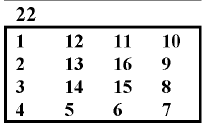
|  |  |
| --- | --- |
|  | **Отчёт по лабораторной работе** № 14 по курсу 1  Фундаментальная информатика  Студент группы M8O-101Б-23 Терентьев Михаил Андреевич, № по списку: 23  Контакты www, e-mail, icq, skype mihateren2003@gmail.com  Работа выполнена: “11 “ декабря 2023г.  Преподаватель: каф. 806 Крылов Сергей Сергеевич  Входной контроль знаний с оценкой  Отчёт сдан “13“ декабря 2023г., итоговая оценка  Подпись преподавателя |

1. **Тема**: Вложенные циклы с параметрами. Обход и линеаризация матриц
2. **Цель работы**: Освоить работу с матрицами в языке Си.
3. **Задание**: Написать программу ввода квадратной матрицы и печати в строку всех её элементов в заданном порядке(вариант 22)   
   
4. **Оборудование** *Процессор* ***Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz*** *с ОП 8 ГБ , ТТН* ***512 ГБ*** *Встроенный монитор 13,2", FHD (1366 x 768) 16:9, частота обновления: 60 Гц*
5. *Другие устройства*
6. **Программное обеспечение:** *ПВЭМ студента* (*лабораторное*):

Программное обеспечение ПЭВМ студента, если использовалось:

Операционная система семейства Windows, наименование версия Windows 11,

Компилятор gcc. Система программирования C.

Редактор текстов Visual Studio Code версия 1.85

1. **Идея, метод, алгоритм** *решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями):*

1. Создается функция createDynamicMatrix(int n), которая выделяет память под двумерный массив размера n x n. В этой функции используется динамическое выделение памяти с помощью функции malloc.

2. Создается функция fillMatrix(int\*\* matrix, int size), которая заполняет матрицу значениями в соответствии с шаблоном "квадратной спирали". В этой функции используются переменные value (для хранения текущего значения, которое будет записано в матрицу) и переменные topRow, bottomRow, leftCol, rightCol (для отслеживания текущих границ области, которая еще не заполнена). Затем выполняется цикл while, в котором последовательно заполняются четыре стороны области внутри матрицы, сужая границы после каждого заполнения.

3. Создается функция printMatrix(int\*\* matrix, int size), которая выводит матрицу на экран. Эта функция просто проходит по каждому элементу матрицы и выводит его значение.

4. Создается функция freeMatrix(int\*\* matrix, int size), которая освобождает память, выделенную для двумерного массива. В этой функции используется функция free для освобождения памяти каждой строки и затем для освобождения указателей на строки.

5. В функции main() пользователю предлагается ввести размер матрицы. Затем создается двумерный массив нужного размера с помощью функции createDynamicMatrix(). После этого массив заполняется значениями согласно шаблону "квадратной спирали" с помощью функции fillMatrix(). Результат выводится на экран с помощью функции printMatrix(). Наконец, память, выделенная для массива, освобождается с помощью функции freeMatrix().

Таким образом, алгоритм программы создает матрицу заданного размера и заполняет ее значениями в соответствии с шаблоном "квадратной спирали", после чего выводит результат на экран и освобождает выделенную память.

1. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int\*\* createDynamicMatrix(int n) {

int\*\* array = (int\*\*)malloc(n \* sizeof(int\*)); // Выделяем память под указатели на строки

for (int i = 0; i < n; i++) {

array[i] = (int\*)malloc(n \* sizeof(int)); // Выделяем память под каждую строку

}

return array;

}

void fillMatrix(int\*\* matrix, int size){

int value = 1;

int topRow = 0, bottomRow = size - 1, leftCol = 0, rightCol = size - 1;

while ((topRow <= bottomRow) && (leftCol <= rightCol)) {

for (int i = topRow; i <= bottomRow; i++) {

matrix[i][leftCol] = value;

value++;

}

leftCol++;

for (int i = leftCol; i <= rightCol; i++) {

matrix[bottomRow][i] = value;

value++;

}

bottomRow--;

for (int i = bottomRow; i >= topRow; i--) {

matrix[i][rightCol] = value;

value++;

}

rightCol--;

for (int i = rightCol; i >= leftCol; i--) {

matrix[topRow][i] = value;

value++;

}

topRow++;

}

}

void printMatrix(int\*\* matrix, int size) {

int i, j;

for (i = 0; i < size; i++){

for (j = 0; j < size; j++){

printf("%2d ", matrix[i][j]);

}

printf("\n");

}

}

void freeMatrix(int\*\* matrix, int size) {

for (int i = 0; i < size; i++) {

free(matrix[i]);

}

free(matrix);

}

int main() {

int size;

printf("Введите размер матрицы: ");

scanf("%d", &size);

int\*\* matrix = createDynamicMatrix(size); // создаем двумерный массив заданного порядка

fillMatrix(matrix, size); // заполняем матрицу нужными значениями

printf("Матрица нужного шаблона: \n");

printMatrix(matrix, size); // смотрим результат

freeMatrix(matrix, size); // очищаем память

return 0;

} }

Допущен к выполнению работы**. Подпись преподавателя**

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем)

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int\*\* createDynamicMatrix(int n) {

int\*\* array = (int\*\*)malloc(n \* sizeof(int\*)); // Выделяем память под указатели на строки

for (int i = 0; i < n; i++) {

array[i] = (int\*)malloc(n \* sizeof(int)); // Выделяем память под каждую строку

}

return array;

}

void fillMatrix(int\*\* matrix, int size){

int value = 1;

int topRow = 0, bottomRow = size - 1, leftCol = 0, rightCol = size - 1;

while ((topRow <= bottomRow) && (leftCol <= rightCol)) {

for (int i = topRow; i <= bottomRow; i++) {

matrix[i][leftCol] = value;

value++;

}

leftCol++;

for (int i = leftCol; i <= rightCol; i++) {

matrix[bottomRow][i] = value;

value++;

}

bottomRow--;

for (int i = bottomRow; i >= topRow; i--) {

matrix[i][rightCol] = value;

value++;

}

rightCol--;

for (int i = rightCol; i >= leftCol; i--) {

matrix[topRow][i] = value;

value++;

}

topRow++;

}

}

void printMatrix(int\*\* matrix, int size) {

int i, j;

for (i = 0; i < size; i++){

for (j = 0; j < size; j++){

printf("%2d ", matrix[i][j]);

}

printf("\n");

}

}

void freeMatrix(int\*\* matrix, int size) {

for (int i = 0; i < size; i++) {

free(matrix[i]);

}

free(matrix);

}

int main() {

int size;

printf("Введите размер матрицы: ");

scanf("%d", &size);

int\*\* matrix = createDynamicMatrix(size); // создаем двумерный массив заданного порядка

fillMatrix(matrix, size); // заполняем матрицу нужными значениями

printf("Матрица нужного шаблона: \n");

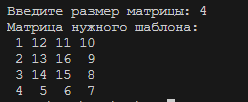
printMatrix(matrix, size); // смотрим результат

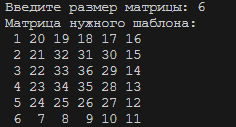
freeMatrix(matrix, size); // очищаем память

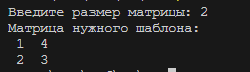
return 0;

} }

*Тесты:*







1. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб.  или  дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

1. **Замечание автора по существу работы**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_
2. **Выводы**

В ходе выполнения лабораторной работы я научился работе с матрицами на языке Си.

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_