Лабораторна робота 8

Завдання до лабораторних робіт з курсу «Програмування» (1 курс)

Завдання:

# Тема. : " Двовимірні масиви. Робота з файлами.".

**1.Задвання**

При виконанні лабораторної роботи слід врахувати наступне:

1. Дати користувачеві можливість вибору: вводити вхідні дані з консолі, чи вводити їх з файлу; у випадку вводу з консолі продемонструвати можливість пере напрямлення потоку вводу/виводу на файл.
2. При програмуванні однотипних підзадач (сортування, ввід/вивід масивів, знаходження суми стовпця матриці тощо) слід оформлювати їх у вигляді функцій.

**Варіант 20.** Задайте матрицю *А* розмірністю (*n* x *n*), де *n* ≥ 7. Виконайте такі завдання:

1. знайдіть мінімальний і максимальний елементи кожного рядка матриці;
2. обчисліть суму елементів у тих стовпцях, де немає від’ємних елементів;
3. упорядкуйте за спаданням значення елементів головної діагоналі.

**2.Опис розв’язку**

**Я використовую динамічний двомірний масив.  
Та кожну підзадачу розбивав на функції щоб можна було використовувати один і той самий код при різних вводах матриці**

**Для роботи з файлами використовується бібліотека** fstream

**Потрібно було зробити перевірку на те чи є файл з якого намагаються зчитати матрицю**

**Якщо ні, пропонується створити його**

**Також при перенапрямленні потоку вводу виводу потрібно вказати і елементи меню та вихід (в кінці файлу 0)**

**Для пошуку мін і макс на кожному рядку матриці, в подвійному циклі одночасно порівнюються числа. Первісно мін, макс = першому числу в рядку**

**Для задачі з сортуванням я зробив окремий масив з головною діагоналлю, і потім при виводі всієї матриці, коли i==j я використовую значення з відсортованого масиву головної діагоналі**

**3.Код програми**

//

// main.cpp

// lb2

//

// Created by Дима Филипенко on 07.03.2020.

// Copyright © 2020 Дима Филипенко. All rights reserved.

//

#include <iostream>

#include <fstream>

**using** **namespace** std;

**void** InsertArray (**int** \*\*A,**int** N)

{

**int** a;

**for**(**int** i=0; i<N;i++){

**for**(**int** j=0;j<N;j++){

cout<<"N["<<i+1<<"]["<<j+1<<"]=";

cin>>a;

// A[i][j]=a;

\*( \*(A + i) + j ) = a;

}

}

}

**void** ShowArray (**int** \*\*A,**int** N)

{

**for** (**int** i = 0; i < N; i++) {

**for** (**int** j = 0; j < N; j++)

cout << A[i][j] << ' ';

cout << '\n';

}

}

**void** MinMax(**int** \*\*A, **int** N){

cout<<"Завдання 1 - мінімальний та максимальний елемент в кожному рядку матриці"<<endl;

**for** (**int** i = 0; i < N; i++) {

**int** min = A[i][0];

**int** max = min;

**for** (**int** j = 0; j < N; j++){

**if**(A[i][j]>max){

max=A[i][j];

}

**if**(A[i][j]<min){

min=A[i][j];

}

}

cout<<"Рядок №"<<i<<" "<<min<<" - min "<<max<<" - max"<<endl;

}

}

**void** SumWithoutMin(**int** \*\*A, **int** N){

cout<<"завданя 2 Cума елементів у тих стовпцях, де немає від’ємних елементів;"<<endl;

**for** (**int** j = 0; j < N; j++){

**int** S=0;

**bool** Sum\_is\_exist = **true**;

**for** (**int** i = 0; i < N; i++) {

**if**(A[i][j]<0){

Sum\_is\_exist = **false**;

}**else**{

S+= A[i][j];

}

}

**if**(!Sum\_is\_exist){

cout<<"Стовпецб №"<<j<<" наявний від'ємний елемент "<<endl;

}

**else**{

cout<<"Стовпецб №"<<j<<" сума дорівню "<<S<<endl;

