5. В соответствии со своим вариантом написать главную функцию, в которой имеются вызовы ***функций пользователя***, реализующих задачи из таблицы ниже. Ввод исходных данных и вывод результатов осуществить в главной функции, при этом использовать***динамические* *массивы***.

Для передачи параметров в функции пользователя и возвращения результатов применить ***указатели*** и ***ссылки***.

Варыянт 3

|  |
| --- |
| 1. В массиве из целых чисел определить сумму элементов, расположенных между первым и последним нулевыми элементами |
| #include <iostream>  #include <Windows.h>  using namespace std;  int& sumBetweenZeros(int\* arr, int size, int& sum) { //функцыя, якая вяртае спасылку на суму  int firstZero = -1;  int lastZero = -1;  for (int i = 0; i < size; ++i) {  if (arr[i] == 0) {  if (firstZero == -1) firstZero = i; //для firstZero захоўваем пазіцыю толькі першага нуля  lastZero = i; //захоўваем пазіцыю апошняга нуля  }  }  if (firstZero != -1 && lastZero != -1 && firstZero != lastZero) { //калі было знойдзена 2 нуля ці больш  for (int i = firstZero + 1; i < lastZero; ++i) {  sum += arr[i]; //знаходзім суму элементаў паміж нулямі  }  }  //Вяртаем спасылку на суму  return sum;  }  int main() {  SetConsoleOutputCP(1251);  int size, rmn = 0, rmx = 10;  cout << "Увядзіце памер масіва: ";  cin >> size;  int\* ptr = new int[size];  //запаўняем масіў выпадковымі лікамі і адразу выводзім  for (int i = 0; i < size; ++i) {  ptr[i] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i] << ' ';  }  cout << '\n';  int sum = 0; //зменная якая будзе захоўваць суму лікаў  sum = sumBetweenZeros(ptr, size, sum); //выклікаем функцыю  cout << "Сума элементаў паміж першым і апошнім нулявымі элементамі: " << sum;  //выдаляем масіў  delete[] ptr;  return 0;  } |
|  |
| 2. Если есть в матрице строка, все элементы которой положительны, то найти сумму этих элементов. Уменьшить все элементы матрицы на эту сумму. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int& sumBetweenZeros(int\*\* arr, int n, int m, int& sum) { //функцыя, якая вяртае спасылку на суму  bool flag = false;  for (int i = 0; i < n; i++) {z  for (int j = 0; j < m; j++) {  if (arr[i][j] <= 0) { //калі лік адмоўны або 0, то выходзім з цыкла і ставім сцяг на праўда, што значыць што быў знойдзены не дадатны лік  flag = true;  break;  }  }  if (!flag) { //калі ўсе лікі радка былі дадатнымі  for (int j = 0; j < m; j++) {  sum += arr[i][j]; //знаходзім суму радка  }  break;  }  flag = false;  }  //памяньшаем усе элементы на суму  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  arr[i][j] = arr[i][j] - sum;  }  }  return sum;  }  int main() {  SetConsoleOutputCP(1251);  int n, m, rmn = -10, rmx = 10;  //Ствараем масіў n на m з дапамогай new  cout << "Увядзіце праз прабел колькасць радкоў і колькасць сталбцоў: "; cin >> n >> m;  int\*\* ptr = new int\* [n];  for (int i = 0; i < n; ++i)  ptr[i] = new int[m];  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  ptr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i][j] << " ";  }  cout << '\n';  }  int sum = 0; //зменная якая будзе захоўваць суму лікаў  sum = sumBetweenZeros(ptr, n, m, sum); //выклікаем функцыю  cout << "Сума элементаў дадатнага радка: " << sum<<endl;    //выводзім масіў пасля выканання функцыі  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  cout << ptr[i][j] << ' ';  }  cout << '\n';  }  //выдаляем масіў  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  return 0;  } |
|  |

