БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

факультет радиофизики и компьютерных технологий

кафедра информатики и компьютерных систем

Н.В. Серикова

ПРАКТИЧЕСКОЕ РУКОВОДСТВО

к лабораторному практикуму

«МАССИВЫ И УКАЗАТЕЛИ»

по курсу

«ПРОГРАММИРОВАНИЕ»

2020

МИНСК

Практическое руководство к лабораторному практикуму «МАССИВЫ. УКАЗАТЕЛИ» по курсу «ПРОГРАММИРОВАНИЕ» предназначено для студентов, изучающих базовый курс программирования на языке С++, специальностей «Радиофизика», «Физическая электроника», «Компьютерная безопасность».

Руководство содержит некоторый справочный материал, примеры решения типовых задач с комментариями.

Все примеры протестированы в среде Microsoft Visual Studio Community 2019.

Автор будет признателен всем, кто поделится своими сообра­же­ниями по совершенствованию данного пособия.

Возможные предложения и замечания можно присылать по адресу:

*E-mail:* [*Serikova@bsu.by*](mailto:Serikova@bsu.by)*,*

ОГЛАВЛЕНИЕ

[Массив 4](#_Toc56035244)

[Синтаксис описания одномерного массива 5](#_Toc56035245)

[Синтаксис описания двумерного массива 6](#_Toc56035246)

[Зависимости индексов элементов в двумерном массиве [n][n] 7](#_Toc56035247)

[Сортировка массивов 8](#_Toc56035248)

[Алгоритм «Решето Эратосфена» 9](#_Toc56035249)

[Арифметика больших чисел 10](#_Toc56035250)

[Указатели 11](#_Toc56035251)

[Ссылки 13](#_Toc56035252)

[ПРИМЕР 1. Инициализация при объявлении и вывод одномерного массива 14](#_Toc56035253)

[ПРИМЕР 2. Инициализация одномерного массива индексами. Ввод значений с клавиатуры 16](#_Toc56035254)

[ПРИМЕР 3. Инициализация одномерного массива случайными числами 17](#_Toc56035255)

[ПРИМЕР 4. Определение чётных элементов и чётных индексов в одномерном массиве 18](#_Toc56035256)

[ПРИМЕР 5. Выбор способа инициализации одномерного массива 19](#_Toc56035257)

[ПРИМЕР 6. Нахождение максимального значения в одномерном массиве. 21](#_Toc56035258)

[ПРИМЕР 7. Сортировка обменом 22](#_Toc56035259)

[ПРИМЕР 8. Сортировка вставкой. 23](#_Toc56035260)

[ПРИМЕР 9. Сортировка выбором. 24](#_Toc56035261)

[ПРИМЕР 10. Алгоритм «Решето Эратосфена» 25](#_Toc56035262)

[ПРИМЕР 11. Инициализация при объявлении и вывод двумерного массива 26](#_Toc56035263)

[ПРИМЕР 12. Инициализация и вывод двумерного массива 28](#_Toc56035264)

[ПРИМЕР 13. Сумма элементов *k* строки матрицы 30](#_Toc56035265)

[ПРИМЕР 14. Сумма элементов *k* столбца матрицы. 31](#_Toc56035266)

[ПРИМЕР 15. Нахождение максимального элемента в матрице 32](#_Toc56035267)

[ПРИМЕР 16. Сумма элементов треугольной части матрицы 33](#_Toc56035268)

[ПРИМЕР 17. Перестановка двух строк 34](#_Toc56035269)

[ПРИМЕР 18. Получить квадратную матрицу заданного порядка N 35](#_Toc56035270)

[ПРИМЕР 19. Указатели 36](#_Toc56035271)

[ПРИМЕР 20. Операции над указателями 37](#_Toc56035272)

[ПРИМЕР 21. Указатели на указатели 38](#_Toc56035273)

[ПРИМЕР 22. Имя массива - константный указатель. Одномерный массив. 39](#_Toc56035274)

[ПРИМЕР 23. Имя массива - константный указатель. Одномерный массив. 40](#_Toc56035275)

[ПРИМЕР 24. Указатели и одномерные массивы 41](#_Toc56035276)

[ПРИМЕР 25. Указатели и одномерные массивы 42](#_Toc56035277)

[ПРИМЕР 26. \*\*\* Указатели и одномерные массивы. 43](#_Toc56035278)

[ПРИМЕР 27. Имя массива - константный указатель. Двумерные массивы. 44](#_Toc56035279)

[ПРИМЕР 28. Имя массива - константный указатель. Двумерные массивы. 45](#_Toc56035280)

[ПРИМЕР 29. \*\*\* Указатели и двумерные массивы. 46](#_Toc56035281)

[ПРИМЕР 30. \*\*\* Указатели и двумерные массивы. 48](#_Toc56035282)

[ПРИМЕР 31. \*\*\* Массив указателей 50](#_Toc56035283)

[Словарь понятий, используемых в заданиях 52](#_Toc56035284)

# Массив

Массив – именованная нумерованная фиксированная по размеру последовательность данных одного типа, которые хранятся в непрерывной области памяти друг за другом.

Все элементы массива имеют один и тот же тип, который называют **базовым**.

Элементы массива могут быть как стандартных типов, таких как целый, вещественный, символьный, логический, так и структурированных типов, таких как массивы, структуры (записи) или объекты.

Члены последовательности данных называются элементами массива.

Номер элемента массива по одной размерности называется его **индексом**.

Число элементов массива называется **размером массива**.

Число элементов статического массива задаётся при его объявлении и в процессе выполнения программы **не меняется**.

