Беларускі дзяржаўны тэхналагічны ўніверсітэт

Факультэт інфармацыйных тэхналогій

Кафедра праграмнай інжэнерыі

 Лабараторная работа 17

Па дысцыпліне «Асновы алгарытмізацыі і праграмавання»

На тэму «Паказальнікі і спасылкі пры рабоце з функцыямі»

Выканала:

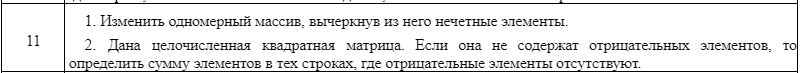
Студэнтка 1 курса 6 группы

Лускіна Вольга Аляксандраўна

Выкладчык: асс. Андронава М.В.

2023, Мінск

Варыянт 11



Заданне 1

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

int\* vidaliaem\_niaczot\_func(int\* a, int& n) { // Функцыя для выдалення няцотных элементаў

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

if (a[i] % 2 != 0) {

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

a[j] = a[j + 1];

}

n--;

i--;

}

}

return a;

}

int main() { // Функцыя main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n;

cout << "Увядзіце памер масіва: ";

cin >> n;

int\* a = new int[n];

cout << "Увядзіце элементы масіва: " << endl;

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

cin >> a[i];

}

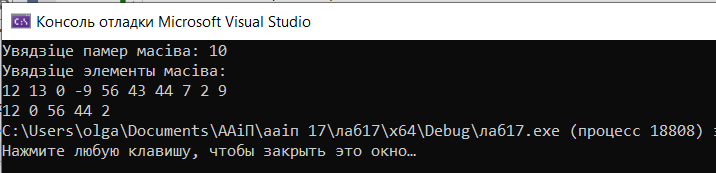
vidaliaem\_niaczot\_func(a, n); // Выклік функцыі выдалення няцотных элементаў

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

cout << a[i] << ' ';

}

}



Заданне 2

#include <iostream>

#include <windows.h>

#include <cstdlib>

using namespace std;

void poshuk\_sum(int\*\* a, int n) {

bool znoideniAdmouni = false; // Зменная для пошуку адмоўнага элемента

int admouniRow = -1; // Індэкс радка з адмоўным элементам

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

bool admouni = false; // Зменная, каб паглядзець, ці ёсць адмоўны элемент у бягучым радку

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

if (a[i][j] < 0) {

admouni = true; // Згойдзен адмоўны элемент

admouniRow = i; // Запамінаем індэкс радка з адмоўным элементам

break;

}

}

if (admouni) {

znoideniAdmouni = true; // Знойдзен адмоўны ээлемент у матрыцы

break;

}

}

if (znoideniAdmouni) {

cout << "У матрыцы ёсць адмоўны элемент у радку з індэксам " << admouniRow << endl;

}

else {

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

int sum = 0; // Зменная для падліку сумы элементаў радка

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

sum += a[i][j]; // Падлік сумы элементаў радка

}

cout << "Сума элементаў радка з індэксам " << i << " роўная " << sum << endl;

}

}

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n;

cout << "Увядзіце памер матрыцы: ";

cin >> n;

int\*\* a = new int\* [n];

cout << "Увядзіце элементы матрыцы: " << endl;

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

a[i] = new int[n];

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

cin >> a[i][j];

}

}

cout << endl;

poshuk\_sum(a, n);

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

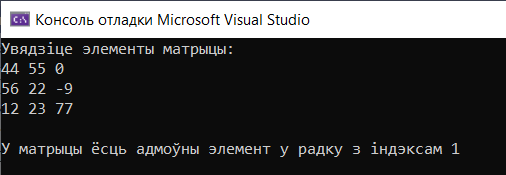
delete[] a[i];

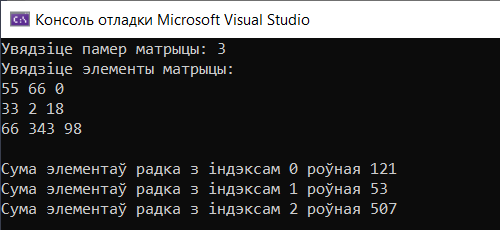
}

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

return 0;

}





Дадатковыя заданні:

Варыянт 12



Заданне 1

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

void LichimAdmounie(int\* element, int pamer) // Функцыя для падліку колькасці адмоўных элементаў , якія стаяць на цотных месцах у масіве

{

int Admounie = 0;

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

if (\*(element + i) < 0) {

if (i % 2 != 0) {

Admounie++;

}

}

}

cout << "Колькасць адмоўных элементаў, якія стаяць на цотных месцах у масіве: " << Admounie << endl;

}

void main(){ // Функцыя main

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n;

cout << "Увядзіце памер масіва: ";

cin >> n;

int\* a = new int[n];

cout << "Увядзіце элементы масіва: " << endl;

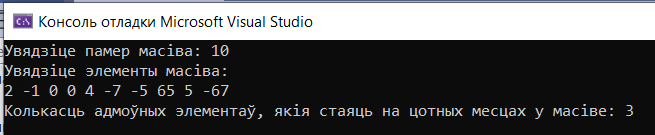
for (int i = 0; i < n; i++) {

cin >> a[i];

