**Міністерство Освіти і Науки України**

**Київський Національний Університет імені Тараса Шевченка**

**Факультет Інформаційних Технологій**

Звіт з лабораторної роботи № 10

з дисципліни **Основи програмування**

Виконав студент групи ІР-12

Федоренко Ярослав Юрійович

Викладач: ас. Пороховніченко І.А.

**Київ – 2023**

**Лабораторна робота №10**

**Варіант № 18**

**Тема:** Структури як типи даних. Робота з файлами, форматування вводу/виводу.

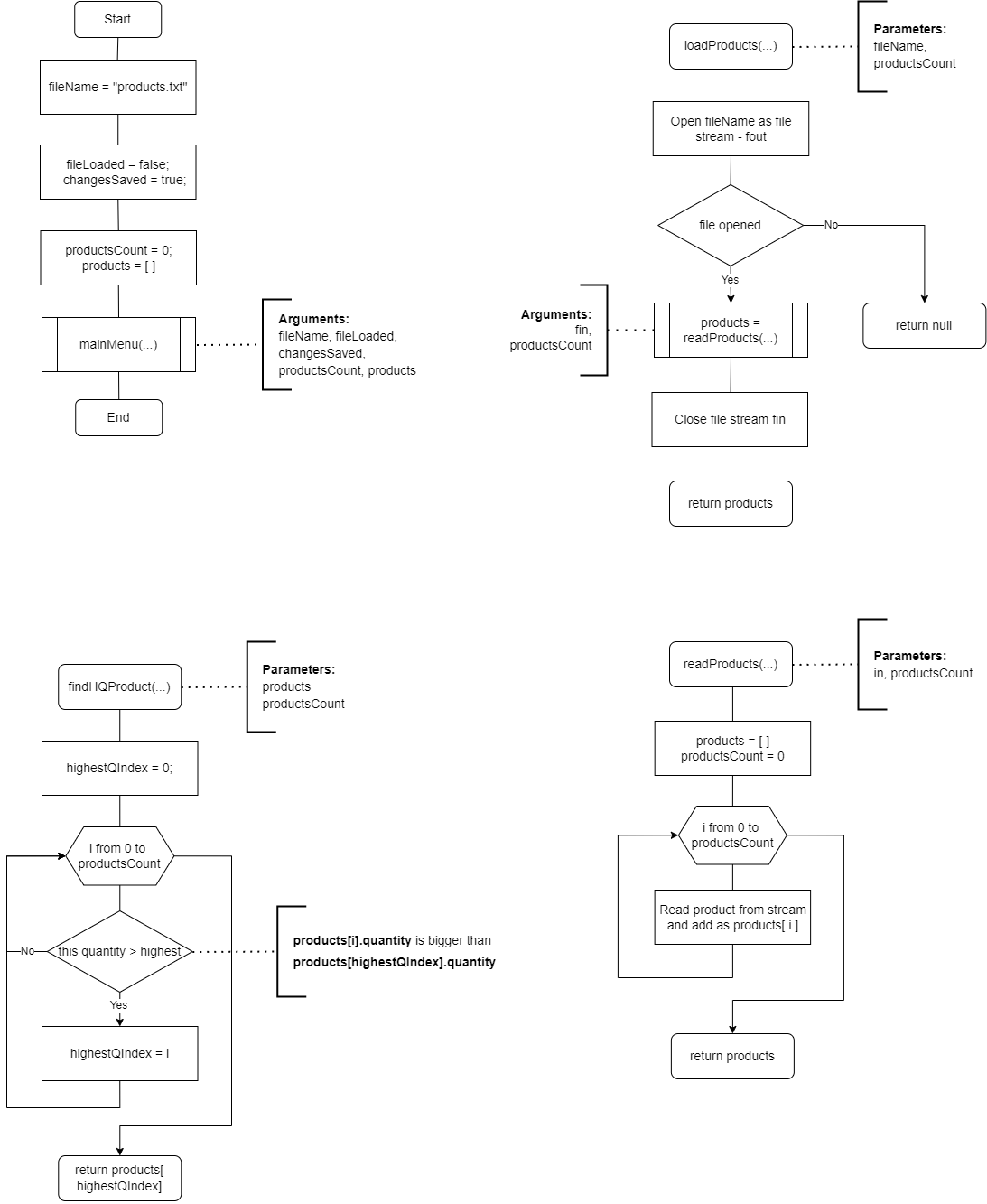
**Мета:** Опанувати програмні методи роботи з текстовими файлами. Опанувати структурний підхід організації роботи з даними.Навчитися розробляти програмні застосунки для обробки текстових файлів із застосуванням структурного підходу. Опанувати конструктивні особливості роботи з строковими типами даних та текстовими файлами, а також синтаксис побудови структурна прикладі мов програмування високого рівня С++.

