МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №5**

**з дисципліни «Основи програмної інженерії»**

**на тему: *«*Розробка програми зі структурою діалогу «питання-відповідь».»**

Виконав: студент гр. ПЗ2011

Горбань М.О.

Прийняла: к.т.н., доцент кафедри КІТ: Горбова О.В.

Дніпро, 2021

**Тема.** Розробка програми зі структурою діалогу «питання-відповідь».

**Мета роботи**. Отримати практичні навички розробки програм зі структурою діалогу «питання-відповідь».

**Завдання**

Розробіть програму для вирішення задачі з індивідуального завдання лабораторної роботи № 1, використавши структуру діалогу «меню». Перед розробкою виконайте проектування меню, аналогічно до прикладу,представленого у теоретичних відомостях.

Вимоги до написання програми:

– матриця представлена у вигляді структури (struct);

– кожна принципово нова дія представлена у вигляді функції. Обов’язковий

набір функцій:

* main – головна функція програми;
* create – створення матриці (зазначення розмірності, виділення пам’яті);
* delete – очищення пам’яті, яку займає матриця;
* processing – обробка матриці, не повинна містити операторів введення та/або виведення;
* manualFilling, randomFilling, readFile – заповнення матриці з клавіатури, генератором випадкових чисел та з файлу відповідно;
* show – виведення матриці на екран;

– функція/ї обробки матриці містить коментарі, виходячи з методу покрокової деталізації;

– усі функції мають коротку та змістовну специфікацію, усі ідентифікатори (назви змінних, функцій, структур тощо) відображають суть названої програмної сутності;

– використовується лише одна матриця, використання додаткових матриць (масивів) заборонено;

– наявність меню у вигляді списку об'єктів, які обирають прямим зазначенням номера. Можливість багаторазового виклику усіх пунктів меню у довільному порядку. Можливість вибору способу заповнення матриці (з клавіатури, гератором випадкових чисел, з файлу). При успішному заповненні генератором випадкових чисел та з файлу виводиться повідомлення про успішне виконання операції;

– для організації роботи меню використовується цикл з постумовою та оператор множинного вибору;

– не використовуються рекурсивні функції та оператор goto;

– відсутність витоку пам’яті;

– при заповненні матриці з файлу, ім’я файлу вказується на етапі виконання.

При відсутності вказаного файлу виводиться повідомлення для користувача;

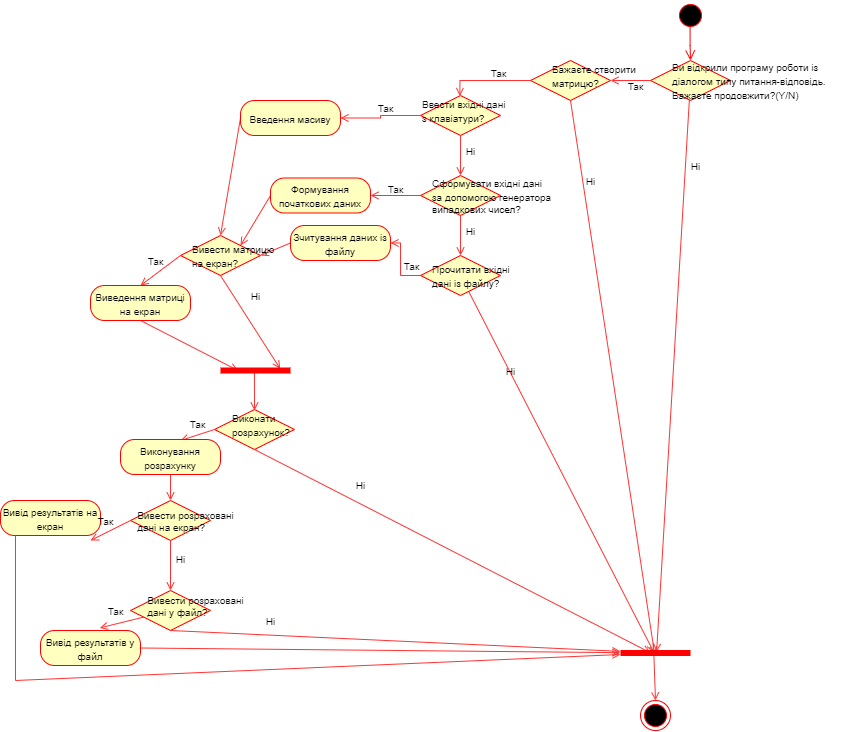
– файл, з якого заповнюється матриця, містить інформацію щодо її розмірності та значення елементів;

– перевірка коректності введення номера пункту меню, розмірності матриці (з точки зору належності області визначення).

**Індивідуальне завдання:**

6. В рядках, в яких сума першого і останнього елементів перевищує значення діагонального елемента, поміняти місцями пари елементів (перший з другим, третій з четвертим і так далі).

