5- Массивы

Массивы (часть 1)

Часто возникает необходимость хранить не одну переменную, а набор однотипных переменных. Например, список учащихся класса – это набор данных строкового типа, координаты вершин многоугольника или коэффициенты многочлена – это набор числовых данных. Для хранения наборов данных используются *структуры данных*. Основная структура данных – это массив.

Массив — это структура однотипных данных, занимающих непрерывную область памяти. Массив имеет размер — количество элементов в нем. Каждый элемент массива имеет свой номер (также называемый *индексом*), обращение к элементу массива осуществляется путем указания его индекса. В языке C++ элементы нумеруются, начиная с 0, поэтому последний элемент массива имеет номер на 1 меньше размера массива.

Массив в языке C++ задается следующим образом:

```
тип_элементов идентификатор[размер];
```

где *тип_элементов* — произвольный тип данных языка C++, который будут иметь элементы массива, например, `int`, `double` и т.д.; *идентификатор* — имя массива, *размер* — число элементов в нем.

К элементу массива можно обращаться, как *идентификатор*`[`*индекс*`]`. Например, если было сделано объявление

```
double A[5];
```

то таким образом создается 5 элементов массива типа `double`: `A[0]`, `A[1]`, `A[2]`, `A[3]`, `A[4]`.

Пример программы, которая создает массив типа `int[]`, заданного пользователем размера, считывает с клавиатуры его элементы, затем прибавляет к каждому элементу массива число `1`, затем выводит результат на экран:

```
#include<iostream>
using namespace std;
int main()
{
    int n;          // Размер массива
    int i;          // Счетчик в циклах

    cout<<"Введите количество чисел: ";
    cin>>n;         // Считываем размер массива

    int arr[n];     // Объявление массива

                    // Считываем массив
    cout<<"Введите "<<n<<" целых чисел: ";
    for(i=0;i<n;++i)
        cin>>arr[i];
```

```
                    // Прибавляем по 1 к каждому элементу
    for(i=0;i<n;++i)
     arr[i]+=1;

                    // Выводим массив на экран
    for(i=0;i<n;++i)
        cout<<arr[i]<<" ";
                    // Переведем курсор на новую строку
    cout<<endl;
    return 0;
}
```

В этом примере при помощи `//` обозначается начало комментария, весь текст после начала комментария и до конца строки компилятором игнорируется. Второй способ объявления комментария: в начале комментария поставить знаки `/*`, а в конце – `*/`. Это позволяет делать комментарии, занимающие несколько строк. В языке C допустимы только такие комментарии.

Упражнения

В упражнениях подразумевается, что массив вводится, как в приведенном примере (сначала – количество элементов в массиве, потом – все элементы массива по одному). Все массивы – числовые типа `int[]`.

1. (A) Выведите на экран все элементы массива с четными индексами (то есть `A[0]`, `A[2]`, `A[4]`, ...).
2. (B) Выведите на экран все четные элементы массива (то есть те элементы, которые являются четными числами).
3. (C) Найдите количество положительных элементов в массиве. Программа должна считать массив и вывести единственное число – количество положительных элементов в массиве.
4. (D) Дан массив. Найдите количество элементов массива, которые больше предыдущего элемента в этом массиве.
5. (E) Массив содержит только ненулевые числа. Определите, есть ли в данном массиве два соседних элемента с одинаковыми знаками. Программа должна вывести `YES`, если есть два числа с одинаковыми знаками и `NO` иначе.
6. (F) Дан массив. Определите, сколько в этом массиве элементов, которые строго больше обоих своих соседей.
7. (G) Дан массив. Переставьте элементы массива в обратном порядке без использования дополнительного массива. Программа должна считать массив, поменять порядок его элементов, вывести результат на экран (просто вывести элементы массива в обратном порядке – недостаточно!).
8. (H) Переставьте соседние элементы массива (0-й элемент поменять с 1-м, 2-й с 3-м и т.д. Если элементов нечетное число, то последний элемент остается на своем месте)
9. (I) Циклически сдвиньте элементы массива вправо (0-й элемент становится 1-м, 1-й становится 2-м, ..., последний становится 0-м, то есть массив {3, 5, 7, 9} превращается в массив {9, 3, 5, 7}).
10. (J) Найдите значение наибольшего элемента в массиве.
11. (K) Дан массив, отсортированный по возрастанию (каждый элемент массива не меньше предыдущего элемента, например, `{1, 2, 2, 3, 3, 3}`). Найдите количество различных чисел в этом массиве.