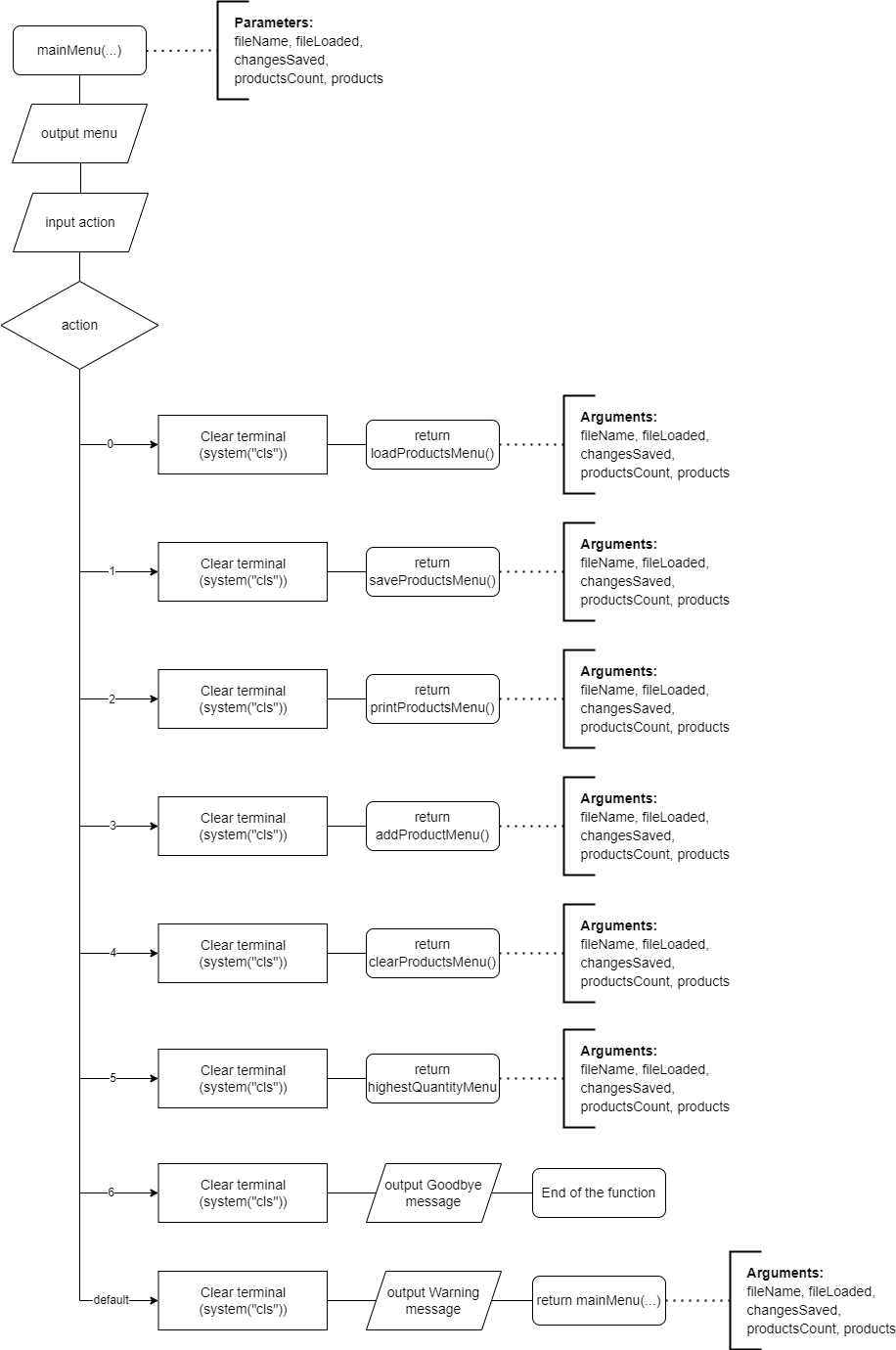
**Завдання**

