Заданне 4. В соответствии со своим вариантом написать программы с использованием ***динамических*** массивов для условий задач из таблицы. Начальные размерности массивов ввести с клавиатуры. Первое задание выполнить с использованием ***функций*** для выделения динамической памяти на языке С, второе − с использованием операторов **new** и **delete**  на языке **С++**.

Варыянт 3

|  |
| --- |
| 1. В одномерном массиве, состоящем из п вещественных элементов, вычислить произведение отрицательных элементов массива и сумму положительных элементов массива, расположенных до максимального элемента. |
| #include <iostream>  #include <Windows.h>  #include <ctime>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, indexOfMaxValue = 0;  double rmn = -20, rmx = 20, productOfNegativeNumbers = 1.0, maxValue, sumOfPositiveBeforeMaxValue = 0.0;  cout << "Увядзіце памер масіву: "; cin >> n;  double\* ptr = (double\*)malloc(n \* sizeof(double));  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  \*(ptr + i) = ((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn;  cout << \*(ptr + i) << " ";  }  maxValue = \*(ptr + indexOfMaxValue); //За максімальны лік бяру першы лік масіву  for (int i = 0; i < n; i++) {  if (\*(ptr + i) > maxValue) { //Калі лік боль за максімальны, захаваны раней, то захоўваем яго як максімальны і яго індэкс  maxValue = \*(ptr + i);  indexOfMaxValue = i;  }  //Знаходзім здабытак адмоўных  if (\*(ptr + i) < 0) {  productOfNegativeNumbers \*= \*(ptr + i);  }  }  //ад пачатку да індэксу максімальнага значэння  for (int i = 0; i < indexOfMaxValue; i++) {  if (\*(ptr + i) > 0) { //Знаходзім суму дадатных лікаў  sumOfPositiveBeforeMaxValue += \*(ptr + i);  }  }  cout << "\nЗдабытак адмоўных лікаў: " << productOfNegativeNumbers;  cout << "\nСума дадатных да максімальнага: " << sumOfPositiveBeforeMaxValue;  } |
|  |
| 2. Дана матрица размером 4x4. Найти сумму наименьших элементов ее нечетных строк и наибольших элементов ее четных строк. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n = 4, rmn = 0, rmx = 20, minValue, maxValue, sum = 0;  //Ствараем масіў 4 на 4 з дапамогай new  int\*\* ptr = new int\* [4];  for (int i = 0; i < 4; ++i)  ptr[i] = new int[4];  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 < n; j++) {  ptr[i][j] = (int)(((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn);  cout << ptr[i][j] << " ";  }  cout << '\n';  }  for (int i = 0; i < n; i++) {  minValue = ptr[i][0], maxValue = ptr[i][0]; // Прызначаем пачатковыя значэнні для кожнага радка за першае  for (int j = 0; j < n; j++) {  if ((i + 1) % 2 != 0) { // Калі гэта няцотны радок, то мяняем значэнне максімальнага значэння для гэтага радка на 0 і знаходзім найменшае  maxValue = 0;  minValue = (ptr[i][j] < minValue) ? ptr[i][j] : minValue;  }  else { //Калі радок цотны, то мяняем мінімальнае значэнне на 0 і знаходзім максімальнае  minValue = 0;  maxValue = (ptr[i][j] > maxValue) ? ptr[i][j] : maxValue;  }  }  //Дабаўляем атрыманыя лікі да сумы. для гэтага і усталеўвалісь значэнні на 0, каб яны не уплывалі на суму і можна было аб'яднать знаходжанне для цотнага і няцотнага радкоў у адным цыкле  sum += minValue;  sum += maxValue;  }  //Выдяляем масіў з памяці  for (int i = 0; i < 4; ++i)  delete[] ptr[i];  delete[] ptr;  //Выводзім вынік  cout <<'\n' << sum;  } |
|  |