Доступ к элементу массива производится путём указания имени массива и номера элемента.

Нумерация элементов может выполняться одной или несколькими последовательностями целых чисел – индексными последовательностями.

Если нумерация выполняется одной последовательностью говорят, что массив является одномерным, в противном случае – многомерным.

Нумерация элементов массива всегда начинается с 0, а номер каждого следующего члена больше номера предыдущего на 1.

Массив определяется:

* именем (идентификатором);
* количеством размерностей - числом номеров, необходимых для указания местонахождения элемента массива;
* размером (диапазоном изменения индексов) по каждой размерности.

В качестве индексов в С++ могут использоваться константы и переменные любых целых типов.

# Синтаксис описания одномерного массива

Имя\_типа\_элемента Имя\_массива [Размер];

Имя\_типа\_элемента Имя\_массива [Размер] = {список значений};

int a[4] = {5, 6, 7, 8};

5

6

7

8

a

…

a[0]

a[1]

a[2]

a[3]

…

5

6

7

8

…

…

double y[ ] = {5.5, 6.8, 8.8, 9.5, 10.3};

y

y[0]

y[1]

y[2]

y[3]

y[4]

5.5

6.8

8.8

9.5

10.3

…

…

5.5

6.8

8.8

9.5

10.3

…

…

char s[ ] = {‘a’, ‘b’, ‘+’};

…

s

…

‘a’

‘b’

‘+’

s[0]

s[1]

s[2]

…

...

double y[5];

y

…

…

y[0]

y[1]

y[2]

y[3]

y[4]

???

???

???

???

???

Синтаксис описания двумерного массива

Имя\_типа\_элемента Имя\_массива [Размер] [Размер];

Имя\_типа\_элемента Имя\_массива [Размер] [Размер] = {список значений};

Имя\_типа\_элемента Имя\_массива [Размер] [Размер]

= {{список значений0},{ список значений1},{ список значений2}};

double a[3][6];

a[0][0]

a[0][1]

a[0][2]

a[0][3]

a[0][4]

a[0][5]

a[1][0]

a[1][1]

a[1][2]

a[1][3]

a[1][4]

a[1][5]

a[2][0]

a[2][1]

a[2][2]

a[2][3]

a[2][4]

a[2][5]

Зависимости индексов элементов в двумерном массиве [n][n]

i < j

i = j

i = n - j -1

i < n - j -1

i > n - j -1

i > j

Сортировка массивов

*Сортировкой* или *упорядочением* [массива](http://www.vzmakh.ru/info/pascal/modules/page9.html) называется расположение его элементов по возрастанию (или убыванию). Если не все элементы различны, то надо говорить о *неубывающем* (или *невозрастающем*) порядке.

Существует много различных алгоритмов. Критерии оценки эффективности этих алгоритмов могут включать следующие параметры:

* количество шагов алгоритма, необходимых для упорядочения;
* количество сравнений элементов;
* количество перестановок, выполняемых при сортировке.

Сортировка обменом.

В основе алгоритма лежит обмен соседних элементов массива. Каждый элемент массива, начиная с первого, сравнивается со следующим, и если он больше следующего, то элементы меняются местами. Таким образом, элементы с меньшим значением продвигаются к началу массива (всплывают), а элементы с большим значением — к концу массива (тонут). Поэтому данный метод сортировки обменом иногда называют методом "пузырька". Этот процесс повторяется столько раз, сколько элементов в массиве, минус единица.

****Сортировка выбором****.

Идея алгоритма достаточно проста. Найдём среди элементов массива минимальный и поменяем его с первым элементом, затем найдём минимальный среди оставшихся и поменяем его со вторым элементом и т.д.

****Сортировка вставкой****.

На *k*-ом этапе "вставляем" *k*-ый элемент массива в нужную позицию среди уже отсортированных элементов от 0 до k-1. После этой вставки первые *k* элементов массива будут упорядочены.

Алгоритм «Решето Эратосфена»

В математике, решето́ Эратосфе́на — простой старинный [алгоритм](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) нахождения всех [простых чисел](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE) до некоторого целого числа *n*. Он был создан древнегреческим математиком [Эратосфеном](http://ru.wikipedia.org/wiki/%D0%AD%D1%80%D0%B0%D1%82%D0%BE%D1%81%D1%84%D0%B5%D0%BD).

Идея алгоритма Эратосфена: из списка всех чисел от 2 до n последовательно вычеркивать числа, кратные уже известным простым числам.

1. Из ряда чисел 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14... вычеркивают кратные двум; 4, 6, 8, 10, 12,...

2. — кратные трем: 6, 9, 12, 15,...

3. — кратные пяти: 10, 15, 20, 25, 30,...

4. — кратные семи: 14, 21, 28, 35, 42, 49,...

и т. д.

Таким образом все составные числа будут *просеяны*, и останутся только простые числа 2, 3, 5, 7, 11, 13...

Основная структура данных программы:

одномерный массив размера 2..n типа bool.

Индексы массива соответствуют анализируемым числам.

Элемент массива имеет значение true, если индекс этого элемента - простое число.

Арифметика больших чисел

Если в задаче возникает необходимость проводить арифметические операции над числами, для которых вещественный тип неприемлем, а типа *long* недостаточно, необходимо хранить числа в виде одномерного массива, каждый элемент которого представляет собой десятичную цифру числа.

Алгоритмы арифметических действий над такими числами сводятся к алгоритмам «столбиком».