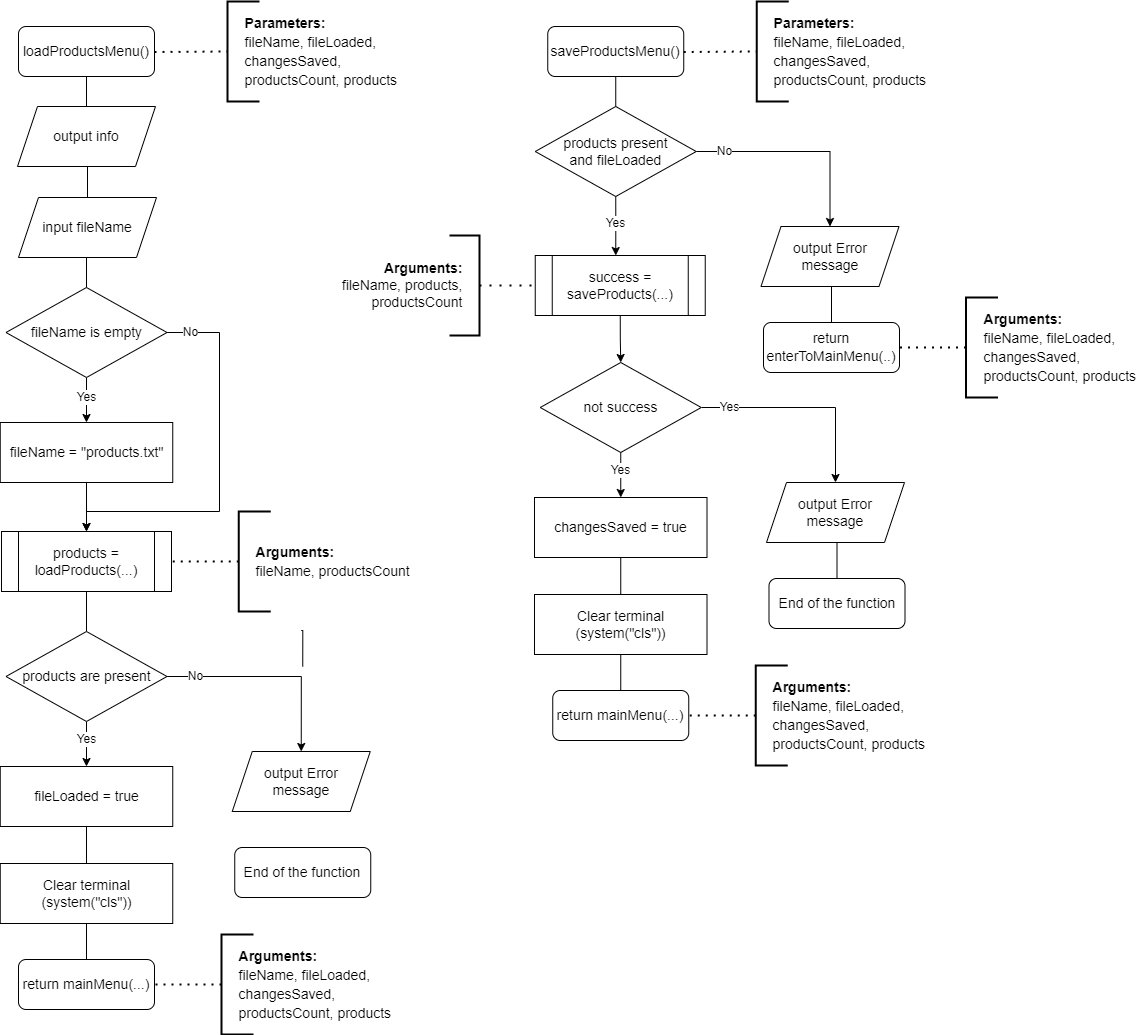
A white rectangular object with black text