Массивы (часть 2)

Передача массива в качестве параметра

Массивы можно передавать функции в качестве параметра. Но при этом размер создаваемого массива может быть неопределен на момент компиляции программы, поэтому функция не может знать размер полученного массива. Поэтому при объявлении функции необходимо задавать два параметра: массив передаваемых элемента (без указания размера массива) и размер массива. Например, функция поиска наименьшего значения в массиве `double A[n]` может быть объявлена так:

```
double Min (double A[], int n)
```

Соответственно, внутри функции `main` мы объявляем массив `double A[n]` и вызываем функцию `Min`, передав в качестве параметров массив `A` и его размер `n`:

```
Min (A, n);
```

Упражнения

Все решения задач необходимо оформлять в виде соответствующих функций. Также программа должна содержать функцию `main`, создающую и считывающую массив и выводящую информацию о результате работы функции.

1. (A) Напишите функцию `int Search (double A[], int n, double x)`, которая находит в массиве `double A[n]` элемент, значение которого равно `x`. Функция возвращает индекс найденного элемента или -1, если такого элемента в массиве нет. Программа получает на вход количество элементов в массиве n, затем n целых чисел – элементы массива, затем число x и должна вывести индекс найденного элемента или число -1, если данный элемент в массиве отсутствует.
2. (B) Напишите функцию `int CountMax (double A[], int n)`, которая подсчитывает, сколько раз в массиве встречается значение, являющееся максимальным. Функция должна выполнять **однократный просмотр** массива. Программа получает на вход количество элементов в массиве n, затем n целых чисел – элементы массива и должна вывести единственное целое число – сколько раз в массиве встречается значение, равное максимальному.
3. (C) Дан массив, состоящий из различных целых чисел. Определите второй по величине элемент этого массива. Напишите для этого функцию`double SecondMax (double A[], int n)`. Функция должна выполнять однократный просмотр массива.
4. (D) Напишите функцию `double SecondMax (double A[], int n)`, которая находит второй по величине элемент в массиве – тот элемент, который будет наибольшим, если из массива удалить наибольший элемент, то есть для массивов `{1, 2, 3, 4}` и `{1, 2, 3, 3}` вторым по величине элементом будет 3. Функция должна выполнять **однократный просмотр** массива.
5. (E) Напишите функцию `double SecondMaxValue (double A[], int n)`, которая находит второе по величине значение в массиве – значение элемента, который будет наибольшим, если из массива удалить наибольший элемент и все ему равные, то есть для массива `{1, 4, 3, 4}`вторым по величине значением будет 3.
6. (F) Даны два отсортированных массива: `int A[n]` и `int B[m]`.

   Объедините их в один отсортированный массив `int C[n+m]`, то есть если `A={1, 4, 6, 7}`, `B={2, 3, 5}`, то `C={1, 2, 3, 4, 5, 6, 7}`. Оформите алгоритм в виде функции `void merge (int A[], int n, int B[], int m, int C[])`. Время работы алгоритма должно быть порядка n+m действий.

Формат входных данных: число n, затем n возрастающих чисел – элементы первого массива, число m, затем m возрастающих чисел – элементы второго массива. Программа должна вывести n+m чисел, полученных объединением двух массивов в порядке возрастания.

Пример:

Вход

```
3
1 5 7
4
2 4 4 5
```

Выход

```
1 2 4 4 5 5 7
```

Задача №63. A[0], A[2], A[4], ...

Дан массив, состоящий из целых чисел. Нумерация элементов начинается с 0. Напишите программу, которая выведет элементы массива, номера которых четны (0, 2, 4...).

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 100$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести все элементы массива с чётными номерами.

Примеры

входные данные

```
6
4 5 3 4 2 3
```

выходные данные

```
4 3 2
```

Задача №64. Вывести четные элементы

Дан массив, состоящий из целых чисел. Напишите программу, которая выводит те элементы массива, которые являются чётными числами.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 100$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести все четные элементы массива (то есть те элементы, которые являются четными числами).

Примеры

входные данные

```
5
1 2 3 4 5
```

выходные данные

```
2 4
```

Задача №65. Количество положительных элементов

Дан массив, состоящий из целых чисел. Напишите программу, которая подсчитывает количество положительных чисел среди элементов массива.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 10000$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо единственное число - количество положительных элементов в массиве.

Примеры

входные данные

```
5
1 2 3 -1 -4
```

выходные данные

```
3
```

Задача №66. Количество элементов, больших предыдущего

Дан массив, состоящий из целых чисел. Напишите программу, которая подсчитает количество элементов массива, больших предыдущего (элемента с предыдущим номером).

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 10000$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести единственное число - количество элементов массива, больших предыдущего.

