МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



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

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №4**

**з дисципліни «Основи програмної інженерії»**

**на тему: *«*Розробка програми з меню»**

Виконав: студент гр. ПЗ2011

Кулик Сергій Вадимович

Прийняла: доц. Нежуміра О.І.

Дніпро, 2020

**Тема.** Розробка програми з меню.

**Мета.** Ознайомитися з поняттям сценарію та структури діалогу. Набути практичних навичок розробки програм зі структурою діалогу «меню».

**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:

14. Знайти суму додатних елементів лівої половини матриці і суму від’ємних правої половини. Визначити яка з них більша за модулем.

**1) Проект меню**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пункт (рівень 1) | № | Пункт (рівень 2) | Пункти, які треба виконати до |
| 1 | Створити матрицю | 1.1 | Заповнити випадковим чином | - |
| 1.2 | Заповнити з клавіатури |
| 1.3 | Заповнити з файлу |
| 1.4 | Повернутися до головного меню |
| 2 | Обробка матриці |  |  | 1 |
| 3 | Вивести матрицю | 3.1 | Вивести матрицю | 1 |
| 3.2 | Записати у файл |
| 3.3 | Повернутится до головного меню |
| 4 | Вихід із програми |  |  | - |

**2) Текст програми**

Файл Source.cpp

//Программа считает суму положительных елементов левой части матрици

//и суму отрицательных правой части. Определяет наибольшую из сум по модулю.

//Вход: количество строк и столбцов матрицы.

//Выход: Наибольшая сума по модулю.

#include <iostream>

#include "func.h"

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

Matrix array = { 0, 0, nullptr };

while (true)

{

outputMenu();

int choice;

cout << "Введите цифру необходимого пункта меню!!!" << endl;

cin >> choice;

switch (choice) //главное меню

{

case 1: //создать матрицу

{

bool exit = true;

while (exit)

{

outputCreateLevel2();

int choice;

cin >> choice;

switch (choice) //меню создания матрици 2ур.

{

case 1: //заполнение рандомными числами

{

create(array);

randomFilling(array);

break;

}

case 2: //ручное заполнение

{

create(array);

manualFilling(array);

break;

}

case 3: //заполнение из файла

{

fileFilling(array);

break;

}

case 4: //выход в главное меню

{

exit = false;

break;

}

case 5: //отчистка консоли

{

system("cls");

break;

}

default:

{

cout << "Такого пункта меню нету, попробуйте ещё раз!" << endl;

}

}

}

break;

}

case 2: //обработка матрицы

{

if (array.arr == nullptr)

{

cout << "Матрица еще не создана!" << endl;

}

else

{

cout << "Сума левой части матрицы = " << leftMatrix(array) << endl;

cout << "Сума правой части матрицы = " << rightMatrix(array) << endl;

cout << "Наибольшая сума по модулю = " << processing(array) << endl;

}

break;

}

case 3: //вывод матрицы

{

bool exit = true;

while (exit)

{

outputMatrixLevel2();

int choice;

cin >> choice;

switch (choice) //меню вывода матрицы 2ур.

{

case 1: //вывести матрицу в консоль

{

if (array.arr == nullptr)

{

cout << "Матрица еще не создана!" << endl;

}

show(array);

break;

}

case 2: //записать матрицу в файл

{

if (array.arr == nullptr)

{

cout << "Матрица еще не создана!" << endl;

}

fileInput(array);

break;

}

case 3: //вернутся в главное меню

{

exit = false;

break;

}

case 4: //отчистить консоль

{

system("cls");

break;

}

default:

{

cout << "Такого пункта меню нету, попробуйте ещё раз!" << endl;

}

}

}

break;

}

case 4: //выход из програмы

{

delete\_matrix(array);

exit(1);

break;

}

case 5: //отчистка консоли

{

system("cls");

break;

}

default:

{

cout << "Такого пункта меню нету, попробуйте ещё раз!" << endl;

}

}

}

}

Файл func.cpp

#include "func.h"

#include <iostream>

#include <iomanip>

#include <ctime>

#include <fstream>

using namespace std;

/// <summary>

/// главное меню

/// </summary>

void outputMenu()

{

cout << "================ МЕНЮ =================" << endl;

cout << "1. Создать матрицу." << endl;

cout << "2. Обработка матрицы." << endl;

cout << "3. Вывести матрицу." << endl;

cout << "4. Выход." << endl;

cout << "5. Отчистить консоль!" << endl;

cout << "=======================================" << endl;

}

/// <summary>

/// Создание матрицы

/// </summary>

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

void create(Matrix& array)

{

cout << "Введите количество строк: ";

cin >> array.row;

cout << "Введите количество столбцов: ";

cin >> array.column;

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;

}

/// <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() % 100 + (-50);

}

}

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>

/// Сума положительных елементов левой части матрицы

/// </summary>

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

/// <returns>Возвращает суму положительных чисел левой части матрицы</returns>

int leftMatrix(Matrix& array)

{

int sum\_left = 0;

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

{

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

{

if (array.arr[i][j] > 0)

sum\_left += array.arr[i][j];

}

}

return sum\_left;

}

/// <summary>

/// Сума отрицательных елементов првой части матрицы

/// </summary>

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

/// <returns>Возвращает суму отрицательных чисел правой части матрицы</returns>

int rightMatrix(Matrix& array)