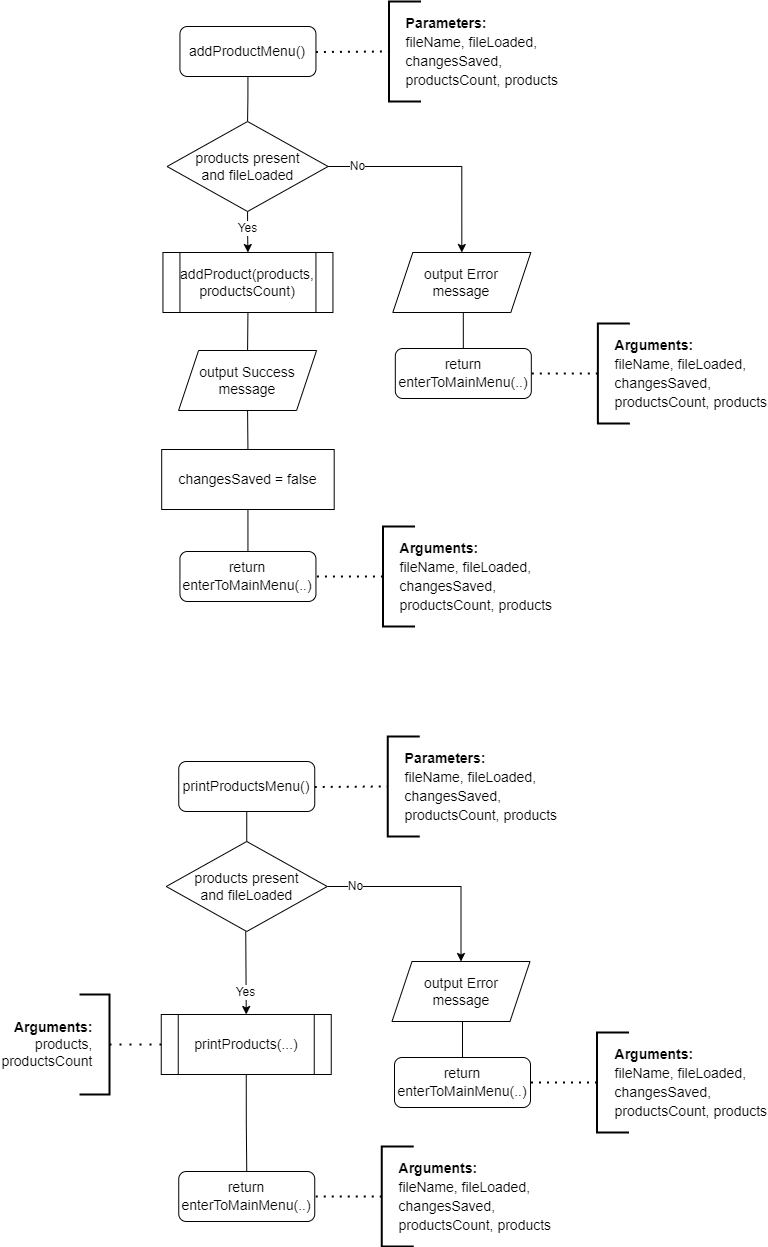
Description automatically generated

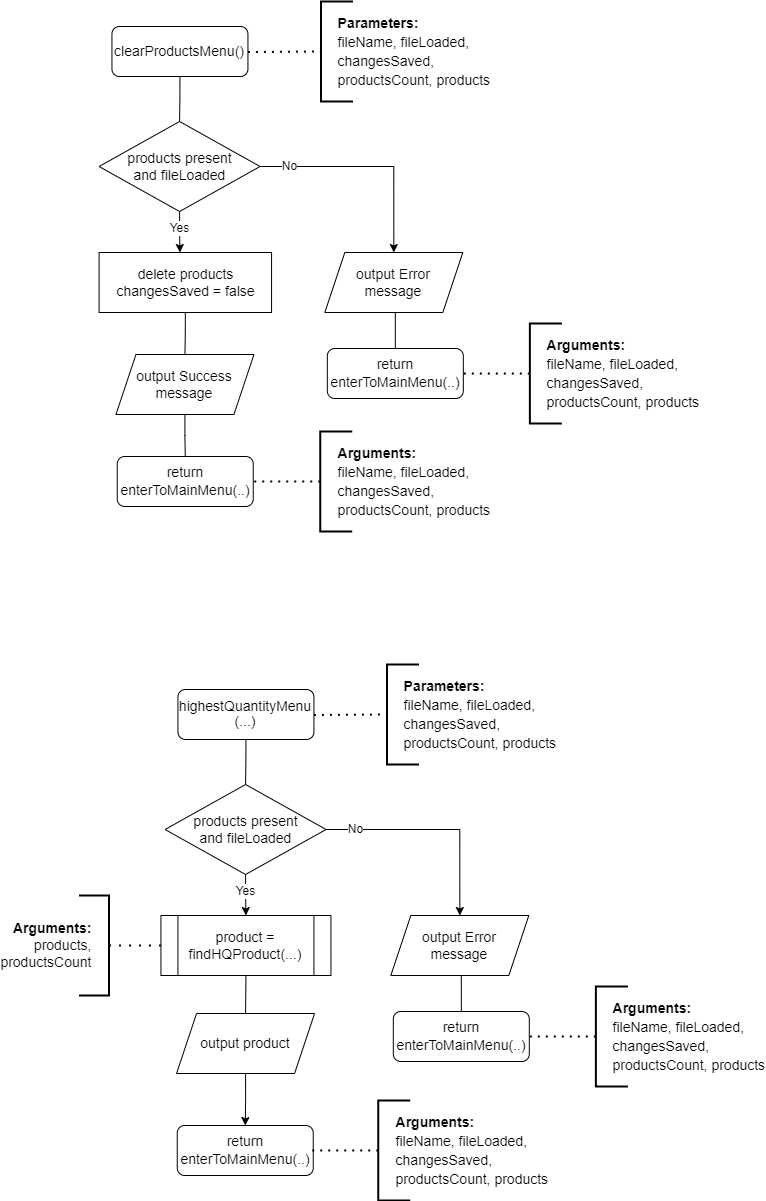
**Блок – схема**

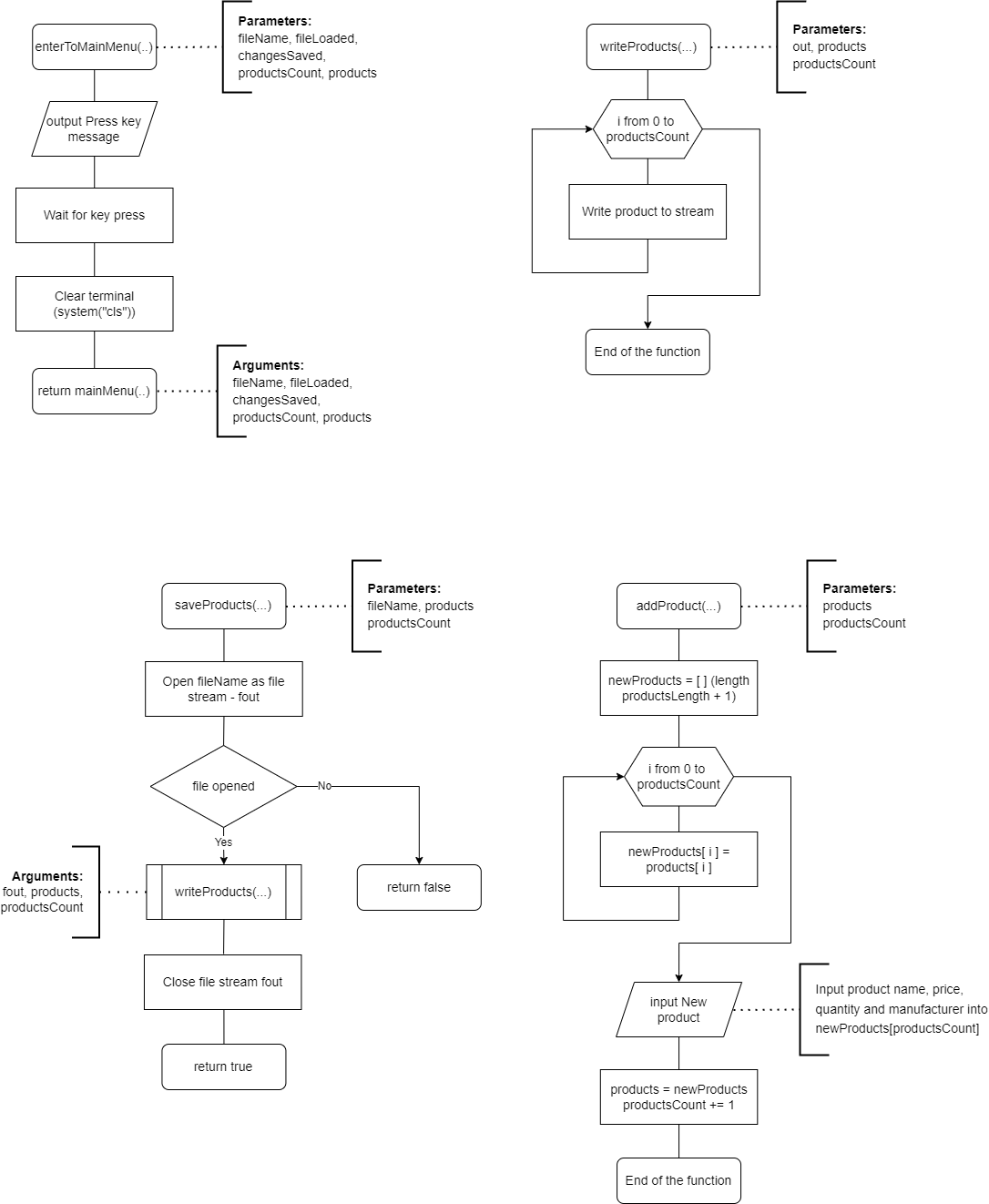
****

****

****

****

****

****

**Код**

**main.cpp**

#include <string>

#include <iostream>

#include "navigation.h"

#include "structs.h"

using namespace std;

int main()

{

string fileName = "products.txt";

int productsCount = 0;

bool fileLoaded = false;

bool changesSaved = true;

Product \* products = new Product[0];

mainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

return 0;

}

**navigation.cpp**

#include "navigation.h"

#include "utility.h"

void mainMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

cout << "╔====================[" << MAGENTA << "Головне Меню" << WHITE << "]=====================╗" << endl;

cout << "║ Поточний файл: "

<< (fileLoaded ? GREEN : GREY)

<< (fileLoaded ? fileName : "Не завантажено") << WHITE

<< (fileLoaded ? string(39 - fileName.length(), ' ') : string(25, ' ')) << "║" << endl;

cout << WHITE << "║ Зміни: "

<< (changesSaved ? GREEN : RED)

