МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ХАРЬКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

для выполнения практических и лабораторных работ

ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ С++. ДВУМЕРНЫЕ МАССИВЫ

по курсу

«ИНФОРМАТИКА»

для студентов факультета «Автоматика и приборостроение»

дневной и заочной форм обучения

Утверждено
редакционно-издательским
советом университета,
протокол № 2 от 25.06.2015 г.

Харьков
НТУ «ХПИ»
2016

Методические указания для выполнения практических и лабораторных работ «Основы программирования С++. Двухмерные массивы» по курсу «Информатика» для студентов факультета «Автоматика и приборостроение» дневной и заочной форм обучения / Сост. Тверитникова Е. Е. , Крылова В. А. – Х.: НТУ «ХПИ», 2016. – 36 с.

Авторы Е. Е. Тверитникова
В. А. Крылова

Рецензент А. В. Ивашко

Кафедра информационно-измерительных технологий и систем

Вступление

Развитие современных технологий невозможно без использования компьютерной техники и программного обеспечения. Подготовка специалистов в области автоматики, измерительной и медицинской техники требует обширных знаний и навыков владения вычислительной техникой, а так же знания основ алгоритмизации и программирования на языках высокого уровня таких как C/C++.

Методические указания предназначены для изучения языка программирования C++ на практических занятиях и лабораторных работах, а также для самостоятельного освоения. В методических указаниях подробно и доступно рассмотрены синтаксис, семантика и техника программирования на языке C++ с использованием двумерных массивов. Описаны все этапы проектирования программ, приведены подробные комментарии программного кода, проанализированы результаты вычислений, показаны типичные проблемы и пути их решения. Большое внимание уделяется алгоритмам инициализации, поиска, сортировки и примерам решения задач с использованием двумерных массивов. Приведены индивидуальные задания для выполнения лабораторных работ и задания для самостоятельного изучения основ алгоритмизации и программирования на языке C++. Рассмотренные примеры и задания, помогут эффективному освоению основ программирования для учащихся и начинающих программистов.

ЛАБОРАТОРНАЯ РАБОТА 10

ДВУМЕРНЫЕ МАССИВЫ, КАК ЧАСТНЫЙ СЛУЧАЙ МНОГОМЕРНЫХ МАССИВОВ

Цель работы – приобретение и закрепление практических навыков при написании программ на языке С++ с использованием двумерных массивов.

Двумерные массивы

Применявшиеся до сих пор массивы называются одномерными, поскольку каждый такой массив можно представить в виде одной строки данных. Двумерный массив, который в большей степени подобен таблице, представляет собой совокупность строк и столбцов, на пересечении которых находится конкретное значение. Структура двумерного массива, состоящего из 3-х строк и 4-х столбцов показана на рис.1.1. Для описания двумерного массива необходимо указать *тип* элементов массива, *имя массива* и *размер массива* (количество строк и столбцов). Если двумерный массив определяется с помощью операторов описания, то обе его размерности (количества строк и количества столбцов) должны быть константами или константными выражениями, так как инструкция по выделению памяти формируется компилятором до выполнения программы.

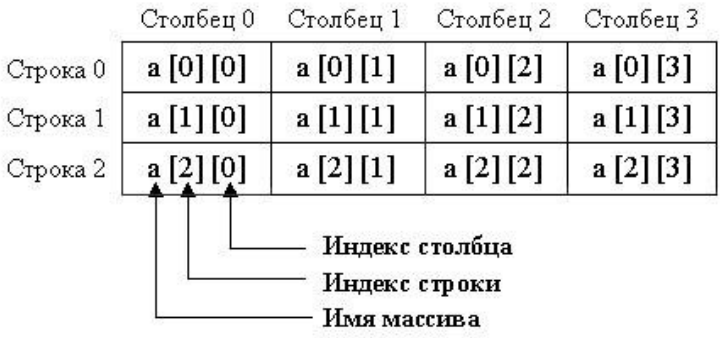


Рисунок 1.1 – Двумерный массив

Общий синтаксис описания двумерного массива:

***тип_данных имя_массива* [*число_строк*][*число_столбцов*];**

В объявлении двумерного массива, также как и в объявлении одномерного массива, первым делом, нужно указать:

- тип данных;
- имя массива.

После чего в квадратных скобках указывается количество строк двумерного массива, во вторых квадратных скобках – количество столбцов двумерного массива:

int a[3][5]; // целочисленная матрица из 3 строк и 5 столбцов

Несмотря на то, что мы представляем массив в виде матрицы, на самом деле − любой двумерный массив располагается в памяти построчно: сначала нулевая строка, затем первая и так далее:

$a_{00}, a_{01}, a_{02}, a_{03}, a_{04}, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{20}, a_{21}, a_{22}, a_{23}, a_{24},$

|– – – 0-я строка – – –| |– – – 1-я строка – – –| |– – –2-я строка – – –|

Строки массива не отделены одна от другой. В памяти сначала располагается одномерный массив *a*[0], представляющей собой нулевую строку массива, затем – массив *a*[1], представляющий собой первую строку массива и т.д. При просмотре массива от начала в первую очередь изменяется правый индекс (номер столбца).

Объявление двумерного массива через константные переменные

const int row = 3; *// строки*
const int col = 4; *// столбцы*
int array[*row*][*col*]; *// массив размером row на col* (3x4)

Для доступа к отдельному элементу массива, так же, как и для одномерного необходимо указать индекс, при этом применяется конструкция вида *a*[*i*][*j*], *i* (номер строки), *j* (номер столбца) – выражение целочисленного типа. Каждый индекс может изменяться от 0 до значения соответствующий размерности, уменьшенной на единицу. Для прохода по двумерному массиву удобнее использовать два цикла ***for***, вложенных друг в друга, первый (внешний) цикл ***for*** с индексом *i* для перебора строк, а второй (внутренний) цикл ***for*** с индексом *j* для перебора столбцов.

Необходимо помнить, что первый индекс всегда воспринимается как номер строки, второй – как номер столбца, независимо от имени переменной.

Инициализация двумерного массива

1. Инициализация элементов массива при создании двумерного массива.

Значения указываются подряд и построчно вписываются в массив:

int array [2][2]={7,8,10,3};

Каждая строка заключается в отдельные фигурные скобки:

int array [2][2]={{1,2},{7,8}};

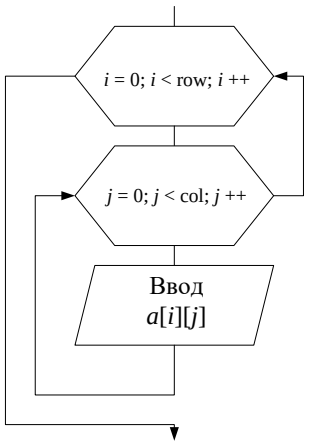
Если значение пропущено, оно будет инициализировано нулем:

int array [3][3]={{7,8},{10,3,5}};

2. Инициализация элементов двумерного массива через клавиатуру с помощью циклов.

```
void main( )
{
const int row=3; // строки
const int col=3; // столбцы
int a[row][col]; // массив размером row на col
for (int i=0; i<row; i++) // перебираем номера строк
  for (int j=0; j<col; j++) // перебираем номер столбца
  {
    cout<<"Введите элемент массива"<<"["<<i<<"]" <<"["<<
j<<"]"<<'\n';
    cin>>a[i][j];
  }
```

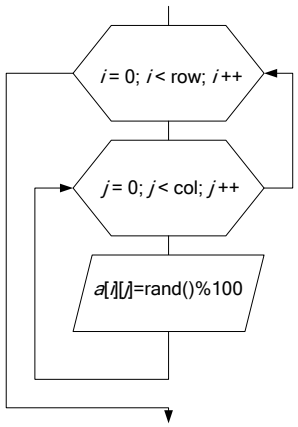
Блок-схема алгоритма ввода двумерного массива с клавиатуры



