

Отчет по лабораторной работе № 14 по курсу _____

Студент группы 108 Каширин Кирилл, № по списку 8

Контакты www, e-mail, icq, skype _____

Работа выполнена: « 19 » ноября 2020 г.

Преподаватель: _____ каф. 806 _____

Входной контроль знаний с оценкой _____

Отчет сдан « » 201 г., итоговая оценка _____

Подпись преподавателя _____

1. Тема: Вложенные циклы с параметрами. Обход и линеаризация матриц.

2. Цель работы: Составить программу ввода квадратной матрицы и печати в строку всех её элементов в заданном порядке следования(обхода).

3. Задание (вариант №14): Вывести элементы матрицы в указанном порядке

4. Оборудование (лабораторное):
ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____ Мб,
НМД _____ Мб. Терминал _____ адрес _____. Принтер _____
Другие устройства _____

Оборудование ПЭВМ студента, если использовалось:
Процессор _____ с ОП _____ Мб, НМД _____ Мб. Монитор _____
Другие устройства _____

5. Программное обеспечение (лабораторное):
Операционная система семейства _____, наименование _____ версия _____
интерпретатор команд _____ версия _____
Система программирования _____ версия _____
Редактор текстов _____ версия _____
Утилиты операционной системы _____

Прикладные системы и программы _____
Местонахождение и имена файлов программ и данных _____

Программное обеспечение ЭВМ студента, если использовалось:
Операционная система семейства _____, наименование _____ версия _____
интерпретатор команд _____ версия _____.
Система программирования _____ версия _____
Редактор текстов _____ версия _____
Утилиты операционной системы _____

Прикладные системы и программы _____
Местонахождение и имена файлов программ и данных на домашнем компьютере _____

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

С помощью конечного автомата будем индексы элементов матрицы.

Заведем несколько состояний, которые будут указывать в каком порядке выводить элементы:

Пусть i, j - индексы элемента, на который мы только что вывели, n^*n - размеры квадратной матрицы

1) DiagDown - вывод элементов матрицы по диагонали(сверху-вниз)

Выводится элемент, расположенный на $i+1$ строке $j+1$ столбца. Если размер матрицы $1*1$

выводим элемент и переходим в состояние Finish

2) DiagUp - вывод элементов матрицы по диагонали(снизу-вверх)

Выводится элемент, расположенный на $i-1$ строке $j-1$ столбца

3) MoveUp - вывод элемента, расположенного на $i-1$ строке j столбца

В заданном порядке это происходит тогда и только тогда, когда $j = n-1$;

4) MoveDown - вывод элемента, расположенного на $i+1$ строке j столбца

В заданном порядке это происходит тогда и только тогда, когда $j = 0$;

В матрице после двух проходов элементов(когда они расположены на 1 или n , т.е. $i = 0$ или $i = n-1$)

происходит симметричное отображение относительно главной диагонали матрицы

Для этого потребовалось еще два состояния

1) MoveMirrorUp - отображение элемента расположенного ниже главной диагонали.

Индексы элемента становятся равными $i=0$ и $j=n-1-j$;

2) MoveMirrorDown - отображение элемента, расположенного выше главной диагонали.

Индексы элементы становятся равными $i=n-1$ и $j=n-1-j$;

Если мы выводим элемент, находящийся на $n-1$ строке и 0 столбце, мы переходим в состояние MoveMirrorUp, выводим элемент и переходим в состояние Finish

Если мы выводим элемент, находящийся на 0 строке и $n-1$ столбце, мы переходим в состояние MoveMirrorDown, выводим элемент и переходим в состояние Finish

Finish - завершает работу конечного автомата.

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

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

Допущен к выполнению работы. Подпись преподавателя _____

8. Распечатка протокола (подклейте листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

```
library.h
enum State {Finish=0, DiagDown, MoveUp, MoveDown, MoveMirrorDown, MoveMirrorUp, DiagUp, Start};
enum State nextState(enum State current, unsigned i, unsigned j, unsigned n);

main.c
#include <stdio.h>
#include "library.h"
int main()
{
    const unsigned max_size = 1000 ;
    unsigned n;
    scanf("%d", &n);
    int matrix[max_size][max_size];
    for (unsigned i = 0; i < n; i++){
        for (unsigned j = 0; j < n;j++){

            scanf("%d", &matrix[i][j]);
        }
    }
    unsigned i = 0;
    unsigned j = 0;
    enum State currentState = Start;
    while (currentState!=Finish){
        if (currentState == DiagDown){

            ++i;
            ++j;
        }
        if (currentState == DiagUp){

            --i;
            --j;
        }
        if (currentState == MoveUp){

            --i;
        }
        if (currentState == MoveDown){

            ++i;
        }
        if (currentState == MoveMirrorDown){

            i = n-1;
            j = n-1-j;
        }
        if (currentState == MoveMirrorUp){

            i=0;
            j=n-1-j;
        }
        printf("%d ",matrix[i][j]);
        currentState = nextState(currentState,i,j,n);
    }
    printf("\n");
}

functions.c
#include <stdio.h>
#include "library.h"
enum State nextState(enum State current, unsigned i, unsigned j, unsigned n)
{
    enum State newState;
    switch(current){
        case Start:
        {
            if (i==0 && n!=1) {
                newState = DiagDown;
            } else {
                newState = Finish;
            }
            break;
        }
        case DiagDown:
        {
            if (j==n-1) {
                newState = MoveUp;
            } else if (i==n-1) {
                newState = MoveMirrorUp;
            }
            break;
        }
        case DiagUp:
        {
            if (j==0){
                newState = MoveDown;
            } else if (i==0){
                newState = MoveMirrorDown;
            }
            break;
        }
        case MoveDown:
        {
            if (i==n-1 && j==0){
                newState = MoveMirrorUp;
            } else if (i==0){
                newState = MoveMirrorUp;
            } else if (j==0) {
                newState = DiagDown;
            }
            break;
        }
        case MoveUp:
        {
            if (i==0){
                newState = MoveMirrorDown;
            } else if (j==n-1){
                newState = DiagUp;
            }
            break;
        }
        case MoveMirrorUp:
        {
            if (i==0 && j==n-1){
                newState = Finish;
            } else if (i==0){
                newState = DiagDown;
            } else {
                newState = Finish;
            }
            break;
        }
        case MoveMirrorDown:
        {
            if (i==n-1 && j==0){
                newState = Finish;
            } else if (i==n-1){
                newState = DiagUp;
            }
            break;
        }
    }
    return newState;
}
```

9. **Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) 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

10. Замечания автора по существу работы _____

11. Выводы

Я составил программу ввода квадратной матрицы и печати в строку всех ее элементов в порядке, указанном в условии задания.

Недочёты при выполнении задания могут быть устранены следующим образом: _____

Подпись студента _____