Указатели

Каждый байт в памяти ЭВМ имеет адрес, по которому можно обратиться к определенному элементу данных

Указатель – это переменная, которая хранит адрес другой переменной определенного типа.

По описанию указателя компилятор получает информацию о том, какова длина области памяти, на которую ссылается указатель (которую занимает переменная, на которую он ссылается) и о том, как интерпретировать данные в этой области памяти.

Синтаксис объявления указателей

1. *Базовый\_тип \*Имя\_Указателя*;
2. *Базовый\_тип\* Имя\_Указателя*;

int \*pa, \*pb, \*pc;

int\* pa;

float\* pb;

Инициализация указателей

* Указатель можно инициализировать адресом переменной, которая уже определена:

double dvar = 0.0;

double \*pvar = &dvar;

* Инициализация значением NULL гарантирует, что указатель не содержит адреса, который воспринимается как корректный, а значение можно проверить в операторе if :

int \*pinteger = NULL;

if (pinteger == NULL)

cout << “pinteger is null”;

* равнозначная альтернатива инициализировать указатель 0:

int \*pinteger = 0;

if (!pinteger)

cout << “pinteger is null”;

Операции над указателями:

* присваивания,
* доступа по указателю (разыменование (\*)),
* инкремента или декремента,
* сложение с константой,
* вычитание двух указателей,
* сравнение указателей (одного типа),
* приведение типов,
* получения адреса (&)

Указатели и массивы

Объявляя массив, мы объявляем имя массива как указатель на его первый элемент, значение которого нельзя изменить (константный указатель).

Ссылки

* Ссылка (reference) – псевдоним для другой переменной.
* Ссылка имеет имя, которое может использоваться вместо имени переменной. Так как ссылка – это псевдоним, а не указатель, переменная, для которой она определяется, должна быть объявлена ранее.
* В отличие от указателя ссылка не может быть изменена, чтобы представлять другую переменную
* Ссылка, в отличие от указателя, не занимает дополнительного пространства в памяти и является просто другим именем величины.
* Объявление и инициализация:

*Базовый\_тип* &*Имя\_Ссылки* = *Имя\_Переменной;*

int number = 0;

int &rnumber = number; // ссылка

int \*pnumber = &number // указатель

* Ссылку, можно использовать вместо имени исходной переменной:

rnumber += 10;

\*pnumber += 10; // требуется разыменование

* Ссылка подобна указателю, который уже разыменован и значение которого нельзя изменить.

ПРИМЕР 1. Инициализация при объявлении и вывод одномерного массива

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 6;

// вывод 6 элементов одномерного массива а

int a[N] = {0, 1, 2, 3, 4, 5};

cout << " massiv a" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

// попытка вывода 16 элементов массива а

cout << endl << " massiv a" << endl;

for (int i = 0; i < N + 10; i++)

cout << setw(3) << a[i];

// попытка проинициализировать 12 элементов массива

int b[N] = {0, 1, 2, 3, 4, 5, 10, 11, 12, 13, 14, 15};

cout << endl << " massiv b" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << b[i];

// вывод 6 элементов одномерного массива с

int c[N] = {0, 1, 2};

cout << endl << " massiv c" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << c[i];

// вывод 6 элементов одномерного массива d

char d[] = {’0’, ’1’, ’2’, ’3’, ’4’, ’5’};

cout << endl << " massiv d" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << d[i];

// вывод 6 элементов одномерного массива е

int e[N] = {0};

cout << endl << " massiv e" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << e[i];

// вывод 6 элементов одномерного массива f

int f[N];

cout << endl << " massiv f" << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << f[i];

cout << endl;

}

ПРИМЕР 2. Инициализация одномерного массива индексами. Ввод значений с клавиатуры

#include <iomanip> // setw

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 6;

int a[N] = {0};

// вывод элементов массива в строку

cout << " 1 " << endl;

for (int i = 0; i < N; i++)

a[i] = 1;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

// вывод элементов массива в столбец

cout << endl << " 2 " << endl;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i] << endl;

// заполняем массив так, чтобы элементы массива

// были равны значениям индексов

cout << endl << " 3 " << endl;

for (int i = 0; i < N; i++)

a[i] = i;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

// вывод элементов в обратном порядке

cout << endl << " 4 " << endl;

for (int i = N - 1; i >= 0; i--)

cout << setw(3) << a[i];

cout << endl;

// ввод элементов массива с клавиатуры

cout << endl << " 5 " << endl;

for (int i = 0; i < N; i++)

cin >> a[i];

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

}

ПРИМЕР 3. Инициализация одномерного массива случайными числами

#include <iomanip> // setw

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <LIMITS.H> // SHRT\_MAX

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 6;

unsigned int a[N];

int b[N];

char c[N];

double d[N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < N; i++)

a[i] = rand() % 100; // 0..100

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl << " massiv b" << endl;

for (int i = 0; i < N; i++)

b[i] = rand() % 100 - rand() % 100; // -100..100

for (int i = 0; i < N; i++)

cout << setw(3) << b[i];

cout << endl << " massiv c" << endl;

for (int i = 0; i < N; i++)

c[i] = char(rand() % 40 + 30); // 30..70

for (int i = 0; i < N; i++)

cout << setw(3) << c[i];

cout << endl << " massiv d" << endl;

for (int i = 0; i < N; i++)

d[i] = rand() / double(SHRT\_MAX); // 0..1

for (int i = 0; i < N; i++)

cout << setw(13) << d[i];

cout << endl;