3. Инициализация элементов двумерного массива через генератор случайных чисел с помощью циклов

```
void main( )
{
const int row=5; // строки
const int col=5; // столбцы
int a[row][col]; // массив размером row на col
for (int i=0; i<row; i++) // перебираем номер строки
  for (int j=0; j<col; j++) // перебираем номер столбца
        a[i][j]=rand()%100; // инициализация целыми значениями в диапазоне от 0 до 100
}
```

Блок-схема алгоритма ввода двумерного массива через генератор случайных чисел



Вывод двумерного массива на экран в виде матрица

```
void main( )
{
const int row=5; // строки
const int col=5; // столбцы
int a[row][col]; // массив размером row на col
for (int i=0; i<row; i++) // перебираем номер строки
{
      for (int j=0; j<col; j++)  // перебираем номер столбца
            cout<<a[i][j]<<'\t';
      cout<<'\n';
}
```

ПРИМЕРЫ РЕШЕНИЯ ЗАДАНИЙ

Задание 1

Написать программу, которая для целочисленной матрицы 5 x 10 находит среднее арифметическое ее элементов и количество четных элементов в каждой строке.

I. *Выбор метода*

Для инициализации значений элементов двумерного массива используется генератор целых случайных чисел. Для нахождения среднего арифметического элементов массива необходимо найти общую сумму, после чего разделить её на количество элементов двумерного массива (50 элементов). Определение количества четных элементов в каждой строке требует просмотра матрицы по строкам, при этом таких количеств будет 5, вывод на экран каждого количества будет осуществляться каждый раз, как только программа будет переходить на следующую строку.

Например, рассмотрим двумерный массив 4 x 4.

5 7 8 9 – 1-я строка
1 2 6 4 – 2-я строка
3 1 7 5 – 3-я строка
8 4 9 3 – 4- строка

Среднее арифметическое значений элементов матрица равно 5,125.

Количество четных элементов в каждой строке:

1-я строка – 1 (8).

2-я строка – 3 (2, 6, 4).

3-я строка – 0.

4-я строка – 2 (8, 4).

II. *Описание решения задачи на псевдокоде*

1. Начало.

2. Задание значений элементов массива через генератор случайных чисел.

3. Вывод на экран созданного двумерного массива.

4. Нахождение суммы всех элементов двумерного массива.

5. Деление суммы элементов массива на их количество для получения среднего арифметического.

6. Нахождение количества четных элементов в каждой строке.

7. Конец.

III. *Схема алгоритма работы программы*

Блок-схема алгоритма программы представлена на рисунке 1.1.

IV. *Разработка текста программы*

1. Подключаем в файле ***stdafx.h*** необходимые для работы программы библиотеки, а также объявляем константы:

 # include <iostream> – для работы операторов ввода/вывода;

 # include <stdlib.h> – для работы функции генератора случайных чисел;

```
using namespace std;
const  int  n = 5; // объявление константы для количества строк
const  int  m = 10; // объявление константы для количества столбцов
```

2. Разработка раздела описания переменных

```
int a [n][m], i, j, k;
float s, sr;
```

a – двумерный массив из целых чисел;

i – целочисленная переменная, индекс строки элемента массива;

j – целочисленная переменная, индекс столбца элемента массива;

s – вещественная переменная, сумма элементов массива;

sr – вещественная переменная, среднее арифметическое элементов массива;

k – целочисленная переменная, количество четных элементов.

3. Разработка тела программы

Для инициализации двумерного массива необходимо организовать два цикла ***for*** с переменными параметрами – индекса строки и столбца элементов массива. В цикле каждому элементу массива присваивается случайное целое число из диапазона от 0 до 50.

```
for (int i=0; i<n; i++) // перебираем номер строки
  for (int j=0; j<m; j++)   // перебираем номер столбца
        a[i][j]=rand()%50;
```

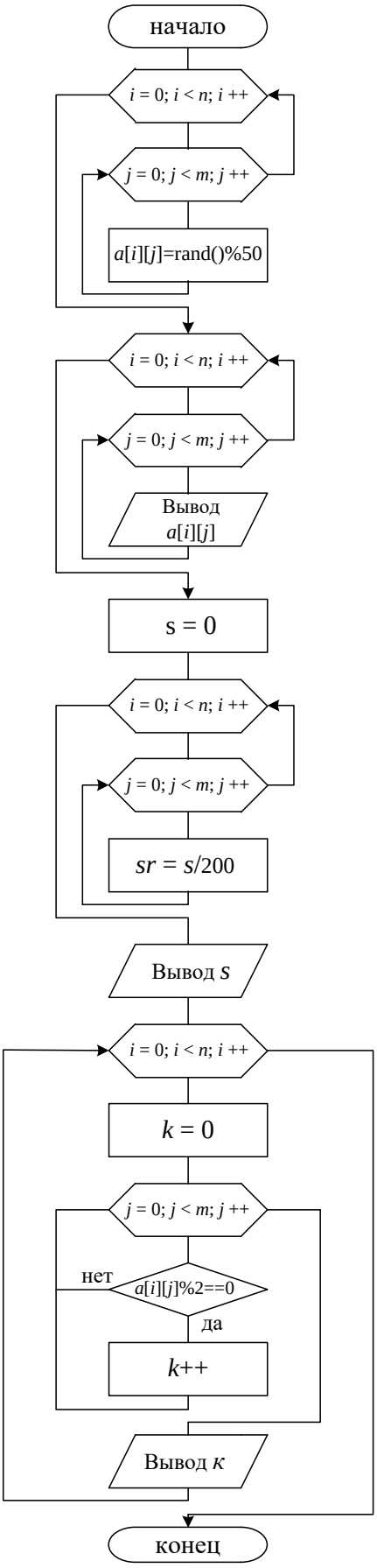


Рисунок 1.1 – Блок-схема алгоритма программы задания 1

Вывод на экран созданного генератором случайных чисел двумерного массива

```
for (int i=0; i<n; i++) // перебираем номер строки
{
    for (int j=0; j<m; j++)    // перебираем номер столбца
        cout<<a[i][j]<<'\t';
    cout<<'\n';
}
```

Для нахождения суммы элементов массива, необходимо сначала обнулить переменную ***s,*** в теле вложенного цикла организовать накопление суммы. Для нахождения среднего арифметического – поделить сумму на количество элементов массива

```
s=0;
for (i=0; i<n; i++) // перебираем номер строки
    for ( j=0; j<m; j++)       // перебираем номер столбца
        s += a[i][j]; // накопление суммы
sr = s/50; // среднее арифметическое
```

При вычислении количества четных элементов для каждой строки массива выполняются следующие действия: фиксируется номер строки (***i***), обнуляется счетчик (переменная ***k***), перебираются номера столбцов (***j***) и осуществляется проверка, является ли элемент массива в ***i*** строке – четным (***a***[***i***][***j***]%2==0), если да, то счетчик увеличивается на единицу (***k***++). Количество четных элементов для каждой строки свое, и в результате мы получим столько значений, сколько строк в матрице. В данной задаче для хранения этих значений можно использовать одну переменную целого типа, т.к. они вычисляются последовательно и после чего выводятся на экран. Однако если эти значения могут впоследствии потребоваться одновременно, тогда для хранения должен использоваться одномерный массив с количеством элементов, равным количеству строк. Переменная, которая вычисляет количество четных элементов (***k***) должна обнуляться перед циклом перебора столбцов (***k=0***), т.е. просмотра очередной строки, поскольку для каждой строки его вычисление начинается заново.

```
for (i=0; i<n; i++) // перебираем номер строки
{
k=0; // обнуление счетчика
for ( j=0; j<m; j++)        // перебираем номер столбца
        if (a[i][j]%2==0) // проверка на четность элемента массива
                k++; // увеличение счетчика на 1
cout<<"Количество четных элементов"<<i<<"строки"<<k<<'\n';
}
```