Варыянт 1

|  |
| --- |
| 1. В одномерном массиве, состоящем из n вещественных элементов, вычислить количество отрицательных элементов массива и сумму модулей элементов, расположенных после минимального по модулю элемента. |
| #include <iostream>  #include <Windows.h>  #include <ctime>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, indexOfMinValue = 0, counterOfNegativeNumbers = 0;  double rmn = -20, rmx = 20, minValue, sumAfterMinValue = 0.0;  cout << "Увядзіце памер масіву: "; cin >> n;  double\* ptr = (double\*)malloc(n \* sizeof(double));  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  \*(ptr + i) = ((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn;  cout << \*(ptr + i) << " ";  }  minValue = abs(\*(ptr + indexOfMinValue)); //За мінімальны лік бяру першы лік масіву  for (int i = 0; i < n; i++) {  if (abs(\*(ptr + i)) < minValue) { //Калі лік менш за мінімальны, захаваны раней, то захоўваем яго як мінімальны і яго індэкс  minValue = \*(ptr + i);  indexOfMinValue = i;  }  //Знаходзім колькасць адмоўных  if (\*(ptr + i) < 0) {  counterOfNegativeNumbers++;  }  }  //Ад індэксу мінімальнага значэння да канца  for (int i = indexOfMinValue + 1; i < n; i++) {  //Знаходзім суму  sumAfterMinValue += abs(\*(ptr + i));  }  cout << "\nКолькасць адмоўных лікаў: " << counterOfNegativeNumbers;  cout << "\nСума модулеў ад мінімальнага па модулю: " << sumAfterMinValue;  } |
|  |
| 2. Найти в матрице первый столбец, все элементы которого положительны. Знаки элементов предыдущего столбца изменить на противоположные. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, m, rmn = -3, rmx = 3, index = -1;  bool flag = false;  //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';  }  cout << '\n';  for (int i = 0; i < m; i++) {  for (int j = 0; j < n; j++) {  if (ptr[j][i] < 0) { //Калі гэта адмоўны, то мяняем сцяг і выходзім з цыкла  flag = true;  break;  }  }  if (!flag) {  index = abs(i-1); //захоўваем індэкс папарядняга стаўбца. калі гэта быў самы першы стаўбец, то каб не было памылак бяру праз модуль і тады будзе наступны.  for (int j = 0; j < n; j++) {// Выводзім стаўбуц, для якога ўсе лікі неадмоўныя  cout << ptr[j][i] << "\n";  ptr[j][index] = -ptr[j][index]; //мяняем папярэдні стаўбец на процілеглыя значэнні  }  cout << "\n";  break; //Как выводзіўся только першы стаўбец, для якога выконваецца ўмова  }  flag = false;  }  //Калі індэкс роўны -1, то не было неадмоўнага стаўбца і значыць мы не мянялі значэнні для папаярэдняга. інакш выводзім папярэдні стаўбец  if (index == -1) {  cout << "Няма радкоў, дзе ўсе лікі неадмоўныя";  }  else {  for (int j = 0; j < n; j++) {  cout << ptr[j][index] << "\n";  }  }  //Выдаляем масіў з памяці  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  } |
|  |