**1) Сценарій діалогу програми**

****

**2) Текст програми**

#include <iostream>

#include<conio.h>

#include "func.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

Matrix array = { 0, 0, nullptr };

char choice;

while (true)

{

cout << "Вас приветсвует програма работы из диалого типа вопрос-ответ!" << endl;

cout << "Хотите продолжить работу? (Y/N)" << endl;

choice = \_getch();

if ((choice == 'Y') || (choice == 'y'))

{

cout << "Хотите создать матрицу? (Y/N)" << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

cout << "Ввести входные данные с клавитуры? (Y/N) " << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

//Ввод данных

input(array);

create(array);//создание матрицы

manualFilling(array);

}

else

{

cout << "Сформировать данные с помощью генератора случайных чисел? (Y/N) " << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

//Ввод данных

input(array);

create(array);//создание матрицы

randomFilling(array);

}

else

{

cout << "Считать данные из файла? (Y/N) " << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

//Ввод данных

fileFilling(array);

}

else

{

return 0;

}

}

}

}

else

{

exit(1);

}

cout << "Желате вывести матрицу в консоль? (Y/N)" << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

show(array);

}

cout << "Поменять пары чисел в рядках\n"<<

"в которых сума первого и последнего елемента\n"<<

"больше чем диагонального ? (Y / N)" << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

processing(array);

}

else

{

exit(1);

}

cout << "Вывести результаты в консоль? (Y/N)" << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

show(array);

}

cout << "Записать результаты в файл? (Y/N)" << endl;

choice = \_getch();

if (choice == 'Y' || choice == 'y')

{

fileInput(array);

cout << "Матрица успешно записана в файл!" << endl << endl;

}

}

else

{

exit(1);

}

}

return 0;

}Файл func.cpp

#include "func.h"

#include <iostream>

#include <iomanip>

#include <ctime>

#include <fstream>

using namespace std;

/// <summary>

/// Ввод размерности матрицы

/// </summary>

/// <param name="array"></param>

void input(Matrix& array)

{

cout << "Введите количество строк: "; cin >> array.row;

cout << "Введите количество столбцов: "; cin >> array.column;

}

/// <summary>

/// Создание матрицы

/// </summary>

/// <param name="array">Масив в виде струкстуры</param>

void create(Matrix& array)

{

if (array.arr != nullptr)

{

delete\_matrix(array);

}

array.arr = new int\* [array.row];

for (int i = 0; i < array.row; i++)

{

array.arr[i] = new int[array.column];

}

}

/// <summary>

/// Отчистка памяти

/// </summary>

/// <param name="array">Масив виде структуры</param>

void delete\_matrix(Matrix& array)

{

for (int i = 0; i < array.row; i++)

{

delete[] array.arr[i];

}

delete[] array.arr;

array.arr == nullptr;

}

/// <summary>

/// Ручное заполнение матрицы

/// </summary>

/// <param name="array">Масив виде структуры</param>

void manualFilling(Matrix& array)

{

for (int i = 0; i < array.row; i++)

{

for (int j = 0; j < array.column; j++)

{

cout << "array" << '[' << i << ']' << '[' << j << "]= ";

cin >> array.arr[i][j];

cout << endl;

}

}

}

/// <summary>

/// Заполнение матрицы случайными числами

/// </summary>

/// <param name="array">Масив виде структуры</param>

void randomFilling(Matrix& array)

{

srand(time(NULL));

for (int i = 0; i < array.row; i++)

{

for (int j = 0; j < array.column; j++)

{

array.arr[i][j] = rand() % 20;

}

}

cout << "Матрица успешно заполнена." << endl;

}

/// <summary>

/// Вывод матрицы на экран

/// </summary>

/// <param name="array">Масив виде структуры</param>

void show(Matrix& array)

{

for (int i = 0; i < array.row; i++)

{

for (int j = 0; j < array.column; j++)

{

cout << setw(4) << array.arr[i][j] << ' ';

}

cout << endl << endl;

}

}

/// <summary>

/// Сравнение сум и елемента диагонали

/// </summary>

/// <param name="array">матрица</param>

/// <param name="row">номер рядка</param>

/// <returns>сума больше или меньше диагонального елемента</returns>

bool calculateSum(Matrix& array, int row)

{

int sum = array.arr[row][0] + array.arr[row][array.column - 1];

if (sum > array.arr[row][row])

return 1;

else return 0;

}

/// <summary>

/// Обработка матрицы

/// </summary>

/// <param name="array">структура з матрицей</param>

void processing(Matrix& array)

{

for (int i = 0; i < array.row; i++)

{

bool swaping = calculateSum(array, i);

if (swaping)

{

for (int j = 0; j < array.column - 1; j += 2)

{

swap(array.arr[i][j], array.arr[i][j + 1]);

}

}

}

}

/// <summary>

/// Заполнение матрицы из файла

/// </summary>

/// <param name="array">структура з матрицей</param>

void fileFilling(Matrix& array)