Синтаксис программы

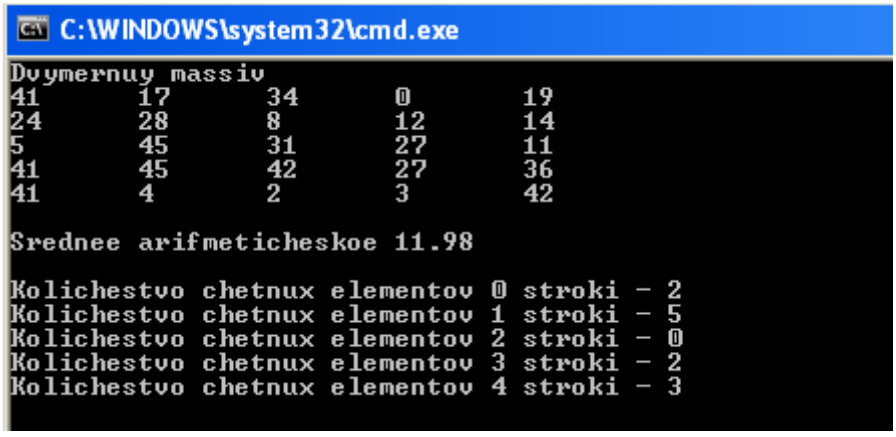
```
# include <iostream>
# include <stdlib.h>
using namespace std;
const  int  n = 5; // объявление константы для количества строк
const  int  m = 10; // объявление константы для количества столбцов

int main ( )
{
    int a [n][m], i, j , k;
    float s, sr;
    for ( i=0; i<n; i++) // перебираем номер строки
            for ( j=0; j<m; j++)       // перебираем номер столбца
                    a[i][j]=rand()%50;
    cout<<"Двумерный массив \n";
    for ( i=0; i<n; i++) // перебираем номер строки
    {
       for ( j=0; j<m; j++)        // перебираем номер столбца
               cout<<a[i][j]<<'\t';
       cout<<'\n';
    }
    s=0;
    for ( i=0; i<n; i++) // перебираем номер строки
       for ( j=0; j<m; j++)        // перебираем номер столбца
                    s += a[i][j]; // накопление суммы
```

```
sr=s/50; // среднее арифметическое
cout<<"Среднее арифметическое"<<sr<<'\n';
for ( i=0; i<n; i++) // перебираем номер строки
{
   k=0; // обнуление счетчика
   for ( j=0; j<m; j++)        // перебираем номер столбца
         if  (a[i][j]%2==0)  // проверка на четность элемента массива
   k++; // увеличение счетчика на 1
   cout<<"Количество четных элементов"<<i<<"строки – "<<k<<'\n';
}
}
```

4. Отладка и запуск программы

Для отладки программы используем клавишу *F*7, убеждаемся в отсутствии ошибок и запускаем программу на исполнение с помощью комбинации клавиш *Ctrl*+*F*5. Ниже приведены результаты работы программы.



Задание 2

Написать программу, которая для целочисленной матрицы размером 5x5, создает одномерный массив, состоящий из максимальных элементов столбцов двумерного массива. Полученный одномерный массив вывести на экран. Найти сумму главной и произведение побочной диагонали массива, результат вывести на экран.

I. *Выбор метода*

Для создания одномерного массива, который содержит максимальные элементы каждого столбца, необходимо объявить одномерный массив размером равным количеству столбцов. После нахождения максимального элемента в столбце, этот элемент должен быть записан в одномерный массив.

Например, дан массив 4 на 4:

6 1 9 2	$a[0][0]$ $a[0][1]$ $a[0][2]$ $a[0][3]$
3 5 8 6	$a[1][0]$ $a[1][1]$ $a[1][2]$ $a[1][3]$
0 2 9 7	$a[2][0]$ $a[2][1]$ $a[2][2]$ $a[2][3]$
4 3 6 0	$a[3][0]$ $a[3][1]$ $a[3][2]$ $a[3][3]$

Тогда созданный одномерный массив из максимальных элементов столбцов будет содержать:

{$a[0][0]$ $a[1][1]$ $a[0][2]$ $a[2][3]$} или {6 5 9 7}

Для нахождения суммы главной диагонали двумерного массива необходимо сложить значения элементов с индексами:

$$a[0][0] + a[1][1] + a[2][2] + \ldots + a[i][i] + \ldots + a[n-1][n-1]$$

Для нашего примера сумма главной диагонали:

$$a[0][0] + a[1][1] + a[2][2] + a[3][3] = 6 + 5 + 9 + 0 = 20$$

Для нахождения произведения побочной диагонали двумерного массива необходимо перемножить значения элементов с индексами:

$$a[0][3] + a[1][2] + a[2][2] + \ldots + a[i][n-1-i] + \ldots + a[n-1][0]$$

Для нашего примера произведения побочной диагонали:

$$a[0][3] + a[1][2] + a[2][2] + a[3][0] = 2 \cdot 8 \cdot 2 \cdot 4 = 128$$

II. *Описание решения задачи на псевдокоде*

1. Начало.
2. Задание элементов массива через генератор случайных чисел.
3. Вывод на экран созданного двумерного массива.
4. Нахождение максимального элемента столбца.

5. Сохранение очередного максимального элемента в одномерный массив.

6. Вывод на экран созданного одномерного массива.

7. Нахождение сумму значений элементов главной диагонали.

8. Нахождение произведения значений элементов побочной диагонали.

III. *Схема алгоритма работы программы*

Блок-схема алгоритма программы представлена на рисунке 1.2.