Варыянт 4

|  |
| --- |
| 1. В одномерном массиве, состоящем из **n** вещественных элементов, вычислить номер минимального элемента массива и сумму элементов массива, расположенных между первым и вторым отрицательными элементами. |
| #include <iostream>  #include <Windows.h>  #include <ctime>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, indexOfMinValue = 0;  double rmn = -20, rmx = 20, minValue, sumBetweenTwoNegatives = 0.0;  bool wasFound = false;  cout << "Увядзіце памер масіву: "; cin >> n;  double\* ptr = (double\*)malloc(n \* sizeof(double));  srand((unsigned)time(NULL));  //Ствараем масіў з выпадковых лікаў і адразу выводзім яго  for (int i = 0; i < n; i++) {  \*(ptr + i) = ((double)rand() / (double)RAND\_MAX) \* (rmx - rmn) + rmn;  cout << \*(ptr + i) << " ";  }  minValue = \*(ptr + indexOfMinValue); //За мінімальны лік бяру першы лік масіву  for (int i = 0; i < n; i++) {  if (\*(ptr + i) < minValue) { //Калі лік менш за мінімальны, захаваны раней, то захоўваем яго як мінімальны і яго індэкс  minValue = \*(ptr + i);  indexOfMinValue = i;  }  if (!wasFound && \*(ptr + i) < 0) { //Калі лік адмоўны і wasFound стаіць на false  wasFound = true; //Мяняем на true как наступныя адмоўныя не ўлічвалісь  for (int k = i + 1; k < n; k++) { //перабіраем усе лікі ад пазіцыі гэтага адмоўнага і да канца  if (\*(ptr + k) < 0) { //калі сустрэўся адмоўны, то выходзім з цыкла  break;  }  else {  sumBetweenTwoNegatives += \*(ptr + k); //інакш павялічваем сумму  }  }  }  }  cout << "\nІндэкс найменшага ліка: " << ++indexOfMinValue; //паколькі індэкс з 0, то павялічваем на 1  cout << "\nСума паміж першымі 2 адмоўнымі: " << sumBetweenTwoNegatives;  } |
|  |
| 2. Дана целочисленная прямоугольная матрица. Определить количество столбцов, не содержащих ни одного нулевого элемента |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, m, rmn = -3, rmx = 3, index = 0;  bool flag = false;  //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';  }  for (int i = 0; i < m; i++) {  for (int j = 0; j < n; j++) {  if (ptr[j][i] < 0) { //Калі гэта адмоўны, то мяняем сцяг і выходзім з цыкла  flag = true;  break;  }  }  if (!flag) {  index++; //павялічваем лічыльнік калі не было сустрэта адмоўных  }  flag = false;  }  cout <<"Неадмоўных стаўбцоў: " << index;  //Выдаляем масіў з памяці  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  } |
|  |

Варыянт 8

|  |
| --- |
| 1. В одномерном массиве, состоящем из **k** целых элементов, вычислить количество положительных элементов массива и сумму элементов массива, расположенных после последнего элемента, равного нулю. |
| #include <iostream>  #include <Windows.h>  #include <ctime>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, indexOfLastZero, counter = 0;  int rmn = -5, rmx = 5, sumOfPositive = 0;  bool wasFound = false;  cout << "Увядзіце памер масіву: "; cin >> n; indexOfLastZero = n; //Адразу ставім індэкс апошняга 0 на n, как калі няма 0, то сума не будзе знаходзіцца і будзе 0  double\* ptr = (double\*)malloc(n \* sizeof(double));  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) << " ";  }  for (int i = 0; i < n; i++) {  if (\*(ptr + i) > 0) { //Калі лік больш за 0, то павялічваем лічыльнік  counter++;  }  if (\*(ptr + i) == 0) { //Калі 0, то захоўваем яго індэкс  indexOfLastZero = i;  }  }  for (int i = indexOfLastZero + 1; i < n; i++) { //Знаходзім суму лікаў пасля апошняга 0  sumOfPositive += \*(ptr + i);  }  cout << "\nКолькасць: " << counter;  cout << "\nСума : " << sumOfPositive;  } |
|  |
| 2. Дана целочисленная прямоугольная матрица. Определить номер первого из столбцов, содержащих хотя бы один нулевой элемент. |
| #include <iostream>  #include <Windows.h>  using namespace std;  int main()  {  SetConsoleOutputCP(1251);  int n, m, rmn = -10, rmx =10, index =NULL;  bool flag = false;  //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';  }  for (int i = 0; i < m; i++) {  for (int j = 0; j < n; j++) {  if (ptr[j][i] == 0) { //Калі гэта 0, то захоўваем індэкс гэтага стаўбца і выходзім з цыкла  index = i;  flag = true;  break;  }  }  if (flag) { //калі індэкс знойдзен, то выводзім яго і выходзім з цыкла  cout << ++index; //++ бо індэксы пачынаюся з 0, а ў матэматыцы з 1  break;  }  }  //Выдаляем масіў з памяці  for (int i = 0; i < n; ++i)  delete[] ptr[i];  delete[] ptr;  } |
|  |