Примеры

входные данные

```
5
1 2 3 4 5
```

выходные данные

```
4
```

Задача №67. Есть ли два элемента с одинаковыми знаками

Дан массив, состоящий из целых чисел. Напишите программу, которая определяет, есть ли в массиве пара соседних элементов с одинаковыми знаками.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 10000$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести слово *YES*, если существует пара соседних элементов с одинаковыми знаками. В противном случае следует вывести слово *NO*.

Примеры

входные данные

```
5
1 -3 4 -2 1
```

выходные данные

```
NO
```

Задача №68. Количество элементов больших обоих соседей

Дан массив, состоящий из целых чисел. Напишите программу, которая в данном массиве определит количество элементов, у которых два соседних и, при этом, оба соседних элемента меньше данного.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \le N \le 100$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести количество элементов массива, у которых два соседа и которые при этом строго больше обоих своих соседей.

Примеры

входные данные

```
5
1 2 3 4 5
```

выходные данные

```
0
```

входные данные

```
5
1 5 1 5 1
```

выходные данные

```
2
```

Задача №69. Переставить элементы в обратном порядке

Напишите программу, которая переставляет элементы массива в обратном порядке без использования дополнительного массива. Программа должна считать массив, поменять порядок его элементов, затем вывести результат *(просто вывести элементы массива в обратном порядке – недостаточно!)*

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 35$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести массив, полученный после перестановки элементов.

Примеры

входные данные

```
6
4 5 3 4 2 3
```

выходные данные

```
3 2 4 3 5 4
```

Задача №70. Переставить соседние элементы

Напишите программу, которая переставляет соседние элементы массива (1-й элемент поменять с 2-м, 3-й с 4-м и т.д. Если элементов нечетное число, то последний элемент остается на своем месте).

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 35$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести массив, полученный после перестановки элементов.

Примеры

входные данные

```
6
4 5 3 4 2 3
```

выходные данные

```
5 4 4 3 3 2
```

Задача №71. Циклический сдвиг вправо

Напишите программу, которая циклически сдвигает элементы массива вправо (например, если элементы нумеруются, начиная с нуля, то 0-й элемент становится 1-м, 1-й становится 2-м, ..., последний становится 0-м, то есть массив {3, 5, 7, 9} превращается в массив {9, 3, 5, 7}).

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 35$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести массив, полученный после сдвига элементов.

Примеры

входные данные

```
6
4 5 3 4 2 3
```

выходные данные

```
3 4 5 3 4 2
```

Задача №72. Максимум в массиве

Вводится массив, состоящий из целых чисел. Найти наибольшее среди них.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 35$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел.

Выходные данные

Необходимо вывести значение наибольшего элемента в массиве.

Примеры

входные данные

```
3
1 2 3
```

выходные данные

```
3
```

Задача №73. Количество различных элементов в монотонном массиве

Дан массив, состоящий из целых чисел. Известно, что числа упорядочены по неубыванию (то есть каждый следующий элемент не меньше предыдущего). Напишите программу, которая определит количество различных чисел в этом массиве.

Входные данные

Сначала задано число N — количество элементов в массиве ($1 \leq N \leq 100$). Далее через пробел записаны N чисел — элементы массива. Массив состоит из целых чисел, находящихся в пределах от -2_{31} до $2_{31}-1$

Выходные данные

Необходимо вывести единственное число - количество различных чисел в массиве.

Примеры

входные данные

```
5
1 1 1 1 1
```

выходные данные

```
1
```

Многомерные массивы

Объявление, ввод и вывод двумерного массива

Объявление `int A[n]` создает в памяти одномерный массив: набор пронумерованных элементов, идущих в памяти последовательно. Но можно создать и массив массивов следующим образом: `int A[n][m]`. Данное объявление создает массив из `n` объектов, каждый из которых в свою очередь является массивом типа `int [m]`. Тогда `A[i]`, где `i` принимает значения от `0` до `n-1` будет в свою очередь одним из `n` созданных обычных массивов, и обратиться к элементу с номером `j` в этом массиве можно через `A[i][j]`.

