**28.02.24. Указатели и двумерные массивы.**

**Массив указателей. Указатели и двумерные массивы. Массив указателей на строки двумерного массива. Примеры программ.**

**Массив указателей**

Как и другие переменные, *указатели могут быть объединены в массивы*.

int \* ptr[10];

Элементы объявленного таким образом массива указателей **ptr[i**] можно *проинициализировать*, например, одним из следующих способов:

* с помощью адреса переменной: int x; ptr[i] = &x;
* с помощью ранее объявленного указателя: int \* qtr = &x; ptr[i] = qtr;
* с помощью адреса массива: int mas[5]; ptr[i] = mas;

**Указатели и двумерные массивы**

***Двумерные массивы*** в языке C++ – это *массивы массивов*, т.е. массивы, *элементами* которых, в свою очередь*,* являются *одномерные массивы*.

При объявлении двумерного массива,например,

int **matr**[4][3]; происходит следующее:

* создаётся массив указателей **matr[4]**;
* каждый из элементов созданногомассива **matr[i]** содержит адрес одномерного массива, представляющего собой **i**-строку массива, состоящего из **3-х** элементов;
* каждый указатель **matr[i]** позволяет обратиться к **i**-ой строке массива.

Доступ к элементам массива указателей**matr[4]**:

matr[i] \*(matr + i)

Соответственно их *адреса*: &matr[i] (matr + i)

Доступ к элементам массива **matr**:

matr[i][j] \*(matr[i] + j) \*(\*(matr +i)+j)

Соответственно адресаэлементов массива:

&matr[i][j] (matr[i]+ j) (\*(matr +i)+j)

**Массив указателей на строки двумерного массива**

При использовании двумерных массивов следует учитывать, что указатели **matr[i]** являются ***константами***, и их значения нельзя изменять во время выполнения программы, т.е. не всегда можно их использовать. Но, например, для передачи *строк* матрицы в функцию элементы массива указателей **matr[i]** использовать можно, что мы и делали в **1-ом семестре**.

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

sum[i] = **summa**(matr[i], m);...

int Summa(int \* ar, int size){ // функция нахождения суммы

int sum = 0; // в одномерном массиве

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

sum = sum + ar[i];

return sum;

}

В программе на **С++** можно создать **массив указателей** **ptr**на строки двумерного массива (матрицы) и использовать его, если потребуется:

int \* ptr[4];

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

ptr[i] = matr[i]; // или ptr[i] = &matr[i][0];

В этом случае д***оступ к элементам*** массива матрицы возможен также и с помощью *указателей*:

ptr[i][j] \*(ptr[i] + j) \*(\*(ptr + i) +j)

Соответственно их адреса:

&ptr[i][j] (ptr[i]+ j) (\*(ptr +i)+j)

Массив указателей на строки матрицы можно использовать для перестановки её строк**,** меняя местами элементы в массиве указателей, т.е, адреса строк, как показано в **Примере 1**.

// **Пример 1**. В данной целочисленной матрице **matr(n, m)** переставить

// местами **0 и 1** строки, используя **массив указателей** на строки массива.

#include <iostream>

using namespace std;

int **main**(){

const int D1 = 20, D2 = 20;

int **matr**[D1][D2], n, m;

int \* ptr[D1]; // массив указателей **ptr**

cout<<"Enter n, m: "; cin>>n>>m;

**srand**(n+m);

for(int i = 0; i < n; i++) // создание **matr**

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

matr[i][j] = rand() % 25 - 10;

cout<<endl<<"**Sourse matrix:**"<<endl;

for(int i = 0;i < n; i++){ // вывод **matr**

for(int j = 0; j < m; j++){

cout.width(5); cout<<matr[i][j];

}

cout<<endl;

}

for(int i = 0; i < n; i++) // инициализация массива **ptr**

ptr[i] = matr[i];

int \* p = ptr [0]; ptr [0] = ptr[1]; ptr [1] = p; // перестановка строк

**/\***x=matr[0]; matr[0]=matr[1]; matr[1]=x;\*/ // **ошибка**!**!!!**

cout<<"**Matrix-rezult**, using array pointer**:**"<<endl;

for(int i = 0;i < n; i++){

for(int j = 0; j < m; j++){

cout.width(5); cout<<\*(ptr[i] + j);

}

cout<<endl;

}

cout<<"**Matrix-rezult**, using name array**:**"<<endl;

for(int i = 0;i < n; i++){

for(int j = 0; j < m; j++){

cout.width(5); cout<<\*(matr[i] + j);

}

cout<<endl;

}

}

**Результат:**

Enter n, m: **3 4**

Sourse matrix:

1 12 5 5

11 6 -8 9

-8 9 8 -1

Matrix-rezult, using array **pointer**: Matrix-rezult, using **name** array:

11 6 -8 9 1 12 5 5

1 12 5 5 11 6 -8 9

-8 9 8 -1 -8 9 8 -1

Массив указателей на строки матрицы также можно использовать для передачи адресов строк в функцию, как показано в **Примере 2.**

// **Пример 2**. Составить функцию, которая определяет есть в одномерном

// массиве отрицательные элементы или их нет.