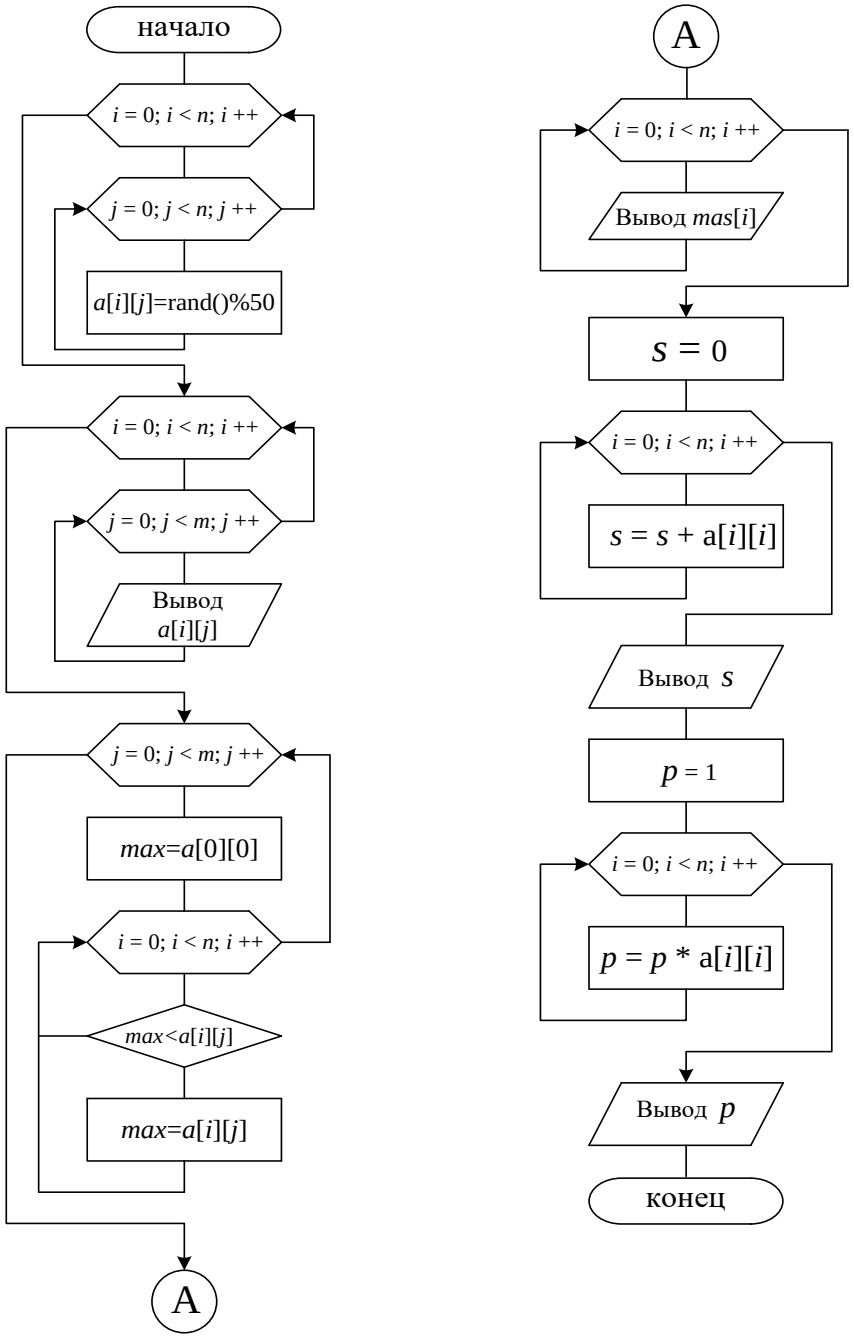


Рисунок 1.2 – Блок-схема алгоритма программы задания 2

IV. *Разработка текста программы*

1. Подключаем в файле ***stdafx.h*** необходимые для работы программы библиотеки, а также объявляем константы:

#include <iostream> – для работы операторов ввода/вывода;

#include <stdlib.h> – для работы функции генератора случайных чисел;

```
using namespace std;
const  int  n = 5; // объявление константы для количества строк и столбцов
```

2. Разработка раздела описания переменных

```
int a [n][n], i, j, max, b [n], s, p;
```

a – двумерный массив из целых чисел;

i – целочисленная переменная, индекс строки элемента массива;

j – целочисленная переменная, индекс столбца элемента массива;

max – целочисленная переменная, максимальный элемент столбца;

b[n] – одномерный массив из целых чисел;

s – целочисленная переменная, сумма главной диагонали;

p – целочисленная переменная, произведение побочной диагонали.

3. Разработка тела программы

Для инициализации двумерного массива необходимо организовать два цикла ***for*** с переменными параметрами – индекса строки и столбца элементов массива. В цикле каждому элементу массива присваивается случайное целое число из диапазона от 0 до 50. Также в этом цикле можно организовать и вывод на экран созданного с помощью генератора случайных чисел двумерный массив.

```
for ( i=0; i<n; i++) // перебираем номер строки
{
  for ( j=0; j< n; j++)       // перебираем номер столбца
  {
          a[i][j]=rand()%50; // задание значения элемента массива
          cout<<a[i][j]<<'\n'; // вывод на экран элемента массива
  }
  cout<<'\n';
}
```

Для создания одномерного массива, который содержит максимальные элементы столбцов, необходимо организовать два цикла ***for***. Первый цикл (***for (j=0; j< n; j++***)) устанавливает номер столбца, затем в теле внешнего цикла переменой ***max*** (максимальному элементу) присваивается нулевой элемент очередного столбца ***max=a[0][j]***, затем перебираются строки и работает внутренний цикл (***for (i=0; i<n; i++***)), в теле которого происходит поиск максимального элемента. Далее этот максимальный элемент записывается в одномерный массив ***b[j]=max***., при этом программа заканчивает тело внешнего цикла и устанавливает следующий номер столбца. И так далее пока не будут обработаны все ***n−1***.

```
for ( j=0; j< n; j++)          // перебираем номера столбцов
{
  max = a[0][j]; |// начальное значение максимального элемента
  for ( i=0; i<n; i++) // перебираем номера строк
        if (a[i][j]>max) //сравнение максимума с элементом массива
              max=a[i][j]; // присвоение максимуму большего
элемента
  b[j]=max; // запись
}
```

Для нахождения суммы элементов главной диагонали двумерного массива, необходимо обнулить переменную суммы (***s=0***) и достаточно организовать один цикл для перебора номера строки ***for (i=0; i<n; i++)***, в теле которого будет накапливаться сумма (***s+=a[i][i]***).

```
s=0; // обнуление суммы
for ( i=0; i<n; i++)
  s+=a[i][i]; // накопление суммы
```

Для нахождения произведения значений элементов побочной диагонали, необходимо переменной, которая хранит результат произведения присвоить единицу (***p=1***). Затем достаточно организовать один цикл для перебора номера строки ***for (i=0; i<n; i++)***, в теле которого будет осуществляться произведение предыдущего значения произведения на текущее значение побочной диагонали (***p*=a[i][n−1−i]***).

```
p=1; // начальное значение произведения
for ( i=0; i<n; i++)
  p*=a[i][n−1−i]; // накопление произведения
```

Синтаксис программы

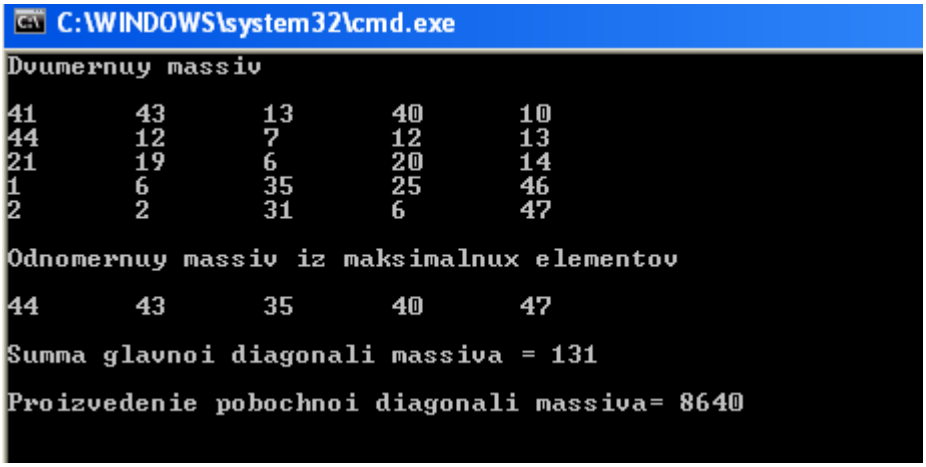
```
# include <iostream>
# include <stdlib.h>
using namespace std;
const  int  n = 5; // объявление константы для количества строк

int main ( )
{
   int a [n][ n], i, j, max, b [n], s, p;
   cout<<" Двумерный массив \n";
  for ( i=0; i<n; i++) // перебираем номер строки
  {
      for ( j=0; j< n; j++)  // перебираем номер столбца
      {
            a[i][j]=rand()%50; // задание значения элемента массива
            cout<<a[i][j]<<'\t'; // вывод на экран элемента массива
            }
      cout<<'\n';
   }
   cout<<'\n';
  for ( j=0; j< n; j++)       // перебираем номера столбцов
  {
      max = a[0][j]; |// начальное значение максимального элемента
      for ( i=0; i<n; i++) // перебираем номера строк
         if (a[i][j]>max) //сравнение максимума с элементом массива
            max=a[i][j]; // присвоение максимуму большего элемента
      b[j]=max; // запись
  }
cout<<" Одномерный массив из максимальных элементов \n";
  for (i=0; i<n; i++)
        cout<<b[i]<<'\t';
  cout<<'\n';
  s=0;
  for ( i=0; i<n; i++)
       s+=a[i][i];
```

```
    cout<<“Произведение побочной диагонали массива = “<<s<<’\n’;
    cout<<’\n’;
   p=1;
   for ( i=0; i<n; i++)
       p*=a[i][n−1−i];
    cout<<“Сумма главной диагонали массива = “<<p<<’\n’;
   cout<<’\n’;
return 0;
}
```

4. Отладка и запуск программы

Для отладки программы используем клавишу *F*7, убеждаемся в отсутствии ошибок и запускаем программу на исполнение с помощью комбинации клавиш *Ctrl*+*F*5. Ниже приведены результаты работы программы.