}

LichimAdmounie(a, n); // Выклік функцыі падліку колькасці няцотных элементаў

}



Заданне 2

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

bool RadokzAdmounimLikam(int\*\* a, int n, int k, int& radokNumber)

{

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

bool Admouni = false; // Сцяг для пошуку адмоўнага элемента ў радку

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

if (a[i][j] < 0) {

Admouni = true;

break; // Перарываем цыкл, бо нам дастаткова аднаго адмоўнага

}

}

if (Admouni) { // Калі знайшлі адмоўны элемент

radokNumber = i; // Запісваем нумар радка ў зменную

return true;

}

}

return false; // Калі не знайшлі адмоўны элемент

}

void YDvaRaziPamenshili(int\*\* a, int n, int k, int slupNumber)

{

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

a[i][slupNumber] /= 2; // Памяншаем элемент слупка ў два разы

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n, k;

cout << "Увядзіце памер матрыцы: ";

cin >> n >> k;

int\*\* a = new int\* [n];

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

a[i] = new int[k];

}

cout << "Увядзіце элементы матрыцы:" << endl;

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

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

cin >> a[i][j];

}

}

int radokNumber;

if (RadokzAdmounimLikam(a, n, k, radokNumber)) {

cout << "Нумар радка з адмоўным элементам: " << radokNumber << endl;

YDvaRaziPamenshili(a, n, k, radokNumber);

cout << "Новая матрыца:" << endl;

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

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

cout << a[i][j] << " ";

}

cout << endl;

}

}

else {

cout << "У матрыцы няма радка з адмоўным элементам" << endl;

}

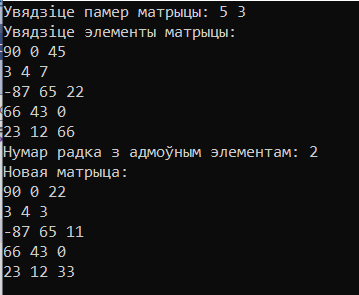
for (int i = 0; i < n; i++) {

delete[] a[i];

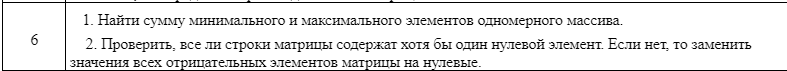
}

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

}



Варыянт 6



Заданне 1

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

void FindMinMaxSum(int\* a, int size, int& min, int& max, int& sum) // Функцыя для падліку сумы найменшага і найбольшага элементаў масіва

{

min = a[0];

max = a[0];

sum = min + max;

for (int i = 1; i < size; i++) {

if (a[i] < min) { // Калі бягучы элемент меншы за найменшы

min = a[i]; // Аднаўляем найменшы элемент

}

if (a[i] > max) { // Калі бягучы элемент большы за найбольшы

max = a[i]; // Аднашляем найбольшы элемент

}

sum = min + max; // Аднаўляем суму

}

}

int main(){

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int size;

cout << "Увядзіце памер масіва: ";

cin >> size;

int\* a = new int[size];

cout << "Увядзіце элементы масіва:" << endl;

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

cin >> a[i];

}

int min, max, sum;

FindMinMaxSum(a, size, min, max, sum);

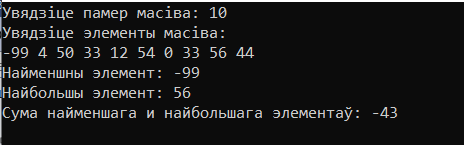
cout << "Найменшны элемент: " << min << endl;

cout << "Найбольшы элемент: " << max << endl;

cout << "Сума найменшага и найбольшага элементаў: " << sum << endl;

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

}



Заданне 2

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

void PoshukNuliouIPeratvarenneAdmounixLikau(int\*\* a, int n, int k) // Функцыя для прагляду радкоў матрыцы і замены адмоўных лікаў

{

bool NuliYRadkax = true; // Сцяг для пошуку нулявых элементаў у радках

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

bool NauavnostNulia = false; // Сцяг для пошуку нулявога элемента

bool NauavnostAdmounaga = false; // Сцяг для пошуку адмоўнага элемента

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

if (a[i][j] == 0) { // Калі знайшлі нулявы элемент

NauavnostNulia = true;

}

if (a[i][j] < 0) { // Калі знайшлі адмоўны элемент

NauavnostAdmounaga = true;

a[i][j] = 0; // Ператвараем адмоўны ў нулявы

}

}

if (!NauavnostNulia) { // Калі бягучы радок не мае нулявых элементаў

NuliYRadkax = false;

}

}

if (!NuliYRadkax) { // Калі не ўсе радкі маюць нулявыя элементы

cout << "Не ўсе радкі маюць нулявы элемент. Усе адмоўныя лікі ператвораны ў нулявыя" << endl;

}

else {

cout << "Усе радкі маюць хаця б адзін нулявы элемент" << endl;

