**ЛАБОРАТОРНА РОБОТА № 5**

**Обробка масивів в програмах на С++**

**Мета роботи**

отримати практичні навики по опису і обробці одновимірних і двовимірних масивів на мові C++

**Варіант 14**

Знайти:

1. добуток від’ємних елементів масиву;
2. кількість елементів з парним номером, менших віддалений від середнього арифметичного всіх елементів масиву;
3. номер найбільшого із елементів масиву;

**Формалізація задачі**

1. Добуток від'ємних елементів масиву:
   * Вхідні дані: Масив цілих чисел.
   * Вихідні дані: Ціле число - добуток всіх від'ємних елементів масиву.
   * Алгоритм:
     + Проходимося по кожному елементу масиву.
     + Якщо елемент від'ємний, додаємо його до змінної int dobutok, яка зберігає добуток.
     + Повертаємо результат - добуток від'ємних елементів.
2. Кількість елементів з парним номером, менших віддалений від середнього арифметичного всіх елементів масиву:
   * Вхідні дані: Масив цілих чисел.
   * Вихідні дані: Ціле число - кількість елементів з парними індексами, менших за середнє арифметичне всіх елементів масиву.
   * Алгоритм:
     + Знайдіть середнє арифметичне всіх елементів масиву.
     + Проходимося по елементам масиву з парними індексами.
     + Якщо елемент менший за середнє арифметичне, збільшуємо лічильник.
     + Повертаємо результат - кількість таких елементів.
3. Номер найбільшого із елементів масиву:
   * Вхідні дані: Масив цілих чисел.
   * Вихідні дані: Ціле число - номер (індекс) найбільшого елемента в масиві.
   * Алгоритм:
     + Знайдіть максимальний елемент в масиві і його значення.
     + Проходимося по кожному елементу масиву та знаходимо індекс першого входження максимального значення.
     + Повертаємо результат - номер найбільшого елемента (індекс).

**Лістинг програми:**

#include <iostream>

#include <ctime>

using namespace std;

int main()

{

const int n = 20;

double serarefm, suma = 0; int min\_diff = INT\_MAX, number\_min\_diff = -1;

int arr[n], dobutok = 1,kilkyst = 0,kilkyst\_negative = 0,maxIndex = 0,maxElement = arr[0],maxIndices[n],count = 0;

srand(time(NULL));

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

arr[i] = rand() % 21-10;

if (arr[i] < 0) {

dobutok \*= arr[i];

kilkyst\_negative++;

}

suma += arr[i];

cout <<"["<<i+1<<"]" << arr[i] << endl;

if (arr[i] > maxElement) {

maxElement = arr[i];

count = 0;

}

if (arr[i] == maxElement) {

maxIndices[count] = i;

count++;

}

}

serarefm = suma / n;

for (int i = 0; i < 20; i += 2) {

int diff = abs(serarefm - arr[i]);

if (diff < min\_diff) {

min\_diff = diff;

number\_min\_diff = i;

kilkyst = 1;

}

else if (diff == min\_diff) {

kilkyst++;

}

}

if (kilkyst\_negative > 0) {

cout << "\nDobutok: " << dobutok << endl;

}

else cout << "Negative elements not found!"<<endl;

cout << "Serarefm: " << serarefm << endl;

cout << "Kilkyst: " << kilkyst << endl;

cout << "max numbers index: ";

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

cout << maxIndices[i]+1;

if (i < count - 1) {

cout << ", ";

}

}

cout << endl;

}

**Результат:**

|  |
| --- |
|  |

**Задача 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 14 | D | 8,8 | Знайти суму квадратів діагональних елементів матриці. | цілі |

**Формалізація задачі:**

1. Завдання:
   * Для заданої матриці D розміром 8x8 знайти суму квадратів її діагональних елементів.
2. Вхідні дані:
   * Масив D розміром 8x8, де кожен елемент є цілим числом.
3. Вихідні дані:
   * int sum\_of\_squares- сума квадратів діагональних елементів матриці D.
4. Алгоритм:
   * Проходимося по діагональним елементам матриці D (елементи, які знаходяться на позиціях D[i][i]).
   * Додаємо квадрат кожного діагонального елемента до змінної, яка зберігає суму квадратів.
   * Повертаємо результат - суму квадратів діагональних елементів.

**Лістинг програми:**

#include <iostream>

#include <ctime>

using namespace std;

int main()

{

setlocale(LC\_ALL, "ukr");

const int n=8;

int D[n][n];

srand(time(NULL));

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

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

D[i][j] = rand() % 21-10;

}

}

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

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

cout << D[i][j] << "\t";

}

cout << endl<<endl;

}

int sum\_of\_squares = 0;

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

sum\_of\_squares += D[i][i] \* D[i][i];

}

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

sum\_of\_squares += D[i][3 - i] \* D[i][3 - i];

}

cout << endl << "Сума квадратів елементів на всіх діагоналях матриці: " << sum\_of\_squares << endl;

return 0;

}

**Результат:**

|  |
| --- |
|  |

**Задача 3**

14. За вказівкою користувача визначити або кількість від'єм­них елементів головної діагоналі матриці m(5,5), або суму еле­ментів, що знаходяться вище головної діагоналі.

**Формалізація задачі**

1. Завдання:
   * Користувач вибирає, чи він хоче обчислити кількість від'ємних елементів головної діагоналі, чи суму елементів, що знаходяться вище головної діагоналі матриці m(5,5).
2. Вхідні дані:
   * Матриця m(5,5), де кожен елемент є цілим числом.
3. Вихідні дані:
   * Залежно від вибору користувача:
     + Кількість від'ємних елементів головної діагоналі (якщо користувач обрав цей варіант).
     + Сума елементів, які знаходяться вище головної діагоналі (якщо користувач обрав цей варіант).
4. Алгоритм:
   * Вивести користувачу меню з двома варіантами: "1 - Кількість від'ємних елементів головної діагоналі" і "2 - Сума елементів вище головної діагоналі".
   * Зчитати вибір користувача.
   * Відповідно до вибору користувача:
     + Для першого варіанту:
       - Проходимося по головній діагоналі матриці і обчислюємо кількість від'ємних елементів.
       - Повертаємо кількість від'ємних елементів головної діагоналі.
     + Для другого варіанту:
       - Проходимося по всім елементам, які знаходяться вище головної діагоналі, і обчислюємо їх суму.
       - Повертаємо суму цих елементів.

**Лістинг програми:**

#include <iostream>

#include <ctime>

using namespace std;

int main()

{//14

srand(time(NULL));

setlocale(LC\_ALL, "ukr");

const int n = 5;

int negativeCount = 0;

int sumDiagonal = 0;

int m[n][n],a;

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

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

m[i][j] = rand() % 3 - 1;

}

}

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

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

cout << m[i][j] << "\t";

}

cout << endl<<endl;

}

cout << "1 - К1льк1сть в1д'ємних елемент1в головної д1агонал1 \"i\" 2 - Сума елемент1в вище головної д1агонал1 : "; cin>> a;

switch (a)

{

case 1:

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

if (m[i][i] < 0) {

negativeCount++;

}

}

cout << endl << "К1льк1сть в1д'ємних елемент1в д1агонал1: " << negativeCount<<endl;

break;

case 2:

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

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

if(i+j>n-1) sumDiagonal += m[i][j];

}

}

cout << "Сума елемент1в, що знаходяться вище головної д1агонал1: " << sumDiagonal << endl;

break;

default:

cout<<endl << "Хибний вибір!";

break;

}

}

**Результати:**

|  |
| --- |
|  |
|  |

*.*