Задание 3

Написать программу, которая упорядочивает строки прямоугольной целочисленной матрицы по возрастанию сумм их элементов.

I. *Выбор метода*

Матрица может быть создана с помощью генератора случайных чисел. Для упорядочивания строк матрицы по возрастанию сумм элементов её строк, необходимо эти суммы вычислить. Поскольку все они понадобится при упорядочивании строк их необходимо где-то хранить. Поэтому найденные суммы должны быть записаны в одномерный массив, количество элементов которого соответствует количеству строк матрицы,

а *i*-ый элемент этого массива содержит сумму *i*-ой строки. Для сортировки строк используется один из методов упорядочивания массива – метод обмена.

Например, дана матрица из 16 элементов:

4 6 7 5 – сумма 22
7 2 8 1 – сумма 18
3 5 4 9 – сумма 21
1 9 2 8 – сумма 20

После сортировки строк по возрастанию сумм элементов ее строк, матрица будет выглядеть:

7 2 8 1 – сумма 18
1 9 2 8 – сумма 20
3 5 4 9 – сумма 21
4 6 7 5 – сумма 22

II. *Описание решения задачи на псевдокоде*

1. Начало.
2. Задание элементов массива через генератор случайных чисел.
3. Вывод на экран созданного двумерного массива.
4. Нахождение суммы элементов строк матрицы.
5. Сохранение очередной суммы в одномерный массив.
6. Вывод на экран созданного одномерного массива.
7. Сортировка строк матрица по возрастанию сумм.
8. Вывод на экран отсортированной матрицы.

III. *Схема алгоритма работы программы*

Блок-схема алгоритма программы представлена на рисунке 1.3.

IV. *Разработка текста программы*

1. Подключаем в файле ***stdafx.h*** необходимые для работы программы библиотеки, а также объявляем константы:

 # include <iostream> – для работы операторов ввода/вывода;

 # include <stdlib.h> – для функции генератора случайных чисел;

```
using namespace std;
const  int  n = 5; // объявление константы для количества строк и столбцов
```

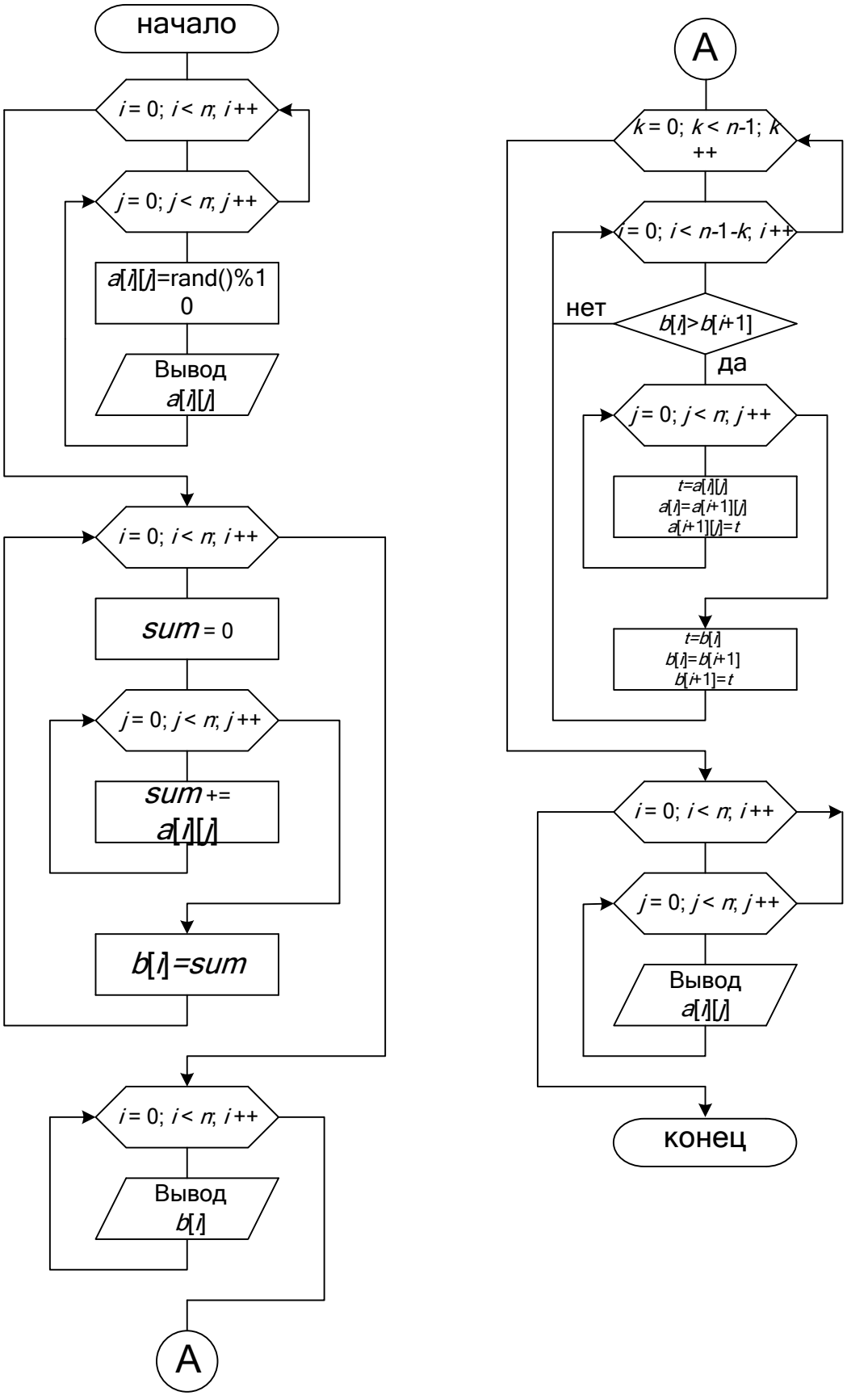


Рисунок 1.3 – Блок-схема алгоритма программы задания 3

2. Разработка раздела описания переменных

int a **[*n*][*n*]*, i, j, sum, b* [*n*]*, k, t;***

a – двумерный массив из целых чисел;

i – целочисленная переменная, индекс строки элемента массива;

j – целочисленная переменная, индекс столбца элемента массива;

sum – целочисленная переменная, сумма элементов строк;

b **[*n*] –** одномерный массив из сумм строк;

k – целочисленная переменная, для сортировки массива;

t – целочисленная переменная, для временного хранения.