<< (changesSaved ? "✅ Збережено" : "❌ Не збережено") << WHITE

<< (changesSaved ? string(35, ' ') : string(32, ' ')) << "║" << endl;

cout << "║ ║" << endl;

cout << WHITE << "║ " << BLUE << "[0] " << WHITE << "Завантажити дані з файлу. ║" << endl;

cout << WHITE << "║ " << BLUE << "[1] " << WHITE << "Зберегти поточні дані в файл. ║" << endl;

cout << WHITE << "║ " << BLUE << "[2] " << WHITE << "Переглянути завантажені дані. ║" << endl;

cout << WHITE << "║ " << BLUE << "[3] " << WHITE << "Додати товар в поточні дані. ║" << endl;

cout << WHITE << "║ " << BLUE << "[4] " << WHITE << "Очистити дані. ║" << endl;

cout << WHITE << "║ " << BLUE << "[5] " << WHITE << "Обчислити вартість товару з найбільшою кількістю. ║" << endl;

cout << WHITE << "║ " << RED << "[6] " << WHITE << "Вихід. ║" << endl;

cout << "╚=======================================================╝" << endl;

cout << endl << "Введіть бажану дію: ";

int action;

cin >> action;

switch (action)

{

case 0:

system("cls");

return loadProductsMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 1:

system("cls");

return saveProductsMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 2:

system("cls");

return printProductsMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 3:

system("cls");

return addProductMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 4:

system("cls");

return clearProductsMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 5:

system("cls");

return highestQuantityMenu(fileName, fileLoaded, changesSaved, productsCount, products);

case 6:

system("cls");

cout << "До побачення!";

delete[] products;

return;

default:

system("cls");

cout << "Невідома дія!";

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

}

void loadProductsMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

cout << "╔=================[" << MAGENTA << "Завантаження даних" << WHITE << "]==================╗" << endl;

cout << "║ Поточний файл: "

<< (fileLoaded ? GREEN : GREY)

<< fileName << WHITE

<< string(39 - fileName.length(), ' ') << "║" << endl;

cout << WHITE << "║ Завантажено: "

<< (fileLoaded ? GREEN : RED)

<< (fileLoaded ? "✅ Так" : "❌ Ні") << WHITE

<< (fileLoaded ? string(35, ' ') : string(36, ' ')) << "║" << endl;

cout << "╚=======================================================╝" << endl;

cout << endl << "Введіть назву файлу (Enter для products.txt): ";

cin.ignore();

getline(cin, fileName);

if (fileName.empty()) {

fileName = "products.txt";

}

delete[] products;

products = loadProducts(fileName.c\_str(), productsCount);

if (!products) {

cout << RED << "Помилка завантаження даних!" << endl;

delete[] products;

return;

}

fileLoaded = true;

system("cls");

return mainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void saveProductsMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

if (!fileLoaded || !products) {

cout << RED <<"Дані відсутні, нема що зберігати!" << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

bool success = saveProducts(fileName.c\_str(), products, productsCount);

if (!success) {

cout << RED << "Помилка збереження даних!" << endl;

delete[] products;

return;

}

changesSaved = true;

return mainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void printProductsMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

if (!fileLoaded || !products) {

cout << RED << "Для перегляду даних вам спочатку необхідно завантажити їх!" << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

printProducts(products, productsCount);

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void addProductMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

if (!fileLoaded || !products) {

cout << RED << "Щоб додати продукт вам спочатку необхідно завантажити файл!" << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

addProduct(products, productsCount);

cout << endl << GREEN << "Продукт успішно додано!" << WHITE;

changesSaved = false;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void clearProductsMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

if (!fileLoaded || !products) {

cout << RED << "Щоб очистити дані вам спочатку необхідно завантажити файл!" << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

delete[] products;

productsCount = 0;

products = nullptr;

changesSaved = false;

cout << endl << GREEN << "Дані очищено." << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void highestQuantityMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

if (!fileLoaded || !products) {

cout << RED << "Щоб виконувати пошук вам спочатку необхідно завантажити файл!" << WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

Product product = findHQProduct(products, productsCount);

cout << "Товар з найбільшою кількістю (x"

<< GREY << product.quantity << WHITE << "): "

