Задание 1

|  |
| --- |
| #include <iostream>  void main()  {  const int ci = 3, cj = 2;  int a[ci][cj] = { { 1, 2 }, { 3, 4 }, { 5, 6 } };  for (int i = 0; i < ci; i++)  for (int j = 0; j < cj; j++)  std::cout << "\n a[" << i << "," << j << "] =" << a[i][j];  }    Изменённый вариант :  #include <iostream>  void main()  {  const int ci = 3, cj = 3;  int a[ci][cj] = { { 1,2,3 }, { 3,4,5 }, { 5,6,7 } };  for (int i = 0; i < ci; i++) {  for (int j = 0; j < cj; j++) {  std::cout << a[i][j] << " ";  }  std::cout << "\n";  }    } |

Задание 2

Условие: Программа должна вычислить и вывести сумму всех элементов двумерного массива.

|  |  |
| --- | --- |
| #include <iostream>  void main()  {  const int n = 3, m = 2;int h = 0;  int A[n][m] = { 0, 1, 2, 3, 4, 5 };  for (int i = 0; i < n; i++)  for (int j = 0; j < m; j++)  h += A[i][j];  std::cout << h;  } | #include <iostream>  void main()  {  const int n = 3, m = 2;int h = 0;  int C[n][m] = { 0, 1, 2, 3, 4, 5 };  for (int i = 0; i < n; i++)  for (int j = 0; j < m; j++)  h += \*(\*(C + i) + j);  std::cout << h;  } |



Задание 3

Условие: Программа должна найти минимальный элемент массива и вывести его и его координаты

|  |
| --- |
| #include <iostream>  void main()  {  setlocale(LC\_CTYPE, "Russian");  using namespace std;  const int n = 2, m = 4;  int B[n][m];  int i, j, row = 0, column = 0;  cout << "Введите элементы массива" << endl;  for (i = 0; i < n; i++)  for (j = 0; j < m; j++)  cin >> B[i][j];  int min = B[0][0];  for (int i = 0; i < n; i++)  for (int j = 0; j < m; j++)  if (min > B[i][j])  {  min = B[i][j];  column = i;  row = j;  }  cout << " Исходный массив:" << endl;  for (int i = 0; i < n; i++)  {  cout << "\n";  for (int j = 0; j < m; j++)  cout << "B[" << i << "," << j << "] =" << B[i][j] << "\t";  }  cout << endl;  cout << "Минимальный элемент B[" << column << "," << row << "] =" << min << endl;  } |

Изменённый вариант

|  |
| --- |
| #include <iostream>  using namespace std;  void main()  {  setlocale(LC\_CTYPE, "Russian");  const int n = 2, m = 4;  int B[n][m];  int i, j;  cout << "Введите элементы массива" << endl;  for (i = 0; i < n; i++)  for (j = 0; j < m; j++)  cin >> \*(\*(B+i)+j);  cout << "Исходный массив:" << endl;  for (i = 0; i < n; i++)  {  for (j = 0; j < m; j++)  cout << "B[" << i << "," << j << "] = " << \*(\*(B + i) + j) << "\t";  cout << endl;  }  cout << "Минимальные элементы по строкам:" << endl;  for (i = 0; i < n; i++)  {  int min = \*(\*(B + i) + 0);  int column = 0;  for (j = 1; j < m; j++) // начинаем с 1, так как 0й уже учтён  {  if (min > \*(\*(B + i) + j))  {  min = \*(\*(B + i) + j);  column = j;  }  }  cout << "Минимальный элемент в строке " << i << ": B[" << i << "," << column << "] = " << min << endl;  }  } |

Вариант 12

|  |
| --- |
| 1. Дана целочисленная матрица **A(n, m)**. Вычислить сумму и произведение нечётных отрицательных элементов матрицы, удовлетворяющих условию **| ai j| < i**. (индексы) |

|  |
| --- |
| #include <iostream>  #include <cmath>  using namespace std;  void main()  {  setlocale(LC\_CTYPE, "Russian");  const int n = 5, m = 2;  int sum = 0, prz = 1;  bool has\_elements = false;  int B[n][m];  int i, j;  cout << "Введите элементы массива:" << endl;  for (i = 0; i < n; i++)  for (j = 0; j < m; j++)  cin >> B[i][j];  cout << "Исходный массив:" << endl;  for (i = 0; i < n; i++)  {  for (j = 0; j < m; j++)  cout << "B[" << i << "," << j << "] = " << B[i][j] << "\t";  cout << endl;  }  for (i = 0; i < n; i++)  {  for (j = 0; j < m; j++)  {  if (B[i][j] % 2 != 0 && B[i][j] < 0 && abs(B[i][j]) < i)  {  sum += B[i][j];  prz \*= B[i][j];  has\_elements = true;  }  }  }  if (has\_elements)  {  cout << "Сумма нечётных отрицательных элементов = " << sum << endl;  cout << "Произведение нечётных отрицательных элементов = " << prz << endl;  }  else  {  cout << "Нет подходящих элементов в массиве" << endl;  }  } |