}

ПРИМЕР 4. Определение чётных элементов и чётных индексов в одномерном массиве

#include <iomanip> // setw

#include <time.h> // time

#include <stdlib.h> // rand srand

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 20;

int array[N];

srand(time(NULL));

// заполнение элементов массива случайными значениями

cout << " massiv " << endl;

for (int i = 0; i < N; i++)

array[i] = rand() % 100;

// вывод на экран элементов массива

for (int i = 0; i < N; i++)

cout << setw(3) << array[i];

cout << endl << endl;

// вывод на экран элементов массива с четными индексами

for (int i = 0; i < N / 2; i++)

cout << setw(3) << array[2 \* i];

cout << endl << endl;

// вывод на экран четных элементов массива

for (int i = 0; i < N; i++)

if ( !(array[i] % 2))

cout << setw(3) << array[i];

cout << endl;

}

ПРИМЕР 5. Выбор способа инициализации одномерного массива

#include <iomanip> // setw

#include <iostream> // cin cout

#include <clocale>

using namespace std;

int main()

{

const int N = 10;

int a[N] = { 0 };

enum enmArrayInit

{ Uninited = 0, WithConst, WithRand, ByUser, };

setlocale(LC\_ALL, "rus");

cout << " Выберите тип инициализации:\n"

<< WithConst << ".константами\n"

<< WithRand << ".случайными числами\n"

<< ByUser << ".ввести вручную\n";

cout << "Любая другая кнопка завершит программу\n Ваш выбор: ";

int ch = 0; cin >> ch;

switch (ch)

{

case WithConst:

{ // заполнение массива значениями индексов

for (int i = 0; i < N; i++) a[i] = i;

break;

}

case WithRand:

{ // Инициализация массива случайными числами

for (int i = 0; i < N; i++) a[i] = rand() % 100;

break;

}

case ByUser:

{ // ввод элементов массива с клавиатуры

for (int i = 0; i < N; i++) cin >> a[i];

break;

}

default: cout << "Программа завершается" << endl;

return 0;

}

for (int i = 0; i < N; i++) cout << setw(3) << a[i];

cout << endl;

return 0;

}

Или лучше так.

#include <iomanip> // setw

#include <iostream> // cin cout

#include <clocale>

using namespace std;

int main()

{ const int N = 10;

int a[N] = { 0 };

enum class enmArrayInit

{ Uninited = 0, WithConst, WithRand, ByUser, };

setlocale(LC\_ALL, "rus");

cout << " Выберите тип инициализации:\n"

<<(int)enmArrayInit::WithConst<< ".константами\n"

<<(int)enmArrayInit::WithRand<<".случайными числами\n"

<<(int)enmArrayInit::ByUser << ".ввести вручную\n";

cout << "Любая другая кнопка завершит программу\n Ваш выбор: ";

int ch = 0; cin >> ch;

switch ((enmArrayInit)ch)

{ case enmArrayInit::WithConst:

{ // заполнение массива значениями индексов

for (int i = 0; i < N; i++) a[i] = i;

break;

}

case enmArrayInit::WithRand:

{ // Инициализация массива случайными числами

for (int i = 0; i < N; i++) a[i] = rand() % 100;

break;

}

case enmArrayInit::ByUser:

{ // ввод элементов массива с клавиатуры

for (int i = 0; i < N; i++) cin >> a[i];

break;

}

default: cout << "Программа завершается" << endl;

return 0;

}

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

return 0;

}

ПРИМЕР 6. Нахождение максимального значения в одномерном массиве.

#include <iomanip> // setw

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 20;

int array[N];

srand(time(NULL));

cout << " massiv " << endl;

for (int i = 0; i < N; i++)

array[i] = rand() % 100;

for (int i = 0; i < N; i++)

cout << setw(3) << array[i];

cout << endl;

int max = array[0];

int k = 0;

for (int i = 1; i < N; i++)

if (max < array[i])

{

max = array[i];

k = i;

}

cout << "max = " << max << " i = " << k << endl;

// в k - номер элемента, содержащего max

// сумма элементов после максимального значения

int sum = 0;

for (int i = k +1; i < N; i++)

sum += array[i];

cout << "sum = " << sum << endl;

}

ПРИМЕР 7. Сортировка обменом

Отсортировать заданный массив в порядке возрастания элементов.

#include <iomanip> // setw

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 20;

int a[N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < N; i++) // заполнение массива

a[i] = rand() % 100;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i]; // вывод массива на экран

cout << endl;

//повторять проходы по массиву N-1 раз

for (int i = 1; i < N; i++)

//проход с N-1-го элемента вверх до i-го

for (int j = N -1; j >= i; j--)

// обмен элементов в случае неправильного порядка

if (a[j-1] > a[j])

{

int x = a[j-1];

a[j-1] = a[j];

a[j] = x;

}

// вывод отсортированного массива на экран

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

}

ПРИМЕР 8. Сортировка вставкой.

Отсортировать заданный массив в порядке возрастания элементов.

#include <iomanip> // setw

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 20;

int a[N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < N; i++)

a[i] = rand() % 100;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

// перебор не отсортированных элементов массива

for (int i = 1; i < N; i++)

{

int x = a[i]; //взятие очередного элемента

int j = i -1;

// повторять пока место вставки не найдено

while (x < a[j])

{

// сдвиг текущего j-го элемента на 1 позицию вправо

a[j+1] = a[j];

j--;

// условие выхода при достижении левой границы

if (j < 0)

break;

}

a[j+1] = x; // вставка взятого элемента

}

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

}

ПРИМЕР 9. Сортировка выбором.

Отсортировать заданный массив в порядке возрастания элементов.

