
                for (k = 0, r = 0; arr[k + 1] != 0.00; k += 2) {
                        if (arr[k] == 0.00)
                                r++;
                        if (arr[k] == (float)i) {
                                newarr[m++] = (float)r;
                                newarr[m++] = arr[k + 1];
                        }
                        if (arr[k] == r)
                                sym++;
                }
        }
        newarr[m++] = 0.00;
        newarr[m++] = 0.00;
        printf("Array of conjugated matrix \n \n");
        printarr(newarr, m);
        printf("\n \n Conjugated matrix \n");
        printmatrix(newarr, col, str);
        printf("\n");
        if (str == col)
                if (sym != 0)
                        printf("Matrix is not skew symmetric \n");
                else {
                        for (k = 1; arr[k] != 0; k += 2) {
                                if (arr[k - 1] != 0 && (arr[k] - newarr[k] != 0.00)) {
                                        printf("Matrix is not skew symmetric \n");
                                        break;
                                }
                                printf("Matrix is skew symmetric \n");
                                break;
                        }

                }
        else
                printf("Matrix is not skew symmetric \n");


}
```

Демонстрация работы программы:

Argv[1] == “matrix1.txt”

```
        Array of source matrix


        0.00   1.00   1.00   1.11   5.00   2.22   0.00   2.00   3.00   3.33   0.00
  3.00  0.00
        4.00  1.00  4.44  2.00  5.55  6.00  6.66  0.00  0.00


        Source matrix
        1.11    0.00    0.00    0.00    2.22    0.00
        0.00    0.00    3.33    0.00    0.00    0.00
        0.00    0.00    0.00    0.00    0.00    0.00
        4.44    5.55    0.00    0.00    0.00    6.66
```

Array of conjugated matrix

0.00 1.00 1.00 1.11 4.00 4.44 0.00 2.00 4.00 5.55 0.00 3.00 2.00

3.33 0.00 4.00 0.00 5.00 1.00 2.22 0.00 6.00 4.00 6.66 0.00 0.00

Conjugated matrix

1.11 0.00 0.00 4.44

0.00 0.00 0.00 5.55

0.00 3.33 0.00 0.00

0.00 0.00 0.00 0.00

2.22 0.00 0.00 0.00

0.00 0.00 0.00 6.66

Matrix is not skew symmetric

Argv[1]== “Matrix3.txt”

Array of source matrix

0.00 1.00 2.00 -1.00 0.00 2.00 1.00 1.00 3.00 -2.25 0.00 3.00 2.00

2.25 5.00 -9.99 0.00 4.00 0.00 5.00 3.00 9.99 0.00 0.00

Source matrix

0.00 -1.00 0.00 0.00 0.00

1.00 0.00 -2.25 0.00 0.00

0.00 2.25 0.00 0.00 -9.99

0.00 0.00 0.00 0.00 0.00

0.00 0.00 9.99 0.00 0.00

Array of conjugated matrix

0.00 1.00 2.00 1.00 0.00 2.00 1.00 -1.00 3.00 2.25 0.00 3.00 2.00 -

2.25 5.00 9.99 0.00 4.00 0.00 5.00 3.00 -9.99 0.00 0.00

Conjugated matrix

0.00 1.00 0.00 0.00 0.00

```
 -1.00    0.00    2.25    0.00    0.00
  0.00   -2.25    0.00    0.00    9.99
  0.00    0.00    0.00    0.00    0.00
  0.00    0.00   -9.99    0.00    0.00

Matrix is skew symmetric
Для продолжения нажмите любую клавишу . . .
```

Argv[1] == “matrix2.txt”

```
Array of source matrix

  0.00  1.00  3.00  2.00  4.00  3.00  0.00  2.00  1.00  6.00  5.00
1.23  0.00
3.00  3.00 21.30  0.00  4.00  0.00  5.00  2.00 34.40  0.00  0.00

 Source matrix
  0.00    0.00    2.00    3.00    0.00
  6.00    0.00    0.00    0.00    1.23
  0.00    0.00   21.30    0.00    0.00
  0.00    0.00    0.00    0.00    0.00
  0.00   34.40    0.00    0.00    0.00




Array of conjugated matrix

  0.00  1.00  2.00  6.00  0.00  2.00  5.00 34.40  0.00  3.00  1.00
2.00  3.00 2
1.30  0.00  4.00  1.00  3.00  0.00  5.00  2.00  1.23  0.00  0.00

 Conjugated matrix
  0.00    6.00    0.00    0.00    0.00
  0.00    0.00    0.00    0.00   34.40
  2.00    0.00   21.30    0.00    0.00
  3.00    0.00    0.00    0.00    0.00
  0.00    1.23    0.00    0.00    0.00

Matrix is not skew symmetric
Для продолжения нажмите любую клавишу . . .
```

Вывод:

Разреженные матрицы – интересная для изучения структура данных. Они часто возникают при решении таких задач, как решение дифференциальных уравнений в частных производных. Подобные задачи наверняка будут встречаться на старших курсах университета. Для их обработки и хранения придумано множество специальных схем. Одну из них я реализовал, для этого мне потребовалось научиться лучше работать с массивами в языке Си.