Варыянт 3

Заданне 4. Выполнить задания из таблицы ниже, используя ***индексы*** для доступа к элементам массивов в первой программе и ***указатели*** − во второй программе.

|  |
| --- |
| 1. Для заданной целочисленной матрицы **A(n, m)** определить, является ли сумма её элементов чётным числом. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  const int n = 10, m = 15, rmn = 0, rmx = 10;  int Arr[n][m], counter = 0;  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {//для кожнана радка  for (int j = 0; j < m; j++) { //для кожнага элемента ў радку  Arr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn); //запаўняем радок лікамі  cout << Arr[i][j] << " "; //выводзім только што створаны элемент  }  cout << '\n';//канец радка  }  //Лічымі суму ўсіх элементаў  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  counter += Arr[i][j];  }  }  //Правяраем суму на цотнасць і выводзім адпаведны вынік  (counter % 2 == 0) ? cout << "Цотная, роўна " << counter : cout << "Няцотная, роўна " << counter;  } |
|  |
| 2. Задана матрица размером **n**x**m** и число **k**. В переменную **f** поместить значение 0, если все элементы **k**-ой строки упорядочены по убыванию, и значение 1 − в противном случае. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main() {  const int n = 10, m = 10, rmn = 0, rmx = 10;  int a[n][m], k, f = 0;  cout << "Enter the k: "; cin >> k;  int\* ptr = a[k - 1]; //паказальнік на радок масіва пад нумарам k  srand(time(NULL));  //запаўняем масіў выпадковымі лікамі  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  a[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  }  }  //выводзім радок пад нумарам k  for (int i = 0; i < m; i++) {  cout << \*(ptr + i)<<" ";  }  //правяраем ці размешчаны лікі ў парадку памяншэння і калі так, то нічога не робім(f застанеца 0). інкаш мяняем на 1 і выходзім з цыкла  for (int i = 0; i < m - 1; i++) {  if (\*(ptr + i) < \*(ptr + i + 1)) {  f = 1;  break;  }  }  //вывад выніку  cout << "\nf = " << f << endl;  return 0;  } |
|  |

|  |
| --- |
| 1. Дана квадратная матрица порядка **2n**, элементы которой фо рмируются случайным образом и находятся в пределах от −10 до 10. Получить новую матрицу, переставляя ее блоки размера **n×n** в соответствии со схемой. |
| #include <iostream>  #include <cstring>  using namespace std;  int main() {  const int n = 4, rmn = -10, rmx = 10;  int A[2 \* n][2 \* n], N[2 \* n][2 \* n];  srand(time(NULL));  //запаўняем масіў выпадковымі лікамі  for (int i = 0; i < 2 \* n; i++) {  for (int j = 0; j < 2 \* n; j++) {  A[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  (A[i][j] < 0) ? cout << A[i][j] << " " : cout << " " << A[i][j] << " "; // выводзім элементы. калі неадмоўны то дабаўляем перад ім прабел как былі ўсе лікі адзін пад аднім  }  cout << '\n';  }  //мяняем месцами блоки памерамі n на n якія стаяць злева ўверсе і справа знізу  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  N[i + n][j + n] = A[i][j];  N[i][j] = A[i + n][j + n];  }  }  //мяняем месцами блоки памерамі n на n якія стаяць злева знізу і справа ўверсе  for (int i = 0; i < n; i++) {  for (int j = n; j < 2\*n; j++) {  N[i+n][j-n] = A[i][j];  N[i][j] = A[i + n][j - n];  }  }  cout << "\n\n\n\n";  // выводзім вынік  for (int i = 0; i < 2\* n; i++) {  for (int j = 0; j < 2\* n; j++) {  (N[i][j] < 0) ? cout << N[i][j] << " " : cout << " " << N[i][j] << " ";  }  cout << '\n';  }  } |
|  |
| 2. Латинским квадратом порядка **n** называется квадратная таблица размером **nхn**, каждая строка и каждый столбец которой содержат все числа от 1 до **n**. Для заданного **n** в матрице **L(n, n**) построить латинский квадрат порядка **n**. |
| #include <iostream>  #include <cstring>  using namespace std;  int main() {  const int n = 10;  int N[n][n], f = 1;  //запаўняем масіў лікамі  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  f = (f > n) ? 1 : f; //калі f = n+1, то не меняем на 1  N[i][j] = f++; //прызначаем на f і павялічваем яго на 1  }  f = N[i][0] + 1; //для кожнага новага радка бяром за пачатаковае значэнне для f першы элемент папярэдняга радка  f = (f > n) ? 1: f;  }  //выводзім вынік  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  (N[i][j] < 0) ? cout << N[i][j] << " " : cout << " " << N[i][j] << " ";  }  cout << '\n';  }  } |
|  |

|  |
| --- |
| 3. Путем перестановки элементов квадратной вещественной матрицы добиться того, чтобы ее максимальный элемент находился в левом верхнем углу, следующий по величине − в позиции (2, 2), следующий − в позиции (3, 3) и т. д., заполнив таким образом всю главную диагональ. |
| #include <iostream>  #include <cstring>  using namespace std;  void bubbleSort(int\* arr, int size) { //Функцыя сартыроўкі шляхам перастановак. калі, напрыклад, гэты элемент 5, наступны 55, то тады мяняем гэтыя элементы ў масіве месцамі. і так да таго пакуль увесь масіў не будзе адсартаваны  for (int i = 0; i < size - 1; i++) {  for (int j = 0; j < size - i - 1; j++) {  if (\*(arr + j) < \*(arr + j + 1)) {  // Мяняем пазіцыямі элементы калі гэты лік менш наступнага  int temp = \*(arr + j);  \*(arr + j) = \*(arr + j + 1);  \*(arr + j + 1) = temp;  }  }  }  }  int main() {  const int n = 10, rmn = 0, rmx = 90;  int A[n][n], N[n \* n], counter = 0;  int\* ptr = N;  srand(time(NULL));  //запаўняем масіў выпадковымі лікамі  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  A[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  (A[i][j] / 10 != 0) ? cout << A[i][j] << " " : cout << " " << A[i][j] << " "; // выводзім элементы. калі не двухзначны то дабаўляем перад ім прабел как былі ўсе лікі адзін пад аднім  }  cout << '\n';  }  //запаўняем масіў N лікамі масіва А  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  N[counter++] = A[i][j];  }  }  //Сартыруем масіў з выкарыстаннем нашай функцыі  bubbleSort(ptr, n \* n);  //першыя n лікаў адсартаванага масіва N прызначаем на галоўную дыяганаль масіву А  for (int i = 0; i < n; i++) {  A[i][i] = N[i];  }  //выводзім вынік  cout << "\n\n\n\n";  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  (A[i][j] / 10 != 0) ? cout << A[i][j] << " " : cout << " " << A[i][j] << " ";  }  cout << '\n';  }  } |
|  |