// Ввести/вывести массив **matr(n, n)**. Найти количество строк в массиве,

// не содержащие отрицательных элементов. Использовать функцию для

// строк массива**.** Создать и использовать **массив указателей** на строки.

#include <iostream>

using namespace std;

int funNeg(int\*, int); // прототип **funNeg()**

int **main()**{

const int D1 = 20, D2 = 20;

int matr[D1][D2], kolStrok = 0, flag;

int \* ptr[D1]; // массив указателей **ptr**

int n, m;

cout << "Enter n, m: "; cin >> n >> m;

cout<<"Enter array-matrix\n";

for (int i = 0; i < n; i++) // ввод маccива

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

cin>>matr[i][j];

cout << "Sourse **array-matrix** : " << endl; // вывод массива

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

cout.width(4); cout << matr[i][j];

}

cout << endl;

}

for(int i = 0; i < n; i++) // инициализация массива **ptr**

ptr[i] = matr[i];

for (int i = 0; i < n; i++) {

flag = funNeg(**ptr[i],** m); // вызов функции **funNeg()**

if (flag == 1) kolStrok++; // аргумент – адрес **i-ой** строки

}

cout << " kolStrok = " << kolStrok << '\n';

}

int **funNeg**(int \* p , int n) { // определение функции **funNeg()**

int pr = 1;

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

if (p[i] < 0) { pr = 0; break; }

return pr;

}

**Результат:**

Enter n, m: **3 4**

**Enter array-matrix:**

1 2 3 4

-1 2 3 4

5 4 3 2

**Sourse array-matrix:**

1 2 3 4

-1 2 3 4

5 4 3 2

kolStrok = 2

**Д. З. на 06 03.2024**

**Постановка задания.**

**!!!** В программах использовать **только** учебный материал .

!!! В тексте программы **обязательно** должны быть **условие задачи** и **тестовый пример**, использованный при отладке программы.

**Ограничения:**

* **!!!** **весь массив в функцию не передавать** **!!!**
* **не** создавать для хранения цифр числа **массив;**
* не задавать числа как **символьный массив**;
* не использовать объекты класса **string**;
* не использовать стандартные функции.

**Требования к программе:**

* в программе создать **массив указателей на строки** заданного двумерного массива **matr(n, m)**;
* использовать созданный массив указателей на строки для перестановки, удаления, вставки строк массива **matr(n, m):**
* в программе обязательно составить и использовать функцию, указанную в условии задачи;
* результат выводить в доступном и понятном виде.

**Задачи**:

1. Составить функцию, которая для одномерного массива целых чисел находит среднее арифметическое положительных элементов. В случае, если в массиве нет положительных элементов, функция возвращает **0.**

В массиве **mas(n, m)**, используя составленную функцию, найти номер строки (**первой**), среднее арифметическое положительных элементов которой является наибольшим, и поменять её местами с первой строкой (**с индексом 0**). Для перестановки строк использовать **массив указателей** на строки. Дополнительный массив не использовать.

**исходный массив** **массив-результат**

**11 12 -22 1 -44** **22 8 -7 6 -25**

**-4 -5 -7 -6 -2**  **-4 -5 -7 -6 -2**

**22 8 -7 6 -25 11 12 -22 1 -44**

20 14 -33 12 2 20 14 -33 12 2

**2 4 -2 -3 9** **2 4 -2 -3 9**

1. \*Составить функцию, которая для одномерного массива натуральных чисел определяет, сколько в нём элементов-палиндромов.

В массиве **matr(n, m)** после строки (**первой**), содержащей наименьшее количество элементов-палиндромов, **вставить нулевую строку** (**все элементы равны 0**). Для перестановки строк использовать **массив указателей** на строки. Дополнительный массив не использовать.

**исходный массив** **массив-результат**

131 34 3223 52 4114 131 34 3223 52 4114

2332 451 15451 672 617 2332 451 15451 672 617

**411 2992 521 72 288**  **411 2992 521 72 288**

42 1331 155 413 513 **0 0 0 0 0 0**

811 5445 166 118 32123 42 1331 155 413 513

811 5445 166 118 32123

1. \*\*Составить функцию, которая для одномерного массива целых чисел определяет **количество пар** соседних элементов, у которых старшие цифры одинаковые.

В массиве **mas(n, m) (m—**чётное), используя функцию, в **каждой строке** определить количество **пар соседних элементов**, у которых старшие цифры одинаковые, и удалить строки, содержащие чётное количество таких пар. Для перестановки строк использовать **массив указателей** на строки. Дополнительный массив не использовать.

**исходный массив** **массив-результат**

413 452 72 421 144 331 413 452 72 421 144 331

**291 618 877 836 25 234** 41 521 666 672 666 511

41 521 666 672 666 511 911 154 144 233 911 123

**861 844 233 322 411 422**

911 154 144 233 911 123

**123 513 543 567 345 321**