3. Разработка тела программы

Для инициализации двумерного массива необходимо организовать два цикла ***for*** с переменными параметрами – индекса строки и столбца элементов массива. В цикле каждому элементу массива присваивается случайное целое число из диапазона от 0 до 10. Также в этом цикле можно организовать и вывод на экран созданного с помощью генератора случайных чисел двумерный массив.

```
for ( i=0; i<n; i++) // перебираем номер строки
{
  for ( j=0; j< n; j++)       // перебираем номер столбца
  {
          a[i][j]=rand()%10; // задание значения элемента массива
          cout<<a[i][j]<<'\t'; // вывод на экран элемента массива
  }
  cout<<'\n';
}
```

Для создания одномерного массива, который содержит суммы элементов строк матрицы, необходимо организовать два цикла ***for***. После первого (внешнего) цикла необходимо обнулить сумму (***sum*****=0**), т.к. для каждой строки своя сумма и перед просмотром элементов по столбцам и накопления суммы она всегда должна обнуляться. Далее работает второй (внутренний) цикл, в теле которого происходит накопление суммы элементов строк (***sum*****+=*a*[*i*][*j*]**).

```
for (i =0; i<n; i++) // перебор строк
{
        sum=0; // обнуление суммы
        for (j =0; j<n; j++) // перебор столбцов
                sum+=a[i][j]; // вычисление суммы элементов строк
}
```

Вывод созданного одномерного массива на экран.

```
for (i =0; i<n; i++) // перебор строк
        cout<<b[i]<<'\t';
```

Для упорядочивания строк матрицы используется одни из методов сортировки – метод обменом. Он, состоит в том, что сравнение элементов начинается с 0-ого элемента одномерного массива ($b[i]>b[i+1]$). Сравнивается нулевой и первый элемент, если нулевой элемент больше первого, то следует их упорядочивание, т.е. поменять местами.

```
t=b[i];
b[i]=b[i+1];
b[i+1]=t;
```

При этом также необходимо упорядочить и элементы матрицы, т.е. поменять местами i-ую и i+1-ую строку, для этого организовывается цикл для перебора номеров столбцов **for (j=0; j<n; j++)** и в теле цикла обмен двух ячеек матрицы **a[i][j]** и **a[i+1][j]**.

```
t=a[i][j];
a[i][j]=a[i+1][j];
a[i+1][j]=t;
```

Сравнив второй и третий для них тоже выполняется упорядочивание и операция продолжается до сравнения последнего и предпоследнего элементов. В результате можно заметить, что наибольший элемент последовательности станнит последним. Укорачивание последовательности до тех пор, пока последовательность не будет содержать два элемента, потом выполняется для двух элементов (обмен) и алгоритм заканчивает свое выполнение.

```
for ( k=0; k<n−1; k++) //
        for ( i=0; i<n−1−k; i++)
                if(b[i]>b[i+1])
```

```
{
        t=b[i];
        b[i]=b[i+1];
        b[i+1]=t;
        for (j=0; j<n; j++)
        {
                t=a[i][j];
                a[i][j]=a[i+1][j];
                a[i+1][j]=t;
}
```

Синтаксис программы

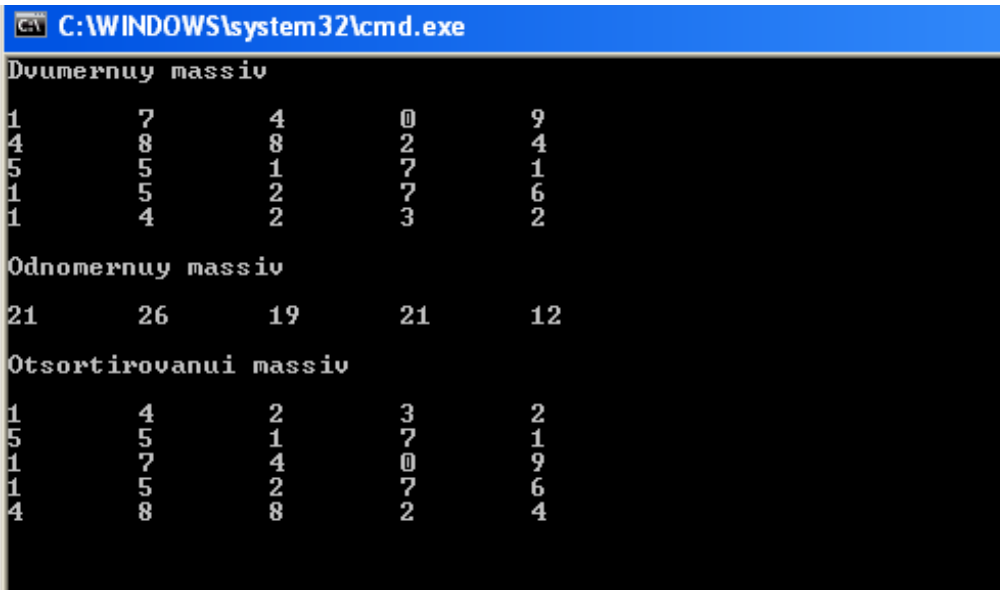
```
# include <iostream>
# include <stdlib.h>
using namespace std;
const  int  n = 5; // объявление константы для количества строк

int main ( )
{
   int a [n][ n], i, j, sum, b [n], k, t;
   cout<<"Dvumernuy massiv \n";
   cout<<'\n';
  for ( i=0; i<n; i++) // перебираем номер строки
  {
     for ( j=0; j< n; j++)  // перебираем номер столбца
     {
      a[i][j]=rand()%10; // задание значения элемента массива
      cout<<a[i][j]<<'\n'; // вывод на экран элемента массива
      }
     cout<<'\n';
   }
   for (i =0; i<n; i++) // перебор строк
   {
      sum=0; // обнуление суммы
      for (j=0; j<n; j++) // перебор столбцов
            sum+=a[i][j]; // вычисление суммы элементов строк
```

```
    }
    cout<<"Odnomernuy massiv \n";
    cout<<'\n';
    for (i =0; i<n; i++) // перебор строк
        cout<<b[i]<<'\t';
    for ( k=0; k<n−1; k++) // сортировка строк матрицы
        for (i=0; i<n−1−k; i++) // индекс одномерного массива и номер строки
            if(b[i]>b[i+1]) // сравнение элементов одномерного массива
            {
                t=b[i]; // обмен элементов одномерного массива
                b[i]=b[i+1];
                b[i+1]=t;
                for (j=0; j<n; j++) // перебор номеров столбцов
                {
                    t=a[i][j]; // обмен между строками матрицы
                    a[i][j]=a[i+1][j];
                    a[i+1][j]=t;
                }
    cout<<"Dvumernuy massiv \n";
    cout<<'\n';
   for ( i=0; i<n; i++) // перебираем номер строки
   {
      for ( j=0; j< n; j++)  // перебираем номер столбца
         cout<<a[i][j]<<'\t'; // вывод на экран элемента массива
      cout<<'\n';
   }
return 0;
}
```

4. Отладка и запуск программы

Для отладки программы используем клавишу *F*7, убеждаемся в отсутствии ошибок и запускаем программу на исполнение с помощью комбинации клавиш *Ctrl*+*F*5. Ниже приведены результаты работы программы.



Порядок выполнения лабораторной работы

1. В соответствии с номером по журналу выберите индивидуальное задание.
2. Разработайте алгоритм решения задания.
3. Составьте текст программы.
4. Создайте проект в интегрированной среде разработки *Microsoft Visual Studio* (*lab*10_фамилия.*cpp).*
5. Введите текст программы.
6. Скомпилируйте программу. Если в программе есть ошибки, их необходимо исправить. Если ошибок нет, то появится сообщение об успешной компиляции.
7. Запустите программу на выполнение, проанализируйте результаты и убедитесь в правильности решения задачи.
8. Выполните отчет по лабораторной работе, который должен содержать: титульный лист; цель работы; индивидуальное задание; алгоритм работы программы; текст программы; результаты работы программы; выводы.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ 1

1. Создать двумерный массив, состоящий из вещественных чисел. Найти номер строки, в которой находится максимальный элемент массива. Найти среднее арифметическое элементов, расположенных только в четных столбцах массива. Результат вывести на экран.

2. Создать двумерный массив, состоящий из вещественных чисел. Вычислить среднее арифметическое значение элементов главной и побочной диагонали матрицы. Найти местоположение максимального элемента двумерного массива. Результат вывести на экран.

3. В созданном двумерном массиве выбрать элементы, которые больше суммы главной диагонали массива, записать полученные значения в одномерный массив. Отсортировать полученный одномерный массив в порядке убывания.