Задание 2

|  |
| --- |
| 2. Проверить, все ли строки матрицы содержат хотя бы один положительный элемент. Если да, то изменить знаки всех элементов матрицы на обратные.(указатели) |

|  |
| --- |
| #include <iostream>  #include <cmath>  using namespace std;  void main() {  setlocale(LC\_CTYPE, "Russian");  const int n = 5, m = 2;  bool allRowsP = true; // проверка наличия положительных элемента в каждой строке  int B[n][m];  int i, j;  cout << "Введите элементы массива:" << endl;  for (i = 0; i < n; i++)  for (j = 0; j < m; j++)  cin >> \*(\*(B + i) + j);  cout << "Исходный массив:" << endl;  for (i = 0; i < n; i++) {  for (j = 0; j < m; j++)  cout << "B[" << i << "," << j << "] = " << \*(\*(B + i) + j) << "\t";  cout << endl;  }  for (i = 0; i < n; i++) {  bool rowHasPositive = false; //строка, содержащая положительный элемент  for (j = 0; j < m; j++) {  if (\*(\*(B + i) + j) > 0) {  rowHasPositive = true;  break;  }  }  if (!rowHasPositive) {  allRowsP = false;  break; // если хотя бы одна строка не содержит положительного элемента, прекращаем проверку  }  }  if (allRowsP) {  // если все строки содержат хотя бы один положительный элемент, изменяем знаки всех элементов  for (i = 0; i < n; i++) {  for (j = 0; j < m; j++) {  \*(\*(B + i) + j) = -(\*(\*(B + i) + j)); // измененяем знак на прротивоположный  }  }  cout << "Изменённый массив:" << endl;  for (i = 0; i < n; i++) {  for (j = 0; j < m; j++)  cout << "B[" << i << "," << j << "] = " << \*(\*(B + i) + j) << "\t";  cout << endl;  }  }  else {  cout << "Не все строки содержат хотя бы один положительный элемент." << endl;  }  } |

Доп.задания:

|  |
| --- |
| Дана квадратная матрица порядка **2n**, элементы которой формируются 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 <cstdlib>  #include <ctime>  using namespace std;  int main() {  setlocale(LC\_CTYPE, "Russian");  srand(time(0));  int n;  cout << "Введите значение n (матрица будет размером 2n x 2n): ";  cin >> n;  int size = 2 \* n;  int matrix[100][100];  cout << "Исходная матрица:" << endl;  for (int i = 0; i < size; i++) {  for (int j = 0; j < size; j++) {  matrix[i][j] = rand() % 21 - 10; // генерация случайных чисел от -10 до 10  cout << matrix[i][j] << "\t";  }  cout << endl;  }  for (int i = 0; i < n; ++i) {  for (int j = 0; j < n; ++j) {  // обмен элементов между блоками  int temp = matrix[i][j];  matrix[i][j] = matrix[i + n][j + n];  matrix[i + n][j + n] = temp;  temp = matrix[i][j + n];  matrix[i][j + n] = matrix[i + n][j];  matrix[i + n][j] = temp;  }  }  cout << "\nИзменённая матрица:" << endl;  for (int i = 0; i < size; i++) {  for (int j = 0; j < size; j++) {  cout << matrix[i][j] << "\t";  }  cout << endl;  }  return 0;  } |

|  |
| --- |
| 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>  using namespace std;  int main() {  setlocale(LC\_CTYPE, "Russian");  const int n = 3;  int L[n][n];  //заполняем латинский квадрат  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  L[i][j] = (i + j) % n + 1;  }  }  cout << "Латинский квадрат порядка " << n << ":" << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  cout << L[i][j] << "\t";  }  cout << endl;  }  return 0;  } |

|  |
| --- |
| 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 <cstdlib>  #include <ctime>  using namespace std;  void men(float& a, float& b) {  float temp = a;  a = b;  b = temp;  }  int main() {  setlocale(LC\_CTYPE, "Russian");  srand(time(0));  const int n = 4;  float matrix[n][n];  // Генерация случайной матрицы  cout << "Исходная матрица:" << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  matrix[i][j] = rand() % 21 - 10;  cout << matrix[i][j] << "\t";  }  cout << endl;  }  // Сортировка элементов на главной диагонали по убыванию  for (int k = 0; k < n; k++) {  int max\_i = k, max\_j = k;  float max\_val = matrix[k][k];  // Поиск максимального элемента в оставшейся подматрице  for (int i = k; i < n; i++) {  for (int j = k; j < n; j++) {  if (matrix[i][j] > max\_val) {  max\_val = matrix[i][j];  max\_i = i;  max\_j = j;  }  }  }  // Обмен максимального элемента с элементом на диагонали  men(matrix[k][k], matrix[max\_i][max\_j]);  }  // Вывод измененной матрицы  cout << "\nИзменённая матрица:" << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++) {  cout << matrix[i][j] << "\t";  }  cout << endl;  }  return 0;  } |