#include <iomanip> // setw

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 20;

int a[N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < N; i++)

a[i] = rand() % 100;

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

for (int i = 0; i < N -1; i++)

{ //присвоение переменной минимум текущего элемента

int min = a[i];

// запоминаем индекс текущего элемента

int index\_min = i;

// поиск минимума в части массива от i+1 до конца

for (int j = i +1; j < N; j++)

if (a[j] < min)

{ // запоминаем текущий найденный минимум

min = a[j];

index\_min = j; // запоминаем его индекс

}

// обмен местами текущего элемента и

// найденного минимального

a[index\_min] = a[i];

a[i] = min;

}

for (int i = 0; i < N; i++)

cout << setw(3) << a[i];

cout << endl;

}

ПРИМЕР 10. Алгоритм «Решето Эратосфена»

Найти простые числа меньше заданного N.

#include <math.h>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 200;

bool a[N]={0};

// заполним все ячейки числом 1

for (int i = 2; i < N; i++)

a[i] = true;

unsigned int n = sqrt(double(N));

for (int i = 2; i <= n; i++)

if (a[i]) // вычеркиваем (обнуляем) кратные числа

for (int j = i \*2; j < N; j += i)

a[j] = false;

for (int i = 0; i < N; i++)

if (a[i])

cout << i << " "; // вывод простых чисел

cout << endl;

}

ПРИМЕР 11. Инициализация при объявлении и вывод двумерного массива

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

// инициализация массива при объявлении

int a[M][N] = { { 0, 1, 2, 3, 4, 5},

{ 6, 7, 8, 9,10,11},

{12,13,14,15,16,17},

{18,19,20,21,22,23}

};

cout << " massiv a" << endl;

// вывод матрицы в виде таблицы

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

// инициализация массива при объявлении

int b[M][N] = {0,1,2,3,4,5,10,11,12,13,14,15};

// вывод матрицы в виде таблицы

cout << endl << " massiv b" << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << b[i][j];

cout << endl;

}

// инициализация массива при объявлении

int c[][N] = { {0,1,2,3,4,5},

{10,11,12,13,14,15},

{20,21,22,23,24,25},

{30,31,32,33,34,35}

};

// вывод матрицы в виде таблицы

cout << endl << " massiv c" << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << c[i][j];

cout << endl;

}

int d[M][N] = {0}; // инициализация 0

// вывод матрицы в виде таблицы

cout << endl << " massiv d" << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << d[i][j];

cout << endl;

}

int e[M][N]; // нет инициализации

// вывод матрицы в виде таблицы

cout << endl << " massiv e" << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << e[i][j];

cout << endl;

}

}

ПРИМЕР 12. Инициализация и вывод двумерного массива

#include <iomanip>

#include <stdlib.h> // rand

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

int a[M][N];

cout << " 1 " << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

a[i][j] = 1; // заполнение 1

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl << " 2" << endl;

for (int i = 0; i < M; i++)

for (int j = 0;j < N; j++)

a[i][j] = i; // заполнение номером строки

for (int i =0; i < M; i++)

{

for (int j =0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl << " 3" << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

a[i][j] = j; // заполнение номером столбца

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl << " 4 " << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++) // заполнение числами

a[i][j] = i \* N + j; // от 0..nּm-1

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl << " 5 " << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++) // заполнение случайными

a[i][j] = rand() % 100; // числами 0..99

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl << " 6 " << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++) // значения элементов

cin >> a[i][j]; // матрицы вводятся с

// клавиатуры

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

cout << endl;

}

ПРИМЕР 13. Сумма элементов *k* строки матрицы

#include <iomanip>

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

int k;

int a[M][N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

a[i][j] = rand() % 10; // заполнение

for (int i = 0; i < M; i++)

{

for (int j =0; j < N; j++)

cout << setw(3) << a[i][j]; // вывод на экран

cout << endl;

}

cout << " k=";

cin >> k; // номер строки

if (k >= 0 && k < M)

{

int sum = 0;

for (int i = 0; i < N; i++)

sum += a[k][i];

cout << " summa elementov " << k << " stroki = "

<< sum << endl;

}

}

ПРИМЕР 14. Сумма элементов *k* столбца матрицы.

#include <iomanip>

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

int k;

int a[M][N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < M; i++)

for (int j =0; j < N; j++)

a[i][j] = rand() % 10; // заполнение

for (int i = 0; i < M ;i++)

{

for (int j =0; j < N; j++)

cout << setw(3) << a[i][j]; // вывод на экран

cout << endl;

}

cout << " k="; cin >> k; // номер столбца

if (k >= 0 && k < N)

{

int sum = 0;

for (int i = 0; i < M; i++)

sum += a[i][k];

cout << " summa elementov " << k << " stolbca = "

<< sum << endl;

}

}

ПРИМЕР 15. Нахождение максимального элемента в матрице

#include <iomanip>

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

int a[M][N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

a[i][j] = rand() % 10; // заполнение

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j]; // вывод на экран

cout << endl;

}

int max = a[0][0];

int k = 0;

int l = 0;

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

if (max < a[i][j])

{

max = a[i][j];

k = i; // номер строки

l = j; // номер столбца

}

cout << " max element = " << max

<< " v " << k << " stroke v "

<< l << " stolbce" << endl;

}

ПРИМЕР 16. Сумма элементов треугольной части матрицы

Найти сумму элементов на главной диагонали матрицы и сумму элементов закрашенной части матрицы.