Подобные объекты (массивы массивов) также называют двумерными массивами. Двумерные массивы можно представлять в виде квадратной таблицы, в которой первый индекс элемента означает номер строки, а второй индекс – номер столбца. Например, массив `A[3][4]` будет состоять из 12 элементов и его можно записать в виде

```
A[0][0]  A[0][1]  A[0][2]  A[0][3]
A[1][0]  A[1][1]  A[1][2]  A[1][3]
A[2][0]  A[2][1]  A[2][2]  A[2][3]
```

Для считывания, вывода на экран и обработки двумерных массивов необходимо использовать вложенные циклы. Первый цикл – по первому индексу (то есть по всем строкам), второй цикл – по второму индексу, то есть по всем элементам в строках. Например, вывести на экран двумерный массив в виде таблицы, разделяя элементы в строке одним пробелом можно следующим образом:

```
int A[n][m];
for(int i=0;i<n;++i)
{  // Выводим на экран строку i
   for(int j=0;j<m;++j)
      cout<<A[i][j]<<" ";
   cout<<endl; // Строка завершается символом перехода на новую строку
}
```

А считать двумерный массив с клавиатуры можно при помощи еще более простого алгоритма (массив вводится по строкам, то есть в порядке, соответствующему первому примеру):

```
for(i=0;i<n;++i)
   for(j=0;j<m;++j)
      cin>>A[i][j];
```

Обработка одномерного массива

Обработка массивов производится аналогичным образом. Например, если мы хотим записать в массив таблицу умножения, то есть присвоить элементу`A[i][j]` значение `i*j`, это можно сделать следующим образом:

```
for(i=0;i<n;++i)
   for(j=0;j<m;++j)
      A[i][j]=i*j;
```

Рассмотрим более сложную задачу и несколько способов ее решения. Пусть дан квадратный двумерный массив `int A[n][n]`. Необходимо элементам, находящимся на главной диагонали проходящей из левого верхнего угла в правый нижний (то есть тем элементам `A[i][j]`, для которых `i==j`) присвоить значение `1`, элементам, находящимся выше главной диагонали – значение 0, элементам, нахощящимся ниже главной диагонали – значение 2. То есть получить такой массив (пример для `n==4`):

```
1 0 0 0
2 1 0 0
2 2 1 0
2 2 2 1
```

Рассмотрим несколько способов решения этой задачи. Элементы, которые лежат выше главной диагонали – это элементы `A[i][j]`, для которых `i<j`, а для элементов ниже главной диагонали `i>j`. Таким образом, мы можем сравнивать значения `i` и `j` и по ним определять значение `A[i][j]`. Получаем следующий алгоритм:

```
for(i=0;i<n;++i)
   for(j=0;j<n;++j)
   {
      if (i<j)
         A[i][j]=0;
      else if(i>j)
         A[i][j]=2;
      else
         A[i][j]=1;
   }
```

Данный алгоритм плох, поскольку выполняет одну или две инструкции `if` для обработки каждого элемента. Если мы усложним алгоритм, то мы сможем обойтись вообще без условных инструкций.

Сначала заполним главную диагональ, для чего нам понадобится один цикл:

```
for(i=0;i<n;++i)
   A[i][i]=1;
```

Затем заполним значением `0` все элементы выше главной диагонали, для чего нам понадобится в каждой из строк с номером `i` присвоить значение элементам `A[i][j]` для `j=i+1`, ..., `n-1`. Здесь нам понадобятся вложенные циклы:

```
for(i=0;i<n;++i)
   for(j=i+1;j<n;++j)
      A[i][j]=0;
```

Аналогично присваиваем значение `2` элементам `A[i][j]` для `j=0`, ..., `i-1`:

```
for(i=0;i<n;++i)
   for(j=0;j<i;++j)
      A[i][j]=2;
```

Можно также внешние циклы объединить в один и получить еще одно, более компактное решение:

```
for(i=0;i<n;++i)
{   // Заполняем строку с номером i
   for(j=0;j<i;++j)
      A[i][j]=2;   // Сначала пишем 2 ниже диагонали
   A[i][j]=1;      // После завершения предыдущего цикла i==j, пишем 1
   for(++j;j<n;++j) // Цикл начинаем с увеличения j на 1
      A[i][j]=0;   // Записываем 0 выше диагонали
}
```

Многомерные массивы

Можно объявлять не только двумерные массивы, но и массивы с большим количеством измерений. Например, объявление `int A[n][m][l]` создает трехмерный массив из `n*m*l` элементов. Для обращения к каждому элементу такого массива необходимо указать три индекса: `A[i][j][k]`, при этом`0<=i`, `i<n`, `0<=j`, `j<m`, `0<=k`, `k<l`. Количество измерений в массиве может быть практически бесконечным (т.е. достаточным для решения любых практических задач).