{

int sum\_right = 0;

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

{

if (array.column % 2 != 0)

{

for (int j = array.column - 1; j > array.column / 2; j--)

{

if (array.arr[i][j] < 0)

sum\_right += array.arr[i][j];

}

}

else

{

for (int j = array.column - 1; j >= array.column / 2; j--)

{

if (array.arr[i][j] < 0)

sum\_right += array.arr[i][j];

}

}

}

return sum\_right;

}

/// <summary>

/// Сравнение сум по модулю

/// </summary>

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

/// <returns>Возвращает наибольшую суму по модулю</returns>

int processing(Matrix& array)

{

int sum1 = leftMatrix(array);

int sum2 = rightMatrix(array);

if (sum1 > abs(sum2))

{

return sum1;

}

else

{

return abs(sum2);

}

}

/// <summary>

/// 2-й уровень меню создания матрицы

/// </summary>

void outputCreateLevel2()

{

cout << "=============== МЕНЮ ================" << endl;

cout << "1. Заполнить рандомными числами." << endl;

cout << "2. Заполнить вручную." << endl;

cout << "3. Заполнить с файла." << endl;

cout << "4. Вернутся в главное меню." << endl;

cout << "5. Отчистить консоль." << endl;

cout << "=====================================" << endl;

cout << "Введите цифру необходимого пункта меню!!!" << endl;

}

/// <summary>

/// 2-й уровень меню вывода матрицы

/// </summary>

void outputMatrixLevel2()

{

cout << "=============== МЕНЮ ================" << endl;

cout << "1. Вывести матрицу в консоль." << endl;

cout << "2. Записать матрицу в файл." << endl;

cout << "3. Вернутся в главное меню." << endl;

cout << "4. Отчистить консоль." << endl;

cout << "=====================================" << endl;

cout << "Введите цифру необходимого пункта меню!!!" << endl;

}

/// <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 матрицы." << endl;

cin >> string;

fstream f(string);

if (f.fail())

{

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

}

//считывание количества строк и столбцов из файла

//и выделение памяти

f >> array.row;

f >> array.column;

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

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

{

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

}

//запись значений в матрицу

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;

}

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.close();

}

Файл func.h

#pragma once

struct Matrix

{

int row;

int column;

int\*\* arr;

};

void outputMenu();

void outputCreateLevel2();

void outputMatrixLevel2();

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);

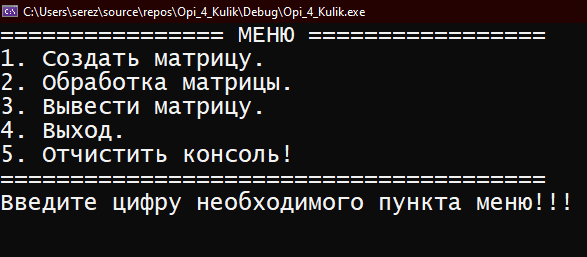
void show(Matrix& array);

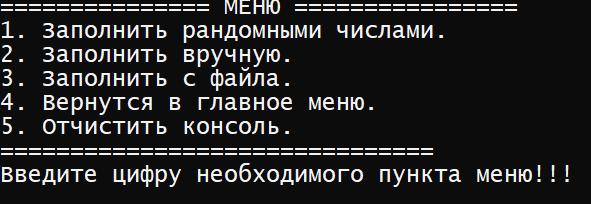
int leftMatrix(Matrix& array);

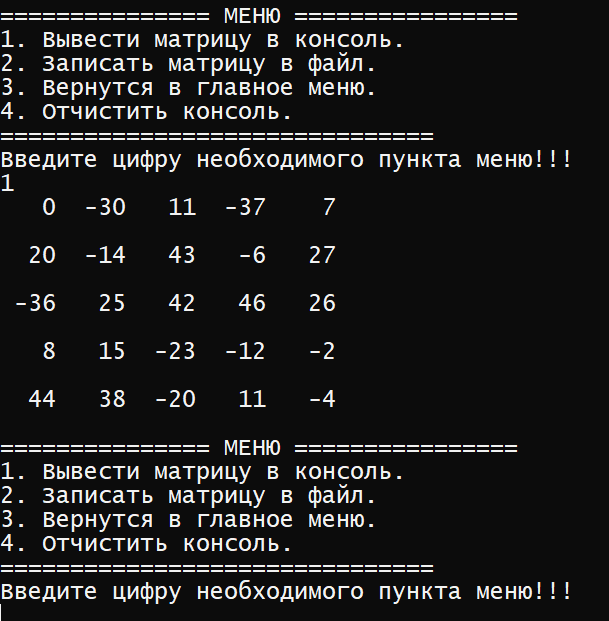
int rightMatrix(Matrix& array);

int processing(Matrix& array);

**3)Аналіз результатів**



****



**4) Висновок**

Під час виконання лабораторної роботи розробляв меню. Меню представляє собобою список функцій, які може виконувати програма, користувач повинен мати доступ, до будь-якого пункту меню в будь який час. Меню може бути багаторівневим. На мій погляд, меню зручна річ, проте в консолі воно не таке зручне, як у віконній програмі, адже там не треба вводити необхідні цифри для переходу до іншохо розділу, а використовується мишка.