Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра экономической математики, информатики и статистики (ЭМИС)

МАТРИЦЫ И ФУНКЦИИ

Отчёт по лабораторной работе по дисциплине “Введение в программирование”

Студент гр. 543-1
 /А.Е. Мухамеджан
“____”_____2023
Доцент кафедры ЭМИС
_____ _____ Е.А. Шельмина
оценка
“__”_____ 2023 г.

Томск 2023

Лабораторная работа №6
“Матрицы и функции”

Цель работы: научиться обрабатывать двумерные массивы с использованием функций.

Описание массива производится с помощью обычного оператора описания, при этом за именем массива в квадратных скобках должна быть записана целая положительная константа или константное выражение, равное размеру этого массива, то есть максимально возможному числу элементов.

Имя массива без квадратных скобок за ним имеет значение, равное адресу первого элемента этого массива.

Имя массива с квадратными скобками, в которых записано индексное выражение целого типа, обозначает значение соответствующего элемента массива. В языке Си нумерация элементов массива начинается с нуля.

Индексированные переменные могут использоваться в любых выражениях в тех местах, где допускается применение переменных соответствующих типов.

При работе с индексированными переменными необходимо внимательно следить за тем, чтобы индексы не вышли из допустимого диапазона, определяемого описаниями массивов. Дело в том, что компилятор не проверяет факт выхода индексов за границы массива, а при ошибочном занесении данных за пределы массива может запортиться нужная информация и, скорее всего, компьютер зависнет.

Ввод и вывод массива: язык Си не имеет встроенных средств для ввода-вывода массива целиком, поэтому массив вводят и выводят поэлементно с помощью циклов.

Инициализация массива: инициализация - присвоение значений вместе с описанием данных. Ранее была рассмотрена инициализация простых переменных, например: int a = 5.

Для инициализации массива за его именем располагают знак присваивания и список инициализации, который представляет собой заключенные в фигурные скобки и разделенные запятыми инициализирующие значения.

Констант в списке инициализации должно быть не больше, чем объявленный размер массива. Если их меньше, то элементы для которых нет констант обнуляются. Для инициализируемого массива допускается вообще не указывать размер. В этом случае размер массива определяется по количеству констант, например по описанию int c[] = { 1, 15, 18, 11, 20 }; транслятор выделит 10 байт для хранения массива из 5 двухбайтовых целых чисел.

Частный случай инициализации массива - инициализация строк.

Двумерные массивы (массивы массивов): элементом массива может быть в свою очередь тоже массив. Таким образом, мы приходим к понятию двумерного массива или матрицы. Описание двумерного массива строится из описания одномерного путем добавления второй размерности.

Анализ подобного описания необходимо проводить в направлении выполнения операций [], то есть слева направо.

Для наглядности двумерный массив можно представить в виде таблицы с числом строк, равным первому размеру массива, и числом столбцов, равным второму размеру массива. Пример изображен на рисунке 1.

Массив a	Столбец 0	Столбец 1	Столбец 2
Строка 0	18	21	5
Строка 1	6	7	11
Строка 2	30	52	34
Строка 3	24	4	67

Рисунок 1 – иллюстрация представления двумерного массива как таблицы

Имя двумерного массива без квадратных скобок за ним имеет значение адреса первого элемента этого массива, то есть значение адреса первой строки - одномерного массива из трех элементов. При использовании в выражениях тип имени двумерного массива преобразуется к типу адреса строки этого массива. В нашем примере тип имени массива a в выражениях

будет приведен к типу адреса массива из трех элементов типа int и может использоваться во всех выражениях, где допускается использование соответствующего адреса.

Имя двумерного массива с одним индексным выражением в квадратных скобках за ним обозначает соответствующую строку двумерного массива и имеет значение адреса первого элемента этой строки.

Имя двумерного массива с двумя индексными выражениями в квадратных скобках за ним обозначает соответствующий элемент двумерного массива и имеет тот же тип.

Ввод двумерного массива осуществляется поэлементно с помощью двух вложенных циклов. Для ввода массива по столбцам достаточно поменять местами строки программы, являющиеся заголовками циклов.

В языке Си допускается использовать не только двумерные, но и трехмерные, четырехмерные и т. д. массивы. Их использование ничем принципиально не отличается от использования двумерных массивов, однако на практике они применяются значительно реже.

Общие сведения о функциях: подпрограмма - именованная, логически законченная группа операторов языка, которую можно вызвать для выполнения любое количество раз из различных мест программы. В языке Си подпрограммы реализованы в виде функций. Функция принимает параметры и возвращает единственное скалярное значение.

Заголовок функции имеет вид: type имя_функции ([список параметров]), type – тип возвращаемого функцией значения; список параметров - список передаваемых в функцию величин, которые отделяются запятыми, каждому параметру должен предшествовать его тип;

Для того, чтобы функция вернула какое-либо значение, в ней должен быть оператор return значение;

Для вызова функции необходимо указать имя функции и в круглых скобках список передаваемых в функцию значений.

Передача параметров в функцию: параметры, указанные в заголовке функции, называются формальными. Параметры, передаваемые в функцию, называются фактическими.