Форматирование чисел при выводе

Допустим, мы заполним массив таблицей умножения: `A[i][j]=i*j` как в примере в начале раздела. Если мы теперь попробуем вывести этот массив на экран, разделяя элементы в строке одним пробелом, то из-за того, что числа имеют различную длину столбцы таблицы окажутся неровными:

```
0 0 0 0 0 0 0 0 0 0
0 1 2 3 4 5 6 7 8 9
0 2 4 6 8 10 12 14 16 18
0 3 6 9 12 15 18 21 24 27
```

Для того, чтобы получить ровные столбцы, необходимо выводить числа так, чтобы на каждое выводимое число отводилось бы, например, ровно 3 позиции, причем "лишние" позиции были бы заполнены пробелами. Тогда получится следующая таблица:

```
  0  0  0  0  0  0  0  0  0  0
  0  1  2  3  4  5  6  7  8  9
  0  2  4  6  8 10 12 14 16 18
  0  3  6  9 12 15 18 21 24 27
```

Для того, чтобы выводимое число или строка имело в точности заданную ширину, необходимо перед выводом его на экран для потока `cout` вызвать метод `width` с соответствующим знаяением параметра (в нашем случае, `3)`. Данный метод устанавливает ширину поля для выводимого значения. Получим следующую программу для вывода:

```
for(int i=0;i<n;++i)
{
   for(int j=0;j<m;++j)
   {
      cout.width(3);
      cout<<A[i][j];
   }
   cout<<endl;
}
```

Заметим, что мы теперь не выводим пробел после каждого числа, поскольку мы добавили этот пробел к ширине выводимого поля. Функция `width`действует однократно, только на следующее выводимый в поток значение, поэтому ее нужно вызывать перед каждым выводом числа на экран.

Внимание! Если выводимое число или строка имеет большую длину, чем это было установлено функцией `width`, то это число или строка будут выведены полностью, а не будет обрезано до указанного значения. То есть предпочтительней вывести результат некрасиво, нежели неверно.

Упражнения

Если в условиях задачи сказано "Дан двумерный массив", то программа получает на вход два числа `n` и `m`, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет `n` строк по `m` чисел, являющиеся элементами массива. Если в условиях задачи сказано "Дан квадратный массив", то в первой строке входных данных содержится только одно число `n`, далее идет `n` строк по `n` чисел в каждой.

1. Дано число `n`. Создайте массив `int A[n][n]`, и заполните его по следующему правилу:

   Числа на диагонали, идущей из правого верхнего в левый нижний угол равны 1.
   Числа, стоящие выше этой диагонали, равны 0.
   Числа, стоящие ниже этой диагонали, равны 2.

   Полученный массив выведите на экран. Числа разделяйте одним пробелом. Пример

Вход	**Выход**
4	0 0 0 1 0 0 1 2 0 1 2 2 1 2 2 2

2. Дано число `n` и квадратный массив `int A[n][n]`. Проверьте, является ли массив симметричным относительно главной диагонали. Программа должна выводить слово `yes` для симметричного массива и слово `no` для несимметричного. Пример

Вход	**Выход**
3 0 1 2 1 2 3 2 3 4	yes

   Указание. Для элемента `A[i][j]` симметричным ему является элемент `A[j][i]`.

3. **Состязания-1.** В метании молота состязается `n` спортсменов. Каждый из них сделал `m` бросков. Победителем считается тот спортсмен, у которого сумма результатов по всем броскам максимальна.

   Если перенумеровать спортсменов числами от `0` до `n-1`, а попытки каждого из них – от `0` до `m-1`, то на вход программа получает массив `int A[n][m]`, состоящий из неотрицательных чисел. Программа должна определить максимальную сумму чисел в одной строке и вывести на экран эту сумму и номер строки, для которой достигается эта сумма. Если таких строк несколько, то выводится номер наименьшей из них. Пример для`n=4` спортсменов и `m=3` попыток:

Вход	**Выход**
4 3	19

```
5 6 7                    1
6 6 7
7 6 6
4 3 5
```

Не забудьте, что нумерация строк (спортсменов) начинается с 0.

4. **Состязания - 2.** Победителем соревнований объявляется тот спортсмен, у которого максимален наилучший результат по всем броскам. Таким образом, программа должна найти значение максимального элемента в данном массиве, а также его индексы (то есть номер спортсмена и номер попытки). Программа выводит значение максимального элемента, затем номер строки и номер столбца, в котором он встречается. Пример

Вход	**Выход**
4 3 1 4 2 5 2 5 5 1 4 1 2 4	5 1 0

Если в массиве несколько максимальных элементов, то нужно вывести минимальный номер строки, в которой встречается такой элемент, а если в этой строке таких элементов несколько, то нужно вывести минимальный номер столбца. Не забудьте, что все строки и столбцы нумеруются с 0.