}

}

}

**void** SortSwap(**int** a, **int** b){

**int** t=a;

a=b;

b=t;

}

**void** SortMAinDiagonal(**int** \*\*A,**int** N){

cout<<"Завдання 3 упорядкувати за спаданням значення елементів головної діагоналі"<<endl;

**int** \* mainDiagonal;

mainDiagonal = **new** **int** [5];

**for**(**int** i=0; i<N;i++){

**for**(**int** j=0;j<N;j++){

**if** (i==j)

mainDiagonal[i] = A[i][j];

}

}

**bool** swap;

**int** temp;

**do**

{

swap = **false**;

**for** (**int** count = 0; count < (N-1); count++)

{

**if** ((mainDiagonal[count]) < (mainDiagonal[count+1]))

{

temp = mainDiagonal[count];

mainDiagonal[count] = mainDiagonal[count+1];

mainDiagonal[count+1] = temp;

swap = **true**;

}

}

}

**while**(swap);

cout<<"Головна діагональ упорядкована за спаданням"<<endl;

**for** (**int** i = 0; i < N; i++) {

**for** (**int** j = 0; j < N; j++){

**if**(i==j){

cout<<mainDiagonal[i]<<" ";

}**else**{

cout << A[i][j] << ' ';

}

}

cout << '\n';

}

}

**void** InputFromConsole(){

**int** N;

**do**{

cout<<"Введіть розмір квадратної матриці NxN"<<endl;

cin>>N;

**if**(N<7){

printf("\x1B[31m має бути більша 7 \033[0m\t\t\n");

**continue**;

}

}**while** (N<7);

**int** \*\* A= **new** **int**\*[N];

**for**( **int** i=0; i<N; i++){

A[i]=**new** **int**[N];

}

InsertArray(A,N);

cout<<endl;

ShowArray(A,N);

cout<<endl;

MinMax(A,N);

cout<<endl;

SumWithoutMin(A,N);

cout<<endl;

SortMAinDiagonal(A,N);

cout<<endl;

}

**bool** InputFromFile(){

**int** N;

**char** fileName[50];

**int** a;

fstream F;

cout<<"Введіть назву файла"<<endl;

cin>>fileName;

F.open(fileName);

**if** (F)

{

**if**(! (F>>N)){

cout<<"Файл пустий"<<endl;

**return** **false**;

}

//зчитуємо клькість

**if**(N>=7){

**int** \*\* A= **new** **int**\*[N];

**for**( **int** i=0; i<N; i++){

A[i]=**new** **int**[N];

}

**for**(**int** i = 0; i<N;i++){

**for**(**int** j = 0; j<N;j++){

**if**(!(F>>a)){

cout<<"Недостатня кількість елементів матриці"<<endl;

**return** **false**;

}

}

}

F.seekg(0, ios::beg);

F>>N;

**for**(**int** i = 0; i<N;i++){

**for**(**int** j = 0; j<N;j++){

F>>A[i][j] ;

}

// F<<endl;

}

ShowArray(A,N);

cout<<endl;

MinMax(A,N);

cout<<endl;

SumWithoutMin(A,N);

cout<<endl;

SortMAinDiagonal(A,N);

cout<<endl;

}**else**{

cout<<"N має бути більше або дорівнювати 7"<<endl;

**return** **false**;

}

//закрытие потока

// cout<<N<<endl;

F.close();

}**else** {

cout<<"Файла не існує"<<endl;

cout<<"Бажаєте створити? (y/n)"<<endl;

**char** answer[1];

cin>>answer[0];

**if**(answer[0] == 'y'){

ofstream file { fileName };

file.close();

cout<<"Файл "<<fileName<<" Створено успішно"<<endl;

}**else**{

cout<<"Файл "<<fileName<<" Не створено"<<endl;

// return false;

}

}

**return** **true**;

}

**void** CreateMatrix(){

**int** N=8;

**char** fileName[50];

// fstream F;

cout<<"Введіть назву файла"<<endl;

cin>>fileName;

cout<<fileName;