При обращении к функции фактические параметры передают свое значение формальным и больше не изменяются. Типы, количество и порядок следования формальных и фактических параметров должны совпадать. С помощью оператора return из функции возвращается единственное значение.

Возврат значений из функций: возврат результата работы функции в вызывающую программу в виде единственного значения можно осуществить с помощью оператора return. При этом результат возвратится как значение самой функции и должен иметь соответствующий тип.

Типы возвращаемых значений могут быть любыми, кроме массивов. Тип void означает, что функция не возвращает никакого значения. Тип void* означает, что функция возвращает указатель на произвольный тип данных.

Ход работы:

Задания для самостоятельной работы:

Задание 1. Требуется определить двумерный массив a целых чисел (например, размера 5×7), заполнить его случайными числами или ввести его элементы с клавиатуры. В одномерный массив b записать характеристики двумерного массива a в соответствии с вариантом задания.

Программа должна содержать следующие функции:

- инициализация элементов двумерного массива случайными числами или вводимыми с клавиатуры;
- заполнение одномерного массива в соответствии с заданием;
- вывод одномерного массива на экран;
- вывод двумерного массива на экран.

Для выполнения задания 1 (приложение А) использовался вариант 12, представленный в лабораторной работе №6 и интегрированная среда разработки Microsoft Visual Studio.

Задание 2. Написать программу для решения задачи согласно варианту. Использовать пользовательские функции аналогично заданию 1 (функция ввода, вывода матрицы, решения задачи согласно варианту).

Для выполнения задания 2 (приложение Б) использовался вариант 12, представленный в лабораторной работе №6 и интегрированная среда разработки Microsoft Visual Studio.

Вывод: в результате выполнения лабораторной работы получены навыки обработки двумерных массивов с использованием функций

Приложение А

(обязательное)

Код программы задания 1

```
#include <iostream>
#include <time.h>
using namespace std;
int n = 5, m = 5;

int** CreateTwoSArr(int n, int m);
int* CreateOneSArr(int n);
int* FillOSArr(int* fillingArr, int** source);
void PrintTSArr(int** arr);
void PrintOSArr(int* arr);

int main()
{
    /*12. Найти максимальные элементы каждого столбца матрицы a(5, 7) и сохранить их
в одномерном массиве b.*/
    srand(time(NULL));
    setlocale(LC_ALL,"Rus");

    int** a = CreateTwoSArr(n, m);
    int* b = CreateOneSArr(m);
    PrintTSArr(a);
    PrintOSArr(b);
    FillOSArr(b, a);
    PrintOSArr(b);



}
int** CreateTwoSArr(int n, int m) {
    int** arr = new int*[n];
    for (int i = 0; i < n;i++) {
        arr[i] = new int[m];
    }
    for (int i = 0;i < n;i++) {
        for (int j = 0;j < m;j++) {
            arr[i][j] = -10 + rand() % (10 - (-10) + 1);
        }
    }
    return arr;
}
int* CreateOneSArr(int n) {
    int* arr = new int[n];
    for (int i = 0; i < n;i++) {
        arr[i] = 0;
    }
    return arr;
}
int* FillOSArr(int* fillingArr, int** source) {
    for (int i = 0; i < m;i++) {
        int max = source[0][i];
        for (int j = 0; j < n; j++) {
            if (max < source[j][i]) {
                max = source[j][i];
            }
        }
        fillingArr[i] = max;
    }
    return fillingArr;
}
```

```
void PrintTSArr(int** arr) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
    cout << endl;
}
void PrintOSArr(int* arr) {
    for (int i = 0; i < m; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;
}
```

Приложение Б

(обязательное)

Код программы задания 2

```
#include <iostream>
#include <time.h>
using namespace std;
int n = 5, m = 5;

int** CreateTwoSArr(int n, int m);
void PrintTSArr(int** arr);
int SumSubD(int** arr);
int SumAll(int** arr);
int main()
{
    /* Задан двумерный массив Y из 4-ч строк и 4-х столбцов.
    Составить программу, которая вычисляет S – сумму элементов побочной диагонали и
значение всех элементов массива*/
    srand(time(NULL));
    int** Y = CreateTwoSArr(n, m);
    PrintTSArr(Y);
    cout << SumSubD(Y) << endl;
    cout << SumAll(Y) << endl;
}

int** CreateTwoSArr(int n, int m) {
    int** arr = new int* [n];
    for (int i = 0; i < n;i++) {
        arr[i] = new int[m];
    }
    for (int i = 0;i < n;i++) {
        for (int j = 0;j < m;j++) {
            arr[i][j] = -10 + rand() % (10 - (-10) + 1);
        }
    }
    return arr;
}
void PrintTSArr(int** arr) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
    cout << endl;
}
int SumSubD(int** arr) {
    int sum = 0;
```

```
    for (int i = 0; i < n; i++) {
        int j = m - 1 - i;
        sum += arr[i][j];
    }
    return sum;
}
int SumAll(int** arr) {
    int sum = 0;
    for (int i = 0;i < n;i++) {
        for (int j = 0; j < m;j++) {
            sum += arr[i][j];
        }
    }
    return sum;
}
```