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n, k;

cout << "Увядзіце памер матрыцы: ";

cin >> n >> k;

// Выделение памяти для матрицы

int\*\* a = new int\* [n];

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

a[i] = new int[k];

}

// Ввод элементов матрицы

cout << "Увядзіце элементы матрыцы: " << endl;

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

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

cin >> a[i][j];

}

}

PoshukNuliouIPeratvarenneAdmounixLikau(a, n, k);

cout << "Новая матрыца:" << endl;

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

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

cout << a[i][j] << " ";

}

cout << endl;

}

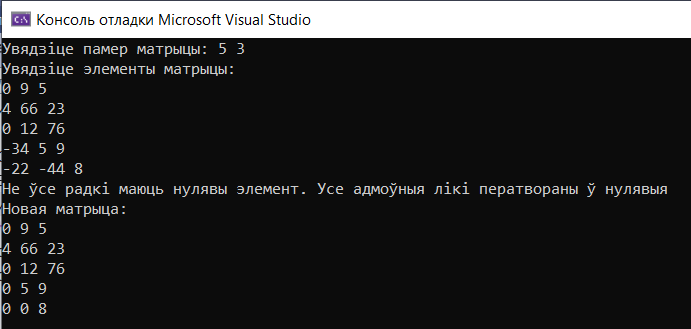
for (int i = 0; i < n; i++) {

delete[] a[i];

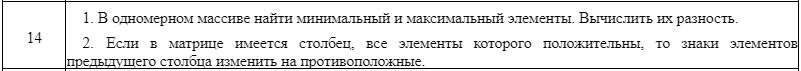
}

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

}



Варыянт 14



Заданне 1

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

void FindMinMaxSum(int\* a, int size, int& min, int& max, int& sum) // Функцыя для падліку сумы найменшага і найбольшага элементаў масіва

{

min = a[0];

max = a[0];

sum = max - min;

for (int i = 1; i < size; i++) {

if (a[i] < min) { // Калі бягучы элемент меншы за найменшы

min = a[i]; // Аднаўляем найменшы элемент

}

if (a[i] > max) { // Калі бягучы элемент большы за найбольшы

max = a[i]; // Аднашляем найбольшы элемент

}

sum = max - min; // Аднаўляем суму

}

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int size;

cout << "Увядзіце памер масіва: ";

cin >> size;

int\* a = new int[size];

cout << "Увядзіце элементы масіва:" << endl;

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

cin >> a[i];

}

int min, max, sum;

FindMinMaxSum(a, size, min, max, sum);

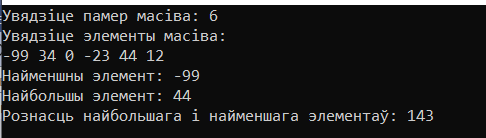
cout << "Найменшны элемент: " << min << endl;

cout << "Найбольшы элемент: " << max << endl;

cout << "Рознасць найбольшага і найменшага элементаў: " << sum << endl;

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

}



Заданне 2

#include <iostream>

#include <windows.h>

#include <cstdlib> // Бібліятэка для выкарыстання функцый, звязаных з выдзяленнем і кірававаннем дынамічнай памяці

using namespace std;

void PoshukPounasczuStanouchagaSlupka(int\*\* a, int n, int k) // Функцыя для пошуку слупка, у якім усе элементы станоўчыя, і, калі такі ёсць, змены знакаў мінулага слупка

{

bool StanouchiSlupokZnoidzen = false; // Сцяг для пошуку патрэбнага слупка

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

bool allStanouchiua = true;

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

if (a[i][j] <= 0) { // Глядзім, ці ўсе зменныя больш за ноль

allStanouchiua = false;

break;

}

}

if (allStanouchiua) {

StanouchiSlupokZnoidzen = true;

if (j > 0) {

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

if (a[i][j - 1] < 0) { // Калі адмоўнае, робім станоўчым

a[i][j - 1] = -a[i][j - 1];

}

else if (a[i][j - 1] > 0) { // Калі станоўчае, робім адмоўным

a[i][j - 1] = -a[i][j - 1];

}

}

}

break;

}

}

if (StanouchiSlupokZnoidzen) {

cout << "У матрыцы ёсць слупок, у якога ўсе элементы станоўчыя, таму знакі мінулага зменены на супрацьлеглыя" << endl;

}

else {

cout << "У матрыцы няма слупка, усе элементы якога станоўчыя" << endl;

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n, k;

cout << "Увядзіце памер матрыцы: ";

cin >> n >> k;

int\*\* a = new int\* [n];

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

a[i] = new int[k];

}

cout << "Увядзіце элементы матрыцы: " << endl;

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

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

cin >> a[i][j];

}

}

PoshukPounasczuStanouchagaSlupka(a, n, k);

cout << "Новая матрыца:" << endl;

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

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

cout << a[i][j] << " ";

}

cout << endl;

}

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

delete[] a[i];

}

delete[] a; // Ачышчаем памяць, якую выдзялілі для двумернага масіва

}