5. **Состязания - 3.** Будем считать, что побеждает спортсмен, у которого максимален наилучший бросок. Если таких несколько, то из них побеждает тот, у которого наилучшая сумма результатов по всем попыткам. Если и таких несколько, победителем считается спортсмен с минимальным номером. Определите номер победителя соревнований.

Вход	**Выход**
4 3 8 8 8 5 9 3 9 4 7 6 6 2	2

6. **Состязания - 4.** Будем считать, что победитель определяется по лучшему результату. Определите количество участников состязаний, которые разделили первое место, то есть определите количество строк в массиве, которые содержат значение, равное наибольшему.

Вход	**Выход**
4 3 1 2 3 4 5 6 6 2 5 2 3 4	2

7. **Состязания - 5.** Решите предыдущую задачу, но на экран выведите еще и номера спортсменов, разделивших первое место. Сначала программа выводит количество спортсменов, показавших наилучший результат, затем – их номера в порядке возрастания.

Вход	**Выход**
4 3 1 2 3 4 5 6	2 1 2

```
6 2 5
2 3 4
```

8. Даны два числа `n` и `m`. Создайте двумерный массив `int A[n][m]`, заполните его таблицей умножения `A[i][j]=i*j` и выведите на экран. При этом **нельзя использовать вложенные циклы**, все заполнение массива должно производиться одним циклом, например,`for(i=0;i<n*m;++i)`.
9. Даны два числа `n` и `m`. Создайте двумерный массив `int C[n][m]` и заполните его по следующим правилам:

   Числа, стоящие в строке 0 или в столбце 0 равны 1 (`A[0][j]=1`, `A[i][0]=1`). Для всех остальных элементов массива `A[i][j]=A[i-1][j]+A[i][j-1]`, то есть каждый элемент равен сумме двух элементов, стоящих слева и сверху от него. Выведите данный массив на экран, отводя на вывод каждого числа ровно 6 символов.

   Вход
   ```
   4 6
   ```

   Выход
   ```
   1     1     1     1     1     1
   1     2     3     4     5     6
   1     3     6    10    15    21
   1     4    10    20    35    56
   ```

   Что получилось в результате?

10. Даны числа `n` и `m`. Создайте массив `int A[n][m]` и заполните его следующей змейкой (ниже приведен пример для `n=4` и `m=6`):

    ```
     0   1   2   3   4   5
    11  10   9   8   7   6
    12  13  14  15  16  17
    23  22  21  20  19  18
    ```

    Выведите массив на экран, отводя на вывод каждого числа ровно 3 символа.

11. Даны числа `n` и `m`. Создайте массив `int A[n][m]` и заполните его следующим образом (ниже приведен пример для `n=4` и `m=6`):

    ```
     0   1   3   6  10  14
     2   4   7  11  15  18
     5   8  12  16  19  21
     9  13  17  20  22  23
    ```

    Выведите массив на экран, отводя на вывод каждого числа ровно 3 символа.

12. Дано число `n`. Создайте массив `int A[2*n+1][2*n+1]` и заполните его по спирали начиная с числа `0` в центральной клетке `A[n][n]`. Спираль выходит вверх, далее закручивается против часовой стрелки. Выведите массив на экран, отводя на вывод каждого числа ровно `3` символа. Ниже приведен пример для `n=2`:

    ```
    12 11 10  9 24
    13  2  1  8 23
    14  3  0  7 22
    15  4  5  6 21
    16 17 18 19 20
    ```

Задача №354. Побочная диагональ

Дано число n, $n \leq 100$. Создайте массив $n \times n$ и заполните его по следующему правилу:
- числа на диагонали, идущей из правого верхнего в левый нижний угол, равны 1;
- числа, стоящие выше этой диагонали, равны 0;
- числа, стоящие ниже этой диагонали, равны 2.

Входные данные
Программа получает на вход число n.

Выходные данные
Необходимо вывести полученный массив. Числа разделяйте одним пробелом.

Примеры
входные данные
```
4
```
выходные данные
```
0 0 0 1
0 0 1 2
0 1 2 2
1 2 2 2
```

Задача №355. Симметричная ли матрица?

Проверьте, является ли двумерный массив симметричным относительно главной диагонали. Главная диагональ — та, которая идёт из левого верхнего угла двумерного массива в правый нижний.

Входные данные
Программа получает на вход число $n \leq 100$, являющееся числом строк и столбцов в массиве. Далее во входном потоке идет n строк по nчисел, являющихся элементами массива.
Выходные данные
Программа должна выводить слово `yes` для симметричного массива и слово `no` для несимметричного.