Варыянт 4

|  |
| --- |
| 1. Создать одномерный массив, содержащий 15 элементов, наполнить его случайными значениями в интервале от 1 до 200. Определить сумму всех нечетных элементов массива. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int& sumBetweenZeros(int\* arr, int size, int& sum) { //функцыя, якая вяртае спасылку на суму  sum = 0;  for (int i = 0; i < size; ++i) {  if ((i + 1) % 2 != 0) { //кали няцотны нумар элемента (индэкс + 1)  sum+=arr[i];  }  }  //Вяртаем спасылку на суму  return sum;  }  int main() {  SetConsoleOutputCP(1251);  int size = 15, rmn = 1, rmx = 200;  int\* ptr = new int[size];  //запаўняем масіў выпадковымі лікамі і адразу выводзім  for (int i = 0; i < size; ++i) {  ptr[i] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i] << ' ';  }  cout << '\n';  int sum = 0; //зменная якая будзе захоўваць суму лікаў  sum = sumBetweenZeros(ptr, size, sum); //выклікаем функцыю  cout << "Сума: " << sum;  //выдаляем масіў  delete[] ptr;  return 0;  } |
|  |
| 2. Проверить, есть ли в матрице хотя бы одна строка, содержащая положительный элемент и найти ее номер. Знаки элементов предыдущей строки изменить на противоположные. |
| #include <iostream>  #include <Windows.h>  using namespace std;  bool findPositiveRowAndInvertPrevious(int\*\* matrix, int rows, int cols, int& rowIndex) {  bool found = false;  for (int i = 0; i < rows; ++i) {  for (int j = 0; j < cols; ++j) {  if (matrix[i][j] > 0) { //Калі лік дадатны  found = true; //ставім сцяг, што быў знойдзены дадатны лік  rowIndex = i; //захоўваем індэкс радка  break; //выходзім з цыкла для элементаў радка  }  }  if (found) { //калі быў знойдзены дадатны лік  if (rowIndex > 0) { // мяняем папярэдні радок, калі гэта не першы радок. інакш не мяняем  for (int j = 0; j < cols; ++j) {  matrix[rowIndex - 1][j] \*= -1;  }  }  break; // выходзім з цыкла  }  }  //вяртаем сцяг, які значыць ці быў знойдзены дадатны элемент  return found;  }  int main() {  SetConsoleOutputCP(1251);  int n, m, rmn = -10, rmx = 10;  //Ствараем масіў n на m з дапамогай new  cout << "Увядзіце праз прабел колькасць радкоў і колькасць сталбцоў: "; cin >> n >> m;  int\*\* ptr = new int\* [n];  for (int i = 0; i < n; ++i)  ptr[i] = new int[m];  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  ptr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i][j] << " ";  }  cout << '\n';  }  int rowIndex = 0; //зменная якая будзе захоўваць індэкс радка, які ўтрымлівае дадатны лік  bool found;  //выклікаем функцыю і яе вынік захоўваем у зменную  found = findPositiveRowAndInvertPrevious(ptr, n, m, rowIndex);  if (found) { //калі быў знойдзены дадатны элемент  cout << "Быў знойдзены дадатны элемент у радку пад нумарам: " << rowIndex + 1 << endl;  cout << "Новы масіў: " << endl;  for (int i = 0; i < n; i++) { //паколькі трэба было змяніць папярэдні радок, то выводзім матрыцу пасля змен  for (int j = 0; j < m; j++) {  cout << ptr[i][j] << ' ';  }  cout << '\n';  }  }  else {  cout << "Усе лікі не дадатныя";  }  //выдаляем масіў  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  return 0;  } |
|  |

Варыянт 6

|  |
| --- |
| 1. Найти сумму минимального и максимального элементов одномерного массива. |
| #include <iostream>  #include <Windows.h>  #include <ctime>  using namespace std;  int sumOfMinAndMax(int\* arr, int size, int& suma) {  //ставім за пачатковыя значэнні першы элемент масіва  int minElement = arr[0];  int maxElement = arr[0];  for (int i = 1; i < size; ++i) {  minElement = min(minElement, arr[i]); //знаходзім мінімальны элемент з дапамогай функцыі min, якая прымае два лікі і вяртае меншы з іх  maxElement = max(maxElement, arr[i]); //знаходзім найбольшы элемент з дапамогай функцыі max, якая прымае два лікі і вяртае большы з іх  }  suma = minElement + maxElement; //знаходзім суму  return suma;  }  int main() {  SetConsoleOutputCP(1251);  int n, m, rmn = -10, rmx = 10;  //Ствараем масіў n на m з дапамогай new  cout << "Увядзіце колькасць элементаў: "; cin >> n;  int\* ptr = new int[n];  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  ptr[i] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i] << " ";  }  int suma = 0; //зменная якая будзе захоўваць суму  //выклікаем функцыю  sumOfMinAndMax(ptr, n, suma);  cout<<'\n' << suma;  //выдаляем масіў  delete[] ptr;  return 0;  } |
|  |
| 2. Проверить, все ли строки матрицы содержат хотя бы один нулевой элемент. Если нет, то заменить значения всех отрицательных элементов матрицы на нулевые. |
| #include <iostream>  #include <Windows.h>  using namespace std;  bool& findPositiveRowAndInvertPrevious(int\*\* matrix, int rows, int cols, bool& suits) {  bool rowContainesZero, allRowsContainZero = suits;  for (int i = 0; i < rows; i++) {  rowContainesZero = false;  for (int j = 0; j < cols; j++) {  if (matrix[i][j] == 0) { //калі знайшлі 0, то ставім сцяг на true і выходзім з цыкла  rowContainesZero = true;  break;  }  }  if (!rowContainesZero) { //калі не быў знойдзены 0 у гэтым радку  allRowsContainZero = false; //то ставім сцяг на false і выходзім з цыкла  break;  }  }  if (!allRowsContainZero) { //калі не ў кожным радку ёсць 0, то мяняем адмоўныя лікі на 0  for (int i = 0; i < rows; i++) {  for (int j = 0; j < cols; j++) {  if (matrix[i][j] < 0) {  matrix[i][j] = 0;  }  }  }  }  return allRowsContainZero;  }  int main() {  SetConsoleOutputCP(1251);  int n, m, rmn = -5, rmx = 5;  //Ствараем масіў n на m з дапамогай new  cout << "Увядзіце праз прабел колькасць радкоў і колькасць сталбцоў: "; cin >> n >> m;  int\*\* ptr = new int\* [n];  for (int i = 0; i < n; ++i)  ptr[i] = new int[m];  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  ptr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i][j] << " ";  }  cout << '\n';  }  bool allRowsContainZero = true; //зменная якая будзе захоўваць індэкс радка, які ўтрымлівае дадатны лік  bool found;  //выклікаем функцыю і яе вынік захоўваем у зменную  found = findPositiveRowAndInvertPrevious(ptr, n, m, allRowsContainZero);  if (!found) { //калі не ўсе радкі маюць 0  cout << "Не ўсе радкі ўтрымліваюць 0" << endl;  cout << "Новы масіў: " << endl;  for (int i = 0; i < n; i++) { //паколькі трэба было змяніць, то выводзім матрыцу пасля змен  for (int j = 0; j < m; j++) {  cout << ptr[i][j] << ' ';  }  cout << '\n';  }  }  else {  cout << "Усе радкі ўтрымліваюць 0";  }  //выдаляем масіў  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  return 0;  } |
|  |