#include <iomanip>

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int N = 5;

int a[N][N];

srand(time(NULL));

cout << " massiv a" << endl;

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

a[i][j] = rand() % 10; // заполнение

for (int i = 0; i < N; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j]; // вывод на экран

cout << endl;

}

int sum = 0;

for (int i = 0; i < N; i++)

sum += a[i][i];

cout << endl << "summa elementov glavnoi diagonali = "

<< sum << endl << endl;

sum = 0;

for (int i = 0; i < N; i++)

for (int j = i; j < N; j++)

sum += a[i][j];

cout << "summa elementov verxnego pravogo "

<< "treygolnika = " << sum << endl;

}

ПРИМЕР 17. Перестановка двух строк

Поменять местами элементы двух строк матрицы.

#include <iomanip>

#include <stdlib.h> // rand srand

#include <time.h> // time

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 4, N = 6;

int k = 0, l = 0;

int a[M][N];

srand(time(NULL));

for (int i = 0; i < M; i++)

for (int j = 0; j < N; j++)

a[i][j] = rand() % 10; // заполнение

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j]; // вывод на экран

cout << endl;

}

cout << " nomer 1 stroki =";

cin >> k;

cout << " nomer 2 stroki =";

cin >> l;

if (k >= 0 && k < M && l >= 0 && l < M)

for (int i = 0; i < N; i++)

{

int p = a[k][i];

a[k][i] = a[l][i];

a[l][i] = p;

}

// вывод на экран результата

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

}

ПРИМЕР 18. Получить квадратную матрицу заданного порядка N

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | … | 3 | 2 | 1 |
| 0 | 1 | 2 | … | 2 | 1 | 0 |
|  |  |  | … |  |  |  |
| 0 | 1 | 2 | … | 2 | 1 | 0 |
| 1 | 2 | 3 | … | 3 | 2 | 1 |

void main()

{

const int N = 7;

int a[N][N]={0};

for (int i = 0; i < (N+1)/2; i++)

for (int j = 0; j < (N+1)/2; j++)

{

int b = j + 1 - i;

if (b < 0)

b = 0;

a[i][j] = b;

a[i][N-j-1] = b;

a[N-i-1][j] = b;

a[N-i-1][N-j-1]= b;

}

for (int i = 0; i < N; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << a[i][j];

cout << endl;

}

}

ПРИМЕР 19. Указатели

#include <iostream> // cin cout

using namespace std;

void main()

{

cout << " 1 " << endl;

int var1 = 11, var2 = 22;

cout << (&var1) << " " << \*(&var1) << " " << var1 << endl;

\*(&var2) = 11;

cout << (&var2) << " " << \*(&var2) << " " << var2 << endl;

cout << " 2 " << endl;

int p1, p2 = 100;

int \*uptr;

uptr = &p2;

p1 = \*uptr \* 5;

cout << " p1= " << p1 << " p2 = " << p2

<< " uptr=" << uptr << endl << endl;

}

ПРИМЕР 20. Операции над указателями

#include <iostream> // cin cout

using namespace std;

void main()

{

int v[3]={1,2,3};

int \*uv = 0;

uv = &v[0];

cout << \*uv << endl;

uv += 1;

cout << \*uv << endl;

uv += 1;

cout << \*uv << endl;

}

ПРИМЕР 21. Указатели на указатели

#include <iostream> // cin cout

using namespace std;

void main()

{

int i = 123;

int \*pi = &i; // pi – указатель на переменную,

// инициализируется адресом i

int \*\*ppi = &pi;

// ppi – указатель на указатель на переменную

// инициализируется адресом указателя pi

int \*\*\*pppi = &ppi;

// pppi – указатель на указатель на указатель на переменную

// инициализируется адресом указателя ppi

cout << i << endl;

cout << \*pi << endl;

cout << \*\*ppi << endl;

cout << \*\*\*pppi << endl;

}

ПРИМЕР 22. Имя массива - константный указатель. Одномерный массив.

#include <iostream> // cin cout

using namespace std;

void main()

{

int a []={0,1,2,3,4,5,6,7};

cout << " a=" << a << " " << &a[0] << endl;

cout << " \*a= " << \*a << " " << a[0] << endl;

cout << " a+1=" << a + 1 << " " << &a[1] << endl;

cout << " \*(a+1)= " << \*(a + 1) << " " << a[1]

<< endl;

// a++; // нельзя - константный указатель

}

ПРИМЕР 23. Имя массива - константный указатель. Одномерный массив.

#include <iostream> // cin cout

using namespace std;

void main()

{

int a[]={1,2,3,4};

// вывод на экран элементов массива

cout << "1 " << endl;

for (int i = 0; i < 4; i++)

cout << a[i];

// можно так вывод на экран элементов массива

cout << endl << "2 " << endl;

for (int i = 0; i < 4; i++)

cout << i[a];

// можно так вывод на экран элементов массива

cout << endl << "3 " << endl;

for (int i = 0; i < 4; i++)

cout << \*(a + i);

// можно так вывод на экран элементов массива

cout << endl << "4 " << endl;

for (int i = 0; i < 4; i++)

cout << \*(i + a);

// нельзя так вывод на экран элементов массива

cout << endl << "5 " << endl;

for (int i = 0; i < 4; i++)

{

cout << \*a;

a += 1; // нельзя константный указатель

}

// нельзя так вывод на экран элементов массива

cout << endl << "6 " << endl;

for (int i = 0; i < 4; i++)

cout << \*(a++); // нельзя константный указатель

cout << endl;