Примеры
входные данные
```
3
0 1 2
1 5 3
2 3 4
```
выходные данные

```
yes
```

входные данные

```
3
0 0 0
0 0 0
1 0 0
```

выходные данные

```
no
```

Задача №356. Состязания

В метании молота состязается `n` спортсменов. Каждый из них сделал `m` бросков. Победителем считается тот спортсмен, у которого сумма результатов по всем броскам максимальна.
Если перенумеровать спортсменов числами от `0` до `n-1`, а попытки каждого из них – от `0` до `m-1`, то на вход программа получает массив`A[n][m]`, состоящий из неотрицательных целых чисел.
Программа должна определить максимальную сумму чисел в одной строке и вывести на экран эту сумму и номер строки, для которой достигается эта сумма.

Входные данные

Программа получает на вход два числа `n` и `m`, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет `n` строк по `m`чисел, являющихся элементами массива.

Выходные данные

Программа должна вывести 2 числа: сумму и номер строки, для которой эта сумма достигается. Если таких строк несколько, то выводится номер наименьшей из них. Не забудьте, что нумерация строк (спортсменов) начинается с 0.

Примеры

входные данные

```
2 2
5 4
3 5
```

выходные данные

```
9
0
```

Задача №357. Состязания – 2

В метании молота состязается `n` спортсменов. Каждый из них сделал `m` бросков. Победителем соревнований объявляется тот спортсмен, у которого максимален наилучший результат по всем

броскам. Таким образом, программа должна найти значение максимального элемента в данном массиве, а также его индексы (то есть номер спортсмена и номер попытки).

Входные данные

Программа получает на вход два числа `n` и `m`, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет `n` строк по `m`чисел, являющихся элементами массива.

Выходные данные

Программа выводит значение максимального элемента, затем номер строки и номер столбца, в котором он встречается. Если в массиве несколько максимальных элементов, то нужно вывести минимальный номер строки, в которой встречается такой элемент, а если в этой строке таких элементов несколько, то нужно вывести минимальный номер столбца. Не забудьте, что все строки и столбцы нумеруются с 0.

Примеры

входные данные

```
3 3
3 1 2
1 3 4
3 3 3
```

выходные данные

```
4
1 2
```

Задача №358. Состязания – 3

В метании молота состязается `n` спортсменов. Каждый из них сделал `m` бросков. Побеждает спортсмен, у которого максимален наилучший бросок. Если таких несколько, то из них побеждает тот, у которого наилучшая сумма результатов по всем попыткам. Если и таких несколько, победителем считается спортсмен с минимальным номером. Определите номер победителя соревнований.

Входные данные

Программа получает на вход два числа `n` и `m`, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет `n` строк по `m`чисел, являющихся элементами массива.

Выходные данные

Программа должна вывести одно число - номер победителя соревнований. Не забудьте, что строки (спортсмены) нумеруются с 0.

Примеры

входные данные

```
3 3
1 2 7
1 3 5
4 1 6
```

выходные данные

```
0
```

Задача №359. Состязания – 4

В метании молота состязается n спортсменов. Каждый из них сделал m бросков. Победитель определяется по лучшему результату. Определите количество участников состязаний, которые разделили первое место, то есть определите количество строк в массиве, которые содержат значение, равное наибольшему.

Входные данные

Программа получает на вход два числа n и m, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет n строк по mчисел, являющихся элементами массива.

Выходные данные

Программа должна вывести одно число - количество победителей соревнования.

Примеры

входные данные

```
3 3
3 1 2
1 3 4
3 3 3
```

выходные данные

```
1
```

Задача №360. Состязания – 5

В метании молота состязается n спортсменов. Каждый из них сделал m бросков. Победитель определяется по лучшему результату. Определите количество участников, а так же самих участников состязаний, которые разделили первое место, то есть определите количество строк в массиве, которые содержат значение, равное наибольшему.

Входные данные

Программа получает на вход два числа n и m, являющиеся числом строк и столбцов в массиве. Далее во входном потоке идет n строк по mчисел, являющихся элементами массива.

Выходные данные

Сначала программа выводит количество спортсменов, показавших наилучший результат, затем – их номера в порядке возрастания. Не забудьте, что строки (спортсмены) нумеруются с 0.

Примеры

входные данные

```
3 3
3 1 2
1 3 4
3 3 3
```

выходные данные

```
1
1
```

Задача №361. Таблица умножения

Даны два числа `n` и `m`. Создайте двумерный массив `A[n][m]`, заполните его таблицей умножения `A[i][j]=i*j` и выведите на экран. При этом**нельзя использовать вложенные циклы**, все заполнение массива должно производиться одним циклом.