4. Создать двумерный массив, состоящий из целых чисел. Найти сумму чисел, которые расположены выше главной диагонали. Определить, расположены ли максимальный и минимальный элементы массива в одной строке или в одном столбце.

5. Создать двумерный массив, состоящий из вещественных чисел. Найти произведение двух максимальных и разность двух минимальных элементов двумерного массива. Создать одномерный массив, состоящий из сумм элементов строк двумерного массива. Результат вывести на экран.

6. Создать двумерный массив, состоящий из целых чисел. Если сумма индексов, максимального элемента двумерного массива имеет четное значение, то подсчитать количество простых элементов массива, иначе − произведение совершенных элементов массива.

7. Создать двумерный массив a, состоящий из целых чисел. Получить массив b из массива a, путем удалением n-й строки и k-го столбца, где n и k – целые числа, введенные с клавиатуры.

8. Создать двумерный массив, состоящий из целых чисел. Найти сумму элементов двумерного массива, которые делятся на пять. Определить делители полученного числа. Создать одномерный массив, состоящий из произведений элементов столбцов.

9. Создать матрицу, состоящий целых чисел, размером 5 х 5. Построить матрицу, обратную заданной. Создать одномерный массив, состоящий из минимальных элементов каждой строки. Отсортировать полученный массив в порядке убывания.

10. В двумерном массиве, состоящем из вещественных чисел, найти максимальный элемент массива и поменять его местами с элементом, стоящем на пересечении двух диагоналей.

11. В двумерном массиве, состоящем из целых чисел, найти сумму элементов, стоящих только на четных строках. Проверить, является ли полученное число – числом Фиббоначи. Создать одномерный массив, состоящий из максимальных значений элементов каждого столбца.

12. Создать двумерный массив, состоящий из целых чисел. В созданном массиве в строках, которые имеют хотя бы один нулевой элемент найти количество элементов кратных 5. Создать одномерный массив, состоящий из значений средних арифметических элементов строк.

13. Дан двумерный массив. К элементам *k*-ой строки прибавить элементы *p*-ой строки (*k, p* – вводятся с клавиатуры). Создать одномерный массив, состоящий из значений средних арифметических элементов столбцов.

14. Создать двумерный массив, состоящий из целых чисел. В созданном массиве найти минимальный элемент побочной диагонали. Создать одномерный массив, состоящий из значений произведений элементов строк двумерного массива.

15. Создать двумерный массив, состоящий из целых чисел. В созданном массиве поменять местами строку, содержащую элемент с минимальным значением, со строкой, содержащей элемент с максимальным значением. Создать одномерный массив, который состоит из простых чисел двумерного массива.

16. Создать матрицу, состоящую из целых чисел Написать программу, которая устанавливает, является ли данная матрица симметричной относительно главной диагонали. Создать одномерный массив, состоящий количеств четных элементов строк двумерного массива.

17. В двумерном массиве заменить нулем все элементы, расположенные на главной диагонали и выше нее. Найти сумму элементов *k*-го столбца и *p*-ой строки (*k, p* – вводятся с клавиатуры).

18. В двумерном массиве найти количество четных элементов в строках, содержащих хотя бы один нулевой элемент. Определить, является ли полученное число – простым. Создать одномерный массив, состоящий из минимальных элементов столбцов двумерного массива.

19. Дан двумерный массив из целых элементов. Найти среднее арифметическое элементов, у которых сумма индексов равна k (k – вводится с клавиатуры). Создать одномерный массив, состоящий из четных элементов двумерного массива.

20. В двумерном массиве удалить k-ую строку и p-ый столбец (k, p – вводятся с клавиатуры). Создать одномерный массив, состоящий из нечетных элементов двумерного массива.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ 2

Дана целочисленная прямоугольная матрица размерностью m x n. Необходимо определить:

Вариант 1

1. Количество строк, не содержащих ни одного нулевого элемента.

2. Определить максимальный элемент в каждой строке.

Вариант 2

1. Определить количество столбцов, не содержащих ни одного нулевого элемента.

2. Определить максимальный элемент в каждом столбце.

Вариант 3

1. Количество столбцов, содержащих хотя бы один нулевой элемент.

2. Определить сумму элементов в каждой строке.

Вариант 4

1. Произведение элементов в тех строках, которые не содержат отрицательных элементов.

2. Определить сумму элементов главной диагонали матрицы.

Вариант 5

1. Сумму элементов в тех строках, которые не содержат отрицательных элементов.

2. Определить минимальные по значению элементы в каждой строке матрицы.

Вариант 6

1. Сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент.

2. Вычислить среднее арифметическое значение отрицательных элементов главной диагонали матрицы.

Вариант 7

1. Среднее арифметическое элементов главной диагонали матрицы.

2. Найти сумму элементов двумерного массива, которые находятся на четных позициях (номер строки и номер столбца – четные числа).

Вариант 8

1. Разделить элементы каждого столбца заданной матрицы на последний элемент столбца. Преобразованную матрицу вывести на экран.

2. Определить сумму элементов в тех столбцах, которые содержат хотя бы один положительный элемент.

Вариант 9

1. Определить и вывести на экран сумму положительных элементов в каждой строке матрицы.

2. Минимум среди сумм элементов диагоналей, параллельных главной диагонали матрицы.

Вариант 10

1. Определить и вывести на экран сумму значений всех элементов матрицы.

2. Определить сумму элементов в тех столбцах, которые содержат хотя бы один отрицательный элемент.

Вариант 11

1. Найти сумму модулей элементов, расположенных ниже главной диагонали.

2. В двумерном массиве выбрать элементы, которые больше суммы положительных элементов главной диагонали массива.

Вариант 12

1. Найти и вывести на экран максимальные по значению элементы в каждой строке матрицы.

2. Определить среднее арифметическое значение отрицательных элементов главной диагонали матрицы.

Вариант 13

1. Определить и вывести на экран минимальные элементы в каждой строке матрицы.

2. Найти и вывести на экран сумму отрицательных элементов в каждой строке матрицы.

Вариант 14

1. Определить номер первого из столбцов, содержащих хотя бы один нулевой элемент.

2. Найти сумму модулей элементов, расположенных выше главной диагонали.

Вариант 15

1. Вычислить среднее арифметическое значение отрицательных элементов главной диагонали матрицы.

2. Найти количество строк, среднее арифметическое элементов которых меньше заданной величины.

Вариант 16

1. Вычислить среднее арифметическое значение отрицательных элементов главной диагонали матрицы.

2. Найти номер первой из строк, содержащих хотя бы один положительный элемент.

Вариант 17

1. Количество строк, содержащих хотя бы один нулевой элемент.

2. Найти номер первой из строк, не содержащих ни одного положительного элемента.

Вариант 18

1. Сумму элементов в тех строках, которые не содержат отрицательных элементов;

2. Определить среднее арифметическое значение отрицательных элементов главной диагонали матрицы.

Вариант 19

1. Количество отрицательных элементов в тех строках, которые содержат хотя бы один нулевой элемент;

2. Найти и вывести на экран сумму отрицательных элементов в каждой строке матрицы.

Вариант 20

1. Поменять местами строку и столбец на пересечении максимального элемента массива.

2. Определить номера строк и столбцов, имеющих нулевые элементы.

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. В двумерном массиве найти сумму элементов главной диагонали двумерного массива. Расположить элементы первой строки и второго столбца в порядке возрастания.

2. В двумерном массиве определить произведение суммы индексов двух максимальных элементов главной диагонали. В созданном массиве упорядочить его строки по возрастанию первых элементов строк.

3. Найти произведение элементов побочной диагонали двумерного массива. Расположить элементы четвертой строки и второго столбца в порядке возрастания.

4. В двумерном массиве найти сумму положительных элементов двух диагоналей массива. Определить, является ли полученное округленное число простым. В созданном массиве упорядочить его строки по убыванию первых элементов строк.