}

ПРИМЕР 24. Указатели и одномерные массивы

#include <iostream> // cin cout

using namespace std;

void main()

{

int a []={0,1,2,3,4,5,6,7};

int \*pa;

pa = &a[0]; // pa = a;

cout << " pa =" << pa << " a=" << a << " " << &a[0]

<< endl;

cout << " pa+1=" << pa + 1 << " " << &a[1] << endl;

cout << " \*(pa+1)= " << \*(pa + 1) << " " << pa[1]

<< " " << a[1] << endl;

cout << " ++pa=" << ++pa << " " << &a[1] << endl;

// нельзя константный указатель

cout << " ++a=" << ++a << " " << &a[1] << endl;

}

ПРИМЕР 25. Указатели и одномерные массивы

#include <iostream> // cin cout

using namespace std;

void main()

{

int a[] = {1,2,3,4};

int \*pint = NULL;

cout << "1 " << endl;

pint = a;

for (int i = 0; i < 4; i++)

cout << pint[i];

cout << endl << "2 " << endl;

pint = a;

for (int i = 0; i < 4; i++)

cout << \*(pint + i);

cout << endl << "3 " << endl;

pint = a;

for (int i = 0; i < 4; i++)

{

cout << \*pint;

pint += 1;

}

cout << endl << "4 " << endl;

pint = a;

for (int i = 0; i < 4; i++)

cout << \*(pint++);

cout << endl << "5 " << endl;

pint = a;

for (int i = 0; i < 4; i++)

cout << (\*pint)++;

cout << endl;

}

ПРИМЕР 26. \*\*\* Указатели и одномерные массивы.

#include <iostream> // cin cout

using namespace std;

void main()

{

int a[] = {1,2,3,4};

cout << "1 " << endl;

for (int i = 0, \*pint = &a[0]; i < 4; i++)

cout << pint[i];

cout << endl << "2 " << endl;

for (int \*pint = &a[0]; pint <= &a[3]; pint++)

cout << \*pint;

cout << endl << "3 " << endl;

for (int \*pint = a, i = 0; pint + i <= a + 3; pint++)

cout << \*pint;

cout << endl << "4 " << endl;

for (int \*pint = a + 3; pint >= a; pint--)

cout << \*pint;

cout << endl << "5 " << endl;

for (int \*pint = a, i = 0; pint <= a + 3; i++)

cout << \*(pint++);

cout << endl;

}

ПРИМЕР 27. Имя массива - константный указатель. Двумерные массивы.

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M=3, N=4;

int k[M][N] = { {77, 1, 2, 3},

{10, 11, 12, 13},

{20, 21, 22, 23}

};

// adress elementa [0][0]

cout << " adress [0][0]" << endl;

cout << setw(12) << k

<< setw(12) << &k

<< setw(12) << &k[0][0] << endl;

// value elementa [0][0]

cout << " value [0][0]" << endl;

cout << setw(5) << \*\*k

<< setw(5) << k[0][0]

<< setw(5) << \*(&k[0][0]) << endl;

// adress elementa [1][0]

cout << " adress [1][0]" << endl;

cout << setw(12) << k[1]

<< setw(12) << k+1

<< setw(12) << \*(k+1)

<< setw(12) << &k[1][0] << endl;

// value elementa [1][0]

cout << " value [1][0]" << endl;

cout << setw(5) << \*k[1]

<< setw(5) << k[1][0]

<< setw(5) << \*\*(k+1)

<< setw(5) << \*(&k[1][0]) << endl;

// value elementa [1][2]

cout << " value [1][2]" << endl;

cout << setw(5) << k[1][2]

<< setw(5) << \*(\*(k+1)+2)

<< setw(5) << \*(k[1]+2) << endl;

}

ПРИМЕР 28. Имя массива - константный указатель. Двумерные массивы.

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 3, N= 4;

int k [M][N] = { {0, 1, 2, 3},

{10, 11, 12, 13} ,

{20, 21, 22, 23}

};

cout << " 1 " << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << k [i][j];

cout << endl;

}

cout << endl << " 2 " << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << \* (\*(k + i) + j) ;

cout << endl;

}

cout << endl << " 3" << endl;

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << \*(k[i] + j) ;

cout << endl;

}

cout << endl;

}

ПРИМЕР 29. \*\*\* Указатели и двумерные массивы.

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M = 3, N = 4;

int k[M][N] = { {0, 1, 2, 3},

{10, 11, 12, 13},

{20, 21, 22, 23}

};

cout << " 1 " << endl;

int \*pi = &k[0][0];

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << \*(pi + (j + i \* N));

cout << endl;

}

cout << endl << " 2 " << endl;

pi = k[0];

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << \*(pi + (j + i \* N));

cout << endl;

}

cout << endl << " 3 " << endl;

for (int i = 0; i < M; i++)

{

int\* p = k[i];

for (int j = 0; j < N; j++)

cout << setw(3) << p[j];

cout << endl;

}

cout << endl << " 4 " << endl;

for (int i = 0; i < M; i++)

{

int\* p = \*(k + i);

for (int j = 0; j < N; j++)

cout << setw(3) << \*(p + j);

cout << endl;

}

cout << endl;

}

ПРИМЕР 30. \*\*\* Указатели и двумерные массивы.