Входные данные

Программа получает на вход два числа `n` и `m` - количество строк и столбцов, соответственно.

Выходные данные

Программа должна вывести полученный массив. Числа разделяйте одним пробелом.

Примеры

входные данные

```
3 3
```

выходные данные

```
  0   0   0
  0   1   2
  0   2   4
```

Задача №362. Треугольник Паскаля

Даны два числа `n` и `m`. Создайте двумерный массив `[n][m]` и заполните его по следующим правилам: Числа, стоящие в строке 0 или в столбце 0 равны 1 (`A[0][j]=1`, `A[i][0]=1`). Для всех остальных элементов массива `A[i][j]=A[i-1][j]+A[i][j-1]`, то есть каждый элемент равен сумме двух элементов, стоящих слева и сверху от него.

Входные данные

Программа получает на вход два числа `n` и `m`.

Выходные данные

Выведите данный массив.

Примеры

входные данные

```
3 3
```

выходные данные

```
     1      1      1
     1      2      3
     1      3      6
```

Задача №365. Заполнение спиралью

Дано число `n`. Создайте массив `A[2*n+1][2*n+1]` и заполните его по спирали, начиная с числа `0` в центральной клетке `A[n+1][n+1]`. Спираль выходит вверх, далее закручивается против часовой стрелки.

Входные данные

Программа получает на вход одно число `n`.

Выходные данные

Программа должна вывести полученный массив, отводя на вывод каждого числа ровно 3 символа.

Примеры

входные данные

```
2
```

выходные данные

```
 12 11 10  9 24
 13  2  1  8 23
 14  3  0  7 22
 15  4  5  6 21
 16 17 18 19 20
```

Задача №363. Заполнение змейкой

Даны числа `n` и `m`. Создайте массив `A[n][m]` и заполните его змейкой (см. пример).

Входные данные

Программа получает на вход два числа `n` и `m`.

Выходные данные

Программа должна вывести полученный массив, отводя на вывод каждого числа ровно 3 символа.

Примеры

входные данные

```
4 10
```

выходные данные

```
  0  1  2  3  4  5  6  7  8  9
 19 18 17 16 15 14 13 12 11 10
 20 21 22 23 24 25 26 27 28 29
 39 38 37 36 35 34 33 32 31 30
```

Задача №364. Заполнение диагоналями

Даны числа `n` и `m`. Создайте массив `A[n][m]` и заполните его, как показано на примере.

Входные данные

Программа получает на вход два числа `n` и `m`.

Выходные данные

Программа должна вывести полученный массив.

Примеры

входные данные

```
4 10
```

выходные данные

```
  0  1  3  6 10 14 18 22 26 30
  2  4  7 11 15 19 23 27 31 34
  5  8 12 16 20 24 28 32 35 37
  9 13 17 21 25 29 33 36 38 39
```

Задача №1444. Сапер

Напишите программу, отображающую игровое поле для игры "Сапер".

Входные данные

Даны числа *N* и *M* (целые, положительные, не превышают 32) – количество строк и столбцов в поле соответственно, далее число *W* (целое, неотрицательное, не больше 1000) – количество мин на поле, далее следует *W* пар чисел, координаты мины на поле (первое число – строка, второе число – столбец).

Выходные данные
Требуется вывести на экран поле. Формат вывода указан в примере.

Примеры

входные данные

```
3 2
2
1 1
2 2
```

выходные данные

```
* 2
2 *
1 1
```

входные данные

```
2 2
0
```

выходные данные

```
0 0
0 0
```

Задача №1458. Переворот

Дан массив $N \times M$. Требуется повернуть его по часовой стрелке на 90 градусов.

Входные данные
На первой строке даны натуральные числа N и M ($1 \le N, M \le 50$). На следующих N строках записано по M неотрицательных чисел, не превышающих 10^9 – сам массив.

Выходные данные
Выведите повернутый массив в формате входных данных.

Примеры

входные данные

```
3 4
1  2  3  4
5  6  7  8
9  10 11 12
```

выходные данные

```
4 3
9 5 1
10 6 2
11 7 3
12 8 4
```

Задача №1464. Спираль

Требуется заполнить массив размера $N \times N$ единичками по спирали (начиная с верхнего левого угла по часовой стрелке, см. пример).

Входные данные

С клавиатуры вводится число N (нечетное, натуральное и не превышающее 50).

Выходные данные

Требуется вывести на экран построенную спираль.

Примеры

входные данные

```
7
```

выходные данные

```
1111111
0000001
1111101
1000101
1011101
1000001
1111111
```