// cin.getline(fileName,50);

ofstream F(fileName);

**if** (F)

{

F.ios\_base::clear();

F<<N;

F<<endl;

srand(time(0));

**for** (**int** i = 0; i<N; i++) {

**for** (**int** j = 0; j<N; j++) {

F<<rand()% (10-(-10)+1) + (-10)<<" ";

}

F<<endl;

}

F.close();

}

}

**int** main(**int** argc, **const** **char** \* argv[]) {

**int** k;

**do**{

cout<<"1.Ввід з консолі"<<endl;

cout<<"2.Ввід з файлу"<<endl;

cout<<"3.Згенерувати матрицю в файлі"<<endl;

cout<<"0.Вихід"<<endl;

cin>>k;

**switch** (k) {

**case** 1:

InputFromConsole();

**break**;

**case** 2:

InputFromFile();

**break**;

**case** 3:

CreateMatrix();

**break**;

**default**:

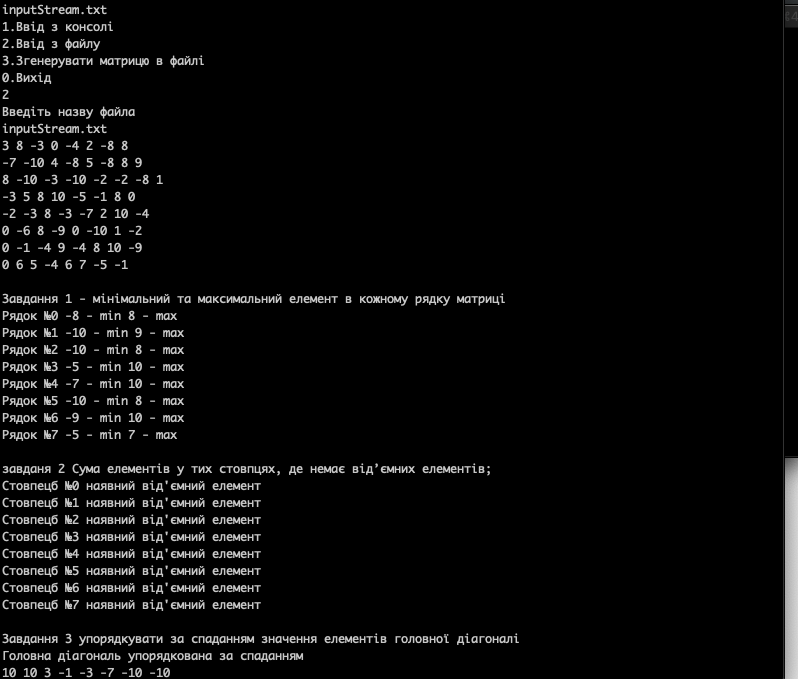
**break**;