{

char string[50];

cout << "Введите путь к файлу c матрицей." << endl;

cin >> string;

fstream f(string);

if (f.fail())

{

cout << "Ошибка открытия файла" << endl;

}

f >> array.row;

f >> array.column;

if (array.arr != nullptr)

{

delete\_matrix(array);

}

array.arr = new int\* [array.row];

for (int i = 0; i < array.row; i++)

{

array.arr[i] = new int[array.column];

}

for (int i = 0; i < array.row; i++)

{

for (int j = 0; j < array.column; j++)

{

f >> array.arr[i][j];

}

}

f.close();

}

/// <summary>

/// Запись матрицы в файл

/// </summary>

/// <param name="array">структура з матрицей</param>

void fileInput(Matrix& array)

{

char string[50];

cout << "Введите путь к файлу для записи матрицы." << endl;

cin >> string;

fstream f(string, ios::in | ios::out | ios::app);

if (f.fail())

{

cout << "Ошибка открытия файла"<<endl;

}

f << array.row;

f << array.column;

for (int i = 0; i < array.row; i++)

{

f << endl;

for (int j = 0; j < array.column; j++)

{

f << array.arr[i][j]<<'\t';

}

}

f << endl;

f.close();

}

Файл func.h

#pragma once

struct Matrix

{

int row;

int column;

int\*\* arr;

};

void input(Matrix& array);

void fileFilling(Matrix &array);

void fileInput(Matrix& array);

void create(Matrix& array);

void delete\_matrix(Matrix& array);

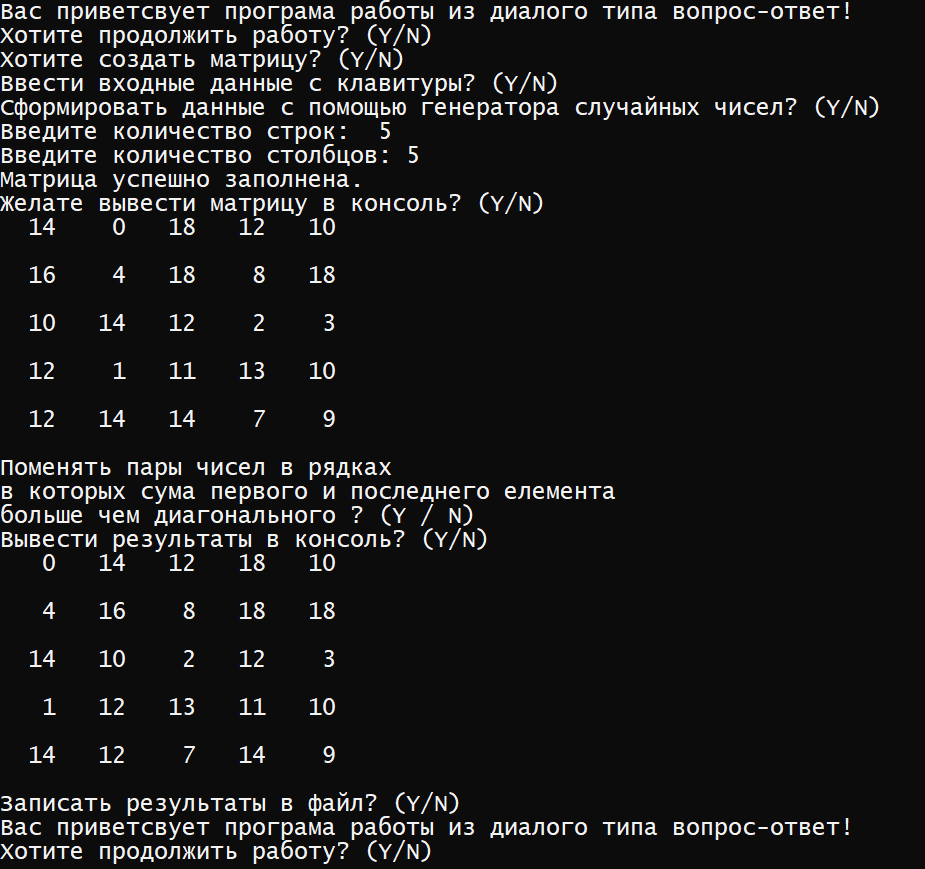
void manualFilling(Matrix& array);

void randomFilling(Matrix& array);

void show(Matrix& array);

void processing(Matrix& array);

**3) Аналіз результатів**



На скріншоті видно результат роботи програми зі структурою питання-відповідь. Програма коректно виконує свої функції.

**4) Висновок**

Під час виконання лабораторної роботи сворював меню у форматі питання-відповідь. Формат такого типу меню забезпечує чітку послідовність дій, які може вибрати користувач. На мою думку, програма з таким типом меню має менше шансів зламатися, проте витрачається більше часу під час користування програмою, адже треба проходити всі пункти меню, щоб добратися до потрібної вам дії.