#include <iomanip>

#include <stdlib.h> // new

#include <iostream> // cin cout

using namespace std;

void main()

{

const int M=3, N=4;

int k [M][N] = {{0, 1, 2, 3},

{10, 11, 12, 13},

{20, 21, 22, 23}};

cout << " 1 " << endl;

int \*pi[M];

for (int i = 0; i < M; i++)

pi[i] = k[i];

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << pi[i][j];

cout << endl;

}

cout << endl << " 2 " << endl;

for (int i = 0; i < M; i++)

pi[i] = &k[i][0];

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << pi[i][j];

cout << endl;

}

cout << endl << " 3 " << endl;

int \*p = &k[0][0];

int \*\*ppi = &p;

for (int i = 0; i < M; i++)

\*(ppi + i) = \*(k + i);

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << ppi[i][j];

cout << endl;

}

cout << endl;

cout << " 4 " << endl;

int \*\*ppi = new int\*[M];

for (int i = 0; i < M; i++)

\*(ppi + i) = \*(k + i);

for (int i = 0; i < M; i++)

{

for (int j = 0; j < N; j++)

cout << setw(3) << ppi[i][j];

cout << endl;

}

cout << endl;

}

ПРИМЕР 31. \*\*\* Массив указателей

#include <iomanip>

#include <iostream> // cin cout

using namespace std;

void main()

{

int a;

int \*b; // указатель на int

int c[5]; // массив из 5 элементов типа int

int \*d[5]; // массив из 5 указателей на int

int (\*e)[5]; // указатель на массив из 5 int элементов

b = &a;

\*b = 5;

cout << " \*b " << \*b << endl << endl;

cout << " c " << endl;

for (int i = 0; i < 4; i++)

\*(c + i) = i;

for (int i = 0; i < 4; i++)

cout << \*(c + i) << " ";

cout << endl << endl;

cout << " d " << endl;

for (int i = 0; i < 4; i++)

d[i] = c + i;

for (int i = 0; i < 4; i++)

\*d[i] = i;

// for (int i = 0; i < 4; i++)

// (\*d)[i] = i;

for (int i = 0; i < 4; i++)

cout << \*d[i] << " ";

cout << endl << endl;

cout << " d " << endl;

for (int i = 0; i < 4; i++)

d[i] = c + i;

for (int i = 0; i < 4; i++)

\*(\*d + i) = i;

for (int i = 0; i < 4; i++)

cout << \*(\*d + i) << " ";

cout << endl << endl;

cout << " e " << endl;

e = &c;

for (int i = 0; i < 4; i++)

(\*e)[i] = i;

for (int i = 0; i < 4; i++)

cout << (\*e)[i] << " ";

cout << endl << endl;

cout << " e " << endl;

e = &c;

for (int i = 0; i < 4; i++)

\*(\*e + i) = i;

for (int i = 0; i < 4; i++)

cout << \*(\*e + i) << " ";

cout << endl << endl;

}

Словарь понятий, используемых в заданиях

**Последовательность** – такой способ организации однотипных данных, называемых элементами последовательности, при котором последовательность может быть либо **пустой** (тогда она не содержит ни одного элемента), либо **непустой**. В последнем случае последовательность содержит **первый** элемент, а для каждого элемента, кроме **последнего**, имеется ровно один **следующий** элемент. Таким образом, для каждого элемента последовательности, кроме первого, имеется ровно один **предыдущий** элемент. Последовательность в отличие от множества может содержать совпадающие элементы.

**Начальным отрезком** после­довательности *w* называется всякая последовательность *v*, для которой существует такая последовательность *у*, что *w* = *vy*.

Например, 101, 010, 0, 1 и 101001 – (различные) подпоследова­тельности последовательности 101001, причем 101, 1 и 101001 – начальные отрезки этой последовательности. Пустая последова­тельность, как следует из этих определений, является подпосле­довательностью и начальным отрезком любой последовательности элементов соответствующего типа.

**Серия –** последовательность, составленная из повторяющихся подпоследовательностей. Например, 123123123 – серия из подпоследовательности 123 .

**Вектор** – непустая последовательность элементов, пронумеро­ванных целыми числами.

**Перестановка длины *n*** – вектор, составленный из попарно различных элементов, каждый из которых – целое число из отрезка 1 ... *n*. Например, 3241 – перестановка длины 4.

**Матрица** – вектор, элементами которого являются векторы.

**Соседями** элемента *aij* в матрице назовем элементы *akl*, где *i* – 1 ≤ *k* ≤ *i* + 1, *j* ‑ 1 ≤ *l* ≤ *j* + 1, (*k*, *l*) ≠ (*i*, *j*).

Элемент матрицы называется **локальным минимумом (максимумом)**, если он строго меньше (больше) всех имеющихся у него соседей.

**Магическим квадратом** порядка *n* называется квадратная таблица размера *n* × *n*, составленная так, что суммы по каждому столбцу, каждой строке и каждой из двух диагоналей равны между собой.

**Латинским квадратом** порядка *n* называется квадратная таблица размера *n* × *n*, каждая строка и каждый столбец которой содержит числа 1, 2, …, *n*.

Матрица размера n x n называется квадратной.

Симметричная матрица – матрица , у которой равны элементы, расположенные симметрично относительно главной диагонали.

Треугольной называется квадратная матрица A, если из i > j следует a ij= 0.

Диагональной называется квадратная матрица A, если из i ≠ j следует aij = 0.

Матрица размера m x n, все элементы которой равны нулю, называется нулевой.

Диагональная матрица размера n x n с единичными диагональными элементами называется единичной матрицей.

Скалярное произведение двух векторов A (a1, a2, … , an) и B (b1, b2, … , bn) есть число, равное сумме попарных произведений их компонент:

Aּ B = a1ּ b1 + a2 ּb2 + … +anּ bn.