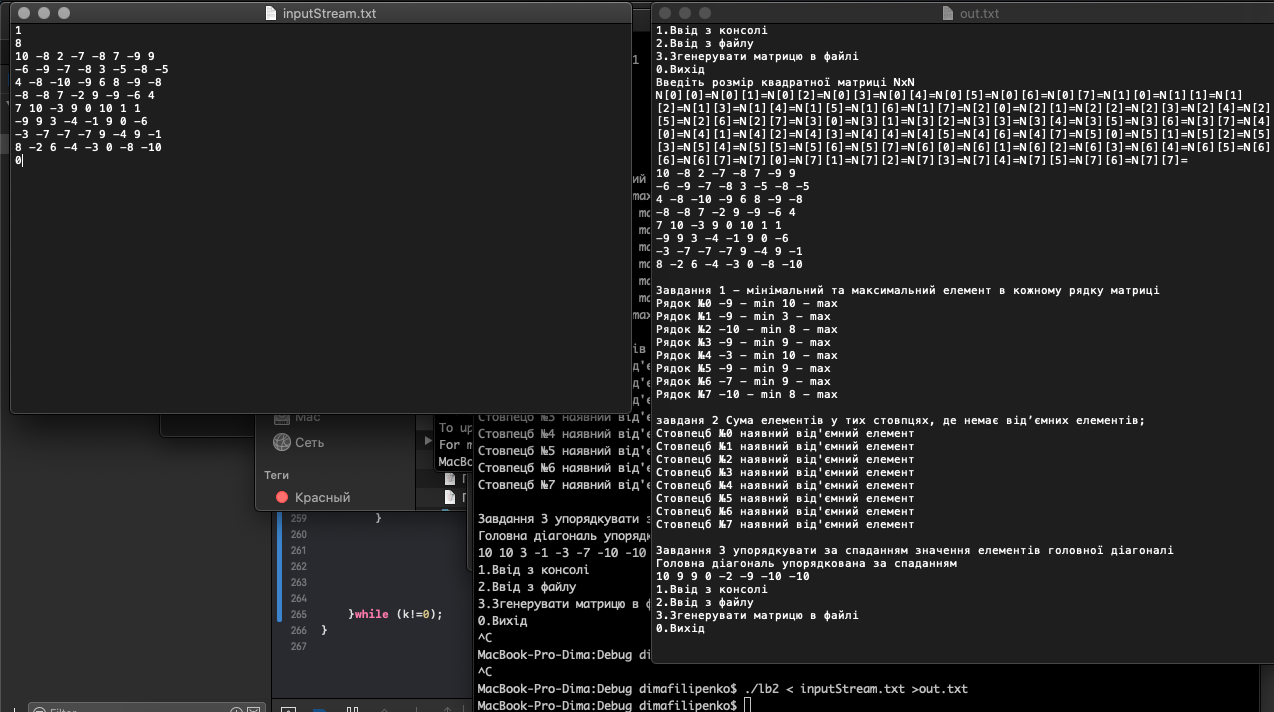
}

}**while** (k!=0);

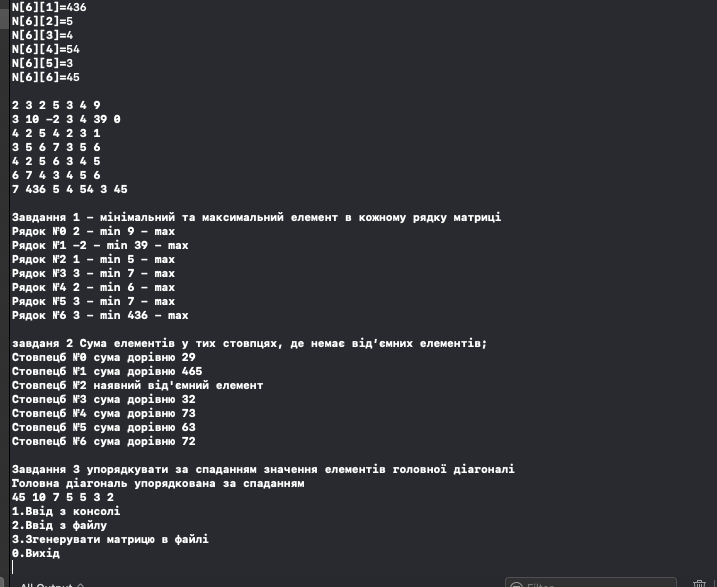
}

**4.Тести**

**Ввід з файлу**

**Перенапрямлення потоку вводу/виводу**

**Ввід з консолі**

****

**5.Керівництво користувача**

**В файлі для просто вводу з файлу потрібно вказати на 1 рядку розмір матриці.  
в файлі для перенапрямлення вводу додатково треба вкзати пункт меню на первому рядку 1 і в кінці вихід (0)**

**Для вводу з файлу, потрібно щоб він існував, якщо його не буде, програма запропонує створити його, потім його треба буде заповнити самостійно або натиснути на 3 пункт щоб заповнити автоматично**

**Додатково можна згенерувати матрицю в файлі для звичайного вводу з файлу (під меню 3)**