Варыянт 7

|  |
| --- |
| 1. Создать одномерный массив, содержащий 15 элементов, наполнить его случайными значениями в интервале от 1 до 200. Определить произведение элементов массива с индексами от 2 до 7 |
| #include <iostream>  #include <Windows.h>  using namespace std;  unsigned long long& sumBetweenZeros(int\* arr, int size, unsigned long long& product) { //функцыя, якая вяртае спасылку на здабытак элементаў з індэксамі ад 2 да 7  product = 1;  for (int i = 2; i <= 7; i++) {  product \*= arr[i];  }  //Вяртаем спасылку на здабытак  return product;  }  int main() {  SetConsoleOutputCP(1251);  int size = 15, rmn = 1, rmx = 200;  int\* ptr = new int[size];  //запаўняем масіў выпадковымі лікамі і адразу выводзім  srand((unsigned)time(NULL));  for (int i = 0; i < size; ++i) {  ptr[i] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i] << ' ';  }  cout << '\n';  unsigned long long product = 1; //зменная якая будзе захоўваць здабытак лікаў. тып дадзеных бо здабытак лікаў можа быть ВЕЛЬМІ вялікім. для 200^6 і unsigned long long выконваецца 64000000000000 < 18446744073709551615  product = sumBetweenZeros(ptr, size, product); //выклікаем функцыю  cout << "Здабытак: " << product;  //выдаляем масіў  delete[] ptr;  return 0;  } |
|  |
| 2. Если есть в матрице столбец, все элементы которого отрицательны, то найти среднее арифметическое этих элементов. Вычесть полученное значение из всех элементов матрицы. |
| #include <iostream>  #include <Windows.h>  using namespace std;  double& sumBetweenZeros(double\*\* arr, int n, int m, double& sum) { //функцыя, якая вяртае спасылку на сярэдняе арыфметычнае  bool flag = false;  sum = 0;  for (int i = 0; i < m; i++) {  for (int j = 0; j < n; j++) {  if (arr[j][i] >= 0) { //калі лік не адмоўны , то выходзім з цыкла і ставім сцяг на праўда, што значыць што быў знойдзены не адмоўны лік  flag = true;  break;  }  }  if (!flag) { //калі ўсе лікі радка былі адмоўнымі  for (int j = 0; j < n; j++) {  sum += arr[j][i]; //знаходзім суму стаўбца  }  break;  }  flag = false;  }  sum = sum / n; //сярэдняе арыфметычнае  //памяньшаем усе элементы на сярэдняе арыфметычнае  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  arr[i][j] = arr[i][j] - sum;  }  }  return sum;  }  int main() {  SetConsoleOutputCP(1251);  int n, m, rmn = -10, rmx = 10;  //Ствараем масіў n на m з дапамогай new  cout << "Увядзіце праз прабел колькасць радкоў і колькасць сталбцоў: "; cin >> n >> m;  double\*\* ptr = new double\* [n];  for (int i = 0; i < n; ++i)  ptr[i] = new double[m];  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  ptr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i][j] << " ";  }  cout << '\n';  }  double sum = 0; //зменная якая будзе захоўваць сярэдняе арыфметычнае радка  sum = sumBetweenZeros(ptr, n, m, sum); //выклікаем функцыю  if (sum == 0) { //Калі sum = 0, значыць не было радка з усімі адмоўнымі, бо сума лікаў аднаго знака не можа быць 0  cout << "Няма стаўбца, дзе ўсе лікі адмоўныя" << endl;  }  else { //інкаш выводзім зменены масіў і сярэдняе арыфметычнае  cout << "сярэдняе арыфметычнае: " << sum << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < m; j++) {  cout << ptr[i][j] << ' ';  }  cout << '\n';  }  }  //выдаляем масіў  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  return 0;  } |
|  |