<< GREEN << product.name

<< endl;

cout << WHITE << "Його сумарна вартість: "

<< GREEN << product.quantity \* product.price

<< WHITE;

return enterToMainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

void enterToMainMenu(

string & fileName,

bool & fileLoaded,

bool & changesSaved,

int & productsCount,

Product \*& products

) {

cout << endl << endl << GREY << "Натисніть Enter, щоб повернутись в головне меню..." << WHITE;

cin.ignore();

cin.get();

system("cls");

return mainMenu(fileName, fileLoaded, changesSaved, productsCount, products);

}

**utility.cpp**

#include "utility.h"

void writeProduct(ostream & out, const Product & product)

{

out << "product: " << product.name << endl

<< "price: " << product.price << endl

<< "quantity: " << product.quantity << endl

<< "manufacturer: " << product.manufacturer << endl;

}

void writeProducts(ostream & out, Product \*& products, const int productsCount)

{

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

writeProduct(out, products[i]);

out << endl << "=========================";

if (!(i == productsCount - 1)) {

out << endl << endl;

}

}

}

bool saveProducts(const char \* filename, Product \*& products, const int productsCount)

{

ofstream fout(filename);

if (!fout.is\_open()) {

return false;

}

writeProducts(fout, products, productsCount);

fout.close();

return true;

}

void addProduct(Product \*& products, int & productsCount)

{

Product \* newProducts = new Product[productsCount + 1];

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

newProducts[i] = products[i];

}

cout << "Введіть назву продукту: ";

cin >> newProducts[productsCount].name;

cout << "Введіть ціну продукту: ";

cin >> newProducts[productsCount].price;

cout << "Введіть кількість продукту: ";

cin >> newProducts[productsCount].quantity;

cout << "Введіть виробника продукту: ";

cin >> newProducts[productsCount].manufacturer;

delete[] products;

products = newProducts;

productsCount++;

}

void printProducts(Product \*& products, const int productsCount)

{

for (int i = 0; i < 64; i++) {

std::cout << "-";

}

cout << endl << left

<< setw(3) << "|"

<< setw(20) << "Продукт"

<< setw(3) << "|"

<< setw(15) << "Ціна"

<< setw(3) << "|"

<< setw(22) << "Кількість"

<< setw(3) << "|"

<< setw(22) << "Виробник"

<< "|" << endl;

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

cout << left

<< setw(3) << "|"

<< setw(13) << products[i].name

<< setw(3) << "|"

<< setw(11) << products[i].price

<< setw(3) << "|"

<< setw(13) << products[i].quantity

<< setw(3) << "|"

<< setw(14) << products[i].manufacturer

<< "|" << endl;

}

for (int i = 0; i < 64; i++) {

std::cout << "-";

}

}

void readProduct(istream & in, Product & product)

{

string str;

in >> str;

in >> product.name;

in >> str;

in >> product.price;

in >> str;

in >> product.quantity;

in >> str;

in >> product.manufacturer;

in >> str;

}

Product \* readProducts(istream & in, int & productsCount)

{

productsCount = 0;

Product \* products = new Product[99999];

for (int i = 0; !in.eof(); i++) {

readProduct(in, products[productsCount]);

productsCount++;

}

return products;

}

Product \* loadProducts(const char \* filename, int & productsCount)

{

ifstream fin(filename);

if (!fin.is\_open()) {

return nullptr;

}

Product \* products = readProducts(fin, productsCount);

fin.close();

return products;

}

Product findHQProduct(Product \*& products, const int productsCount)

{

int highestQIndex = 0;

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

if (products[i].quantity > products[highestQIndex].quantity) {

highestQIndex = i;

}

}

return products[highestQIndex];

}

**Тестування**



**Висновок**

Завдяки цій лабораторній роботі я навчився працювати з текстовими файлами, записувати та зчитувати дані за допомогою бібліотек. Також я ознайомився зі структурами та навчися застосовувати їх на практиці, набув досвіду написання програми з мінімальним інтерфейсом.