5. В двумерном массиве выбрать элементы, которые больше суммы элементов первого столбца, и записать их в одномерный массив. Расположить элементы полученного одномерного массива в порядке убывания.

6. Определить количество столбцов в двумерном массиве, содержащих 0-ой элемент. Расположить элементы последней строки в порядке возрастания.

7. В двумерном массиве найти местоположение максимального и минимального элементов и поменять их местами. Расположить элементы первой строки в порядке возрастания.

8. Вычислить среднее арифметическое значение положительных элементов не главной диагонали матрицы. Расположить элементы последнего столбца в порядке возрастания.

9. В двумерном массиве определить количество строк содержащих 0-ой элемент. Расположить элементы второй строки и второго столбца в порядке убывания.

10. В двумерном массиве найти сумму только простых чисел. Расположить элементы последней строки в порядке возрастания.

11. В двумерном массиве, состоящем из целых чисел заменить элементы главной диагонали на числа из ряда Фиббоначи. Расположить элементы четвертого столбца в порядке возрастания.

12. Дан двумерный массив. Создать одномерный массив, состоящий из максимальных элементов столбцов двумерного массива, отсортировать полученный массив. Расположить элементы последнего столбца двумерного массива в порядке возрастания.

13. Положительные элементы двумерного массива записать в одномерный массив a. Создать одномерный массив b, обратный полученному массиву a. Расположить элементы последней строки двумерного массива в порядке возрастания.

14. Найти сумму элементов двумерного массива, которые делятся на пять. Создать одномерный массив, состоящий из делителей полученного числа. Расположить элементы второго столбца в порядке убывания.

15. Составить программу, которая проверяет, является ли данная матрица симметричной относительно главной диагонали. Расположить элементы третьей строки в порядке возрастания.

16. Составить программу, которая проверяет, является ли данная матрица симметричной относительно побочной диагонали. Расположить элементы последнего столбца в порядке убывания.

17. Дан двумерный массив. Преобразовать массив таким образом, чтобы столбец с номером p непосредственно следовал за столбцом с номером q, сохранив порядок следования остальных столбцов.

18. В двумерном массиве удалить *k*-ую строку и *l*-й столбец. Расположить элементы первой строки в порядке возрастания и последней строки в порядке убывания.

19. В двумерном массиве упорядочить его строки по возрастанию первых элементов строк.

20. Дана прямоугольная матрица. Поменять местами строку, содержащую элемент с максимальным значением, со строкой, содержащей элемент с минимальным значением.

Контрольные вопросы

1. Отметьте правильные варианты объявления двумерного массива:

Варианты ответов:

А. *int ar*[5][5]

Б. *float br* [5.5][5.5];

В. *const int str*=5; *const int stl*=3; *double ar*[stl][str];

Г. *double* [3][3];

Д. *long ar*[6][6];

2. Что будет на экране после выполнения следующего фрагмента кода?

```
const double row=10;
const double col=10;
double ar[row][col];
for (int i=0; i<10; i++){
      for (int j=0; j<10; j++){
            cout<<ar[i][j]<<’  ‘;
       }
       cout<<”\n\n”;
}
```

Варианты ответов:

А. Все нули.

Б. Все единицы.

В. Случайные числа (мусор).

Г. Ошибка на этапе выполнения.

Д. Ошибка на этапе компиляции.

3. Что будет на экране после выполнения следующего фрагмента кода?

```
const int size = 2;
int ar[size][size]={{2, 2},{2, 2}};
for (int i=0; i<size; i++){
      for (i=0; i<size; i++){
            ar[i][i]=i;
            cout<<ar[i][i]<<" ";
       }
       cout<<"\n\n";
}
```

Варианты ответов:

А. 01

Б. 10

В. 00

Г. 11

Д. Ошибка на этапе компиляции

4. Что будет на экране после выполнения следующего фрагмента кода?

```
int  ar[2][2]={1,2,3,4,5,6};
cout<<ar[1][1];
```

Варианты ответов:

А. 4

Б. Ошибка на этапе компиляции

В. 1

Г. 6

Д. 3

5. Что будет на экране после выполнения следующего фрагмента кода?

```
int a=2;
int ar[10]={a*=a, a*=a , a*=a , a*=a};
for (int i=0; i<10; i++){
        cout<<ar[i]<<' ';
        }
```

Варианты ответов:

А. 2 2 2 2 2 2 2 2 2 2

Б. 2 2 2 2 0 0 0 0 0 0

В. 4 16 256 65536 0 0 0 0 0 0

Г. 0 0 0 0 0 0 0 0 0 0

Д. Ошибка на этапе компиляции

6. Что будет на экране после выполнения следующего фрагмента кода?

```
const int size = 2;
int ar[size][size]={0,0};
for (int i=0; i<size; i++){
      for (j=0; j<size; j++){
              ar[i][j]=rand()%5+10;
              cout<<ar[i][i]<<" ";
       }
       cout<<"\n\n";
}
```

Варианты ответов:

А. 0 0 0 0

Б. 0 0 1 1

В. 2 3 1 7

Г. 13 11 12 14

Д. Ошибка на этапе компиляции

СПИСОК ЛИТЕРАТУРЫ

1. Светозарова Г. И. Практикум по программированию на алгоритмических языках / Г. И. Светозарова, Е. В. Сигитов, А. В. Козловский. – Минск : Наука, 1980. – 317 с.
2. Дейтел Х. М. Как программировать на C++ / Х. М. Дейтел, П. Дж. Дейтел. – М. : ЗАО «Издательство БИНОМ», 2000 – 1024 с.
3. Страуструп Бьерн. Язык программирования C++ / Бьерн Страуструп. – М : Бином, 2002. – 1098с.
4. Шилдт Герберт. Справочник программиста по C/C++ / Герберт Шилдт; пер. с англ. – 3-е изд. –М. : Изд. дом «Вильямс», 2003. – 432 с.
5. Павловская Т. А. Структурное программирование C/C++. Практикум / Т. А. Павловская, Ю. А. Щупак. – Спб. : Питер, 2007. – 239 с.
6. Подбельский В. В. Язык Си+ : учеб. пособ. / В. В. Подбельский. – М. : БИНОМ, 1995 – 400 с.
7. Керниган Б. Язык программирования Си / Б. Керниган, Д. Ритчи. – М. : Финансы и статистика, 1992.
8. Либерти Д. Освой самостоятельно C++ за 21 день / Д. Либерти.
9. Васильченков О.Г., Тверитникова Е.Е., Крылова В.А. Алгоритмизация задач и основы программирования на C/C++ учеб. пособие – Х.: Изд-во «Підручник НТУ «ХПІ», 2015 – 228 с.

НАВЧАЛЬНЕ ВИДАННЯ

Методичні вказівки з курсу «Інформатика»

для студентів факультету «Автоматика і приладобудування»
денної та заочної форми навчання

Укладачі: ТВЕРИТНИКОВА Олена Євгенівна
КРИЛОВА Вікторія Анатоліївна

ОСНОВИ ПРОГРАМУВАННЯ НА МОВІ С++. ДВОВИМІРНІ МАСИВИ

Роботу до друку рекомендував А.М. Борисенко

Редактор

План 2015 р.,

Підписано до друку . .07. Формат 60×84 1/16. Папір друк. № 2.
Друк–ризографія. Гарнітура Times New Roman. Ум. друк. арк. 3,2.
Обл. – вид. арк. 2,7. Наклад 100 прим. Зам. № . Ціна договірна.

Видавничий центр НТУ «ХПІ». 61002, Харків, вул.. Фрунзе, 21.
Свідоцтво про реєстрацію ДК № 3657 від 24.12.2009 р.

Друкарня НТУ «ХПІ», 61002, Харків, вул. Фрунзе, 21.