Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет

ЗВІТ
ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 11
з навчальної дисципліни: Базові технології та Методології програмування

“ РЕАЛІЗАЦІЯ ПРОГРАМНИХ ЗАСОБІВ ОБРОБЛЕННЯ ДИНАМІЧНИХ СТРУКТУР ДАНИХ ТА БІНАРНИХ ФАЙЛІВ ”

ВИКОНАЛА
студентка академічної групи КІ-22-2
Тітарова А. А.

ПЕРЕВІРИВ
викладач кафедри кібербезпеки
та програмного забезпечення
Собінов Олександр Георгійович

Кропивницький – 2023

ТЕМА: Реалізація програмних засобів оброблення динамічних структур даних та бінарних файлів

Мета роботи полягає у набутті ґрунтовних вмінь і практичних навичок командної (колективної) реалізації програмного забезпе- чення, розроблення функцій оброблення динамічних структур даних, використання стандартних засобів C++ для керування динамічною пам'яттю та бінарними файловими потоками.

Варіант №10

Завдання:

https://github.com/odorenskyi/Tkachenko-Oleksii-KI222
https://github.com/odorenskyi/Titarova-Anastasiia-KI222
https://github.com/odorenskyi/Karpova-Yelyzaveta-KI222

Варіант 10

—— Завдання на розроблення програмного забезпечення ——

Створити **е-довідник кодів товарів згідно з Українською класифікацією товарів зовнішньоекономічної діяльності** (номер розділу, назва розділу, група/код/ товару, назва товару).

За вибором користувача застосунок забезпечує:

- *пошук* запису в е-довіднику за введеним кодом товару;
- *зберігання* е-довідника у заданий текстовий файл;
- *додавання* нового запису в е-довідник;
- *вилучення* заданого запису з е-довідника;
- *завершення* роботи програми з автоматичним записом даних у файл.

Дані е-довідника автоматично завантажуються з файлу під час запуску ПЗ.

ДОВІДНИК кодів товарів згідно з Українською класифікацією товарів зовнішньоекономічної діяльності (УКТ ЗЕД)

Розділ I. Живі тварини; продукти тваринного походження >>>

- Група 01 Живі тварини
- Група 02 М'ясо та істівні субпродукти
- Група 03 Риба і ракоподібні, молюски та інші водяні безхребетні
- Група 04 Молоко та молочні продукти; яйця птиці; натуральний мед; істівні продукти тваринного походження, в іншому місці не зазначені
- Група 05 Інші продукти тваринного походження, в іншому місці не зазначені

Розділ II. Продукти рослинного походження >>>

- Група 06. Живі дерева та інші рослини; цибулини, коріння та інші аналогічні частини рослин; зрізані квіти і декоративна зелень
- Група 07. Овочі та деякі істівні коренеплоди і бульби
- Група 08. Істівні плоди та горіхи; шкірки цитрусових або динь
- Група 09. Кава, чай, мате, або парагвайський чай, прянощі
- Група 10. Зернові культури
- Група 11. Продукція борошномельно-круп'яної промисловості; солод; крохмалі; інулін; пшенична

...

Український класифікатор товарів ЗЕД

I (з 01 по 05) Живі тварини; продукти тваринного походження

Код товару	Найменування товару
01	Живі тварини
02	М'ясо та їстівні субпродукти
03	Риба і ракоподібні, молюски та інші водяні безхребетні
04	Молоко та молочні продукти; яйця птиці; натуральний мед; їстівні продукти тваринного походження, в іншому місці не зазначені
05	Інші продукти тваринного

```
#include <iostream>
#include <fstream>
#include <vector>

struct Product {
    int sectionNumber;
    std::string sectionName;
    std::string groupCode;
    std::string productName;
};

class ProductDirectory {
private:
    std::vector<Product> products;

public:
    void loadDirectoryFromFile(const std::string& filename) {
        std::ifstream file(filename);
        if (!file) {
            std::cout << "Cannot open file: " << filename << std::endl;
            return;
        }

        products.clear();
        Product product;
        std::string line;
        while (std::getline(file, line)) {
            if (line.empty()) {
                continue;
            }
```

```
            if (line.find("Розділ") != std::string::npos) {
                product.sectionNumber =
std::stoi(line.substr(line.find_first_of("0123456789")));
                product.sectionName = line.substr(line.find_first_of(".") + 2);
            } else if (line.find("Група") != std::string::npos) {
                product.groupCode = line.substr(line.find_first_of("0123456789"));
                product.productName = line.substr(line.find_first_of(" ") + 1);
                products.push_back(product);
            }
        }

        file.close();
        std::cout << "Directory loaded from file: " << filename << std::endl;
    }

    void saveDirectoryToFile(const std::string& filename) {
        std::ofstream file(filename);
        if (!file) {
            std::cout << "Cannot open file: " << filename << std::endl;
            return;
        }

        for (const auto& product : products) {
            file << "Розділ " << product.sectionNumber << ". " <<
product.sectionName << std::endl;
            file << "Група " << product.groupCode << " " << product.productName <<
std::endl;
        }

        file.close();
```

```
        std::cout << "Directory saved to file: " << filename << std::endl;
    }

    void searchProductByCode(const std::string& productCode) {
        bool found = false;
        for (const auto& product : products) {
            if (product.groupCode == productCode) {
                std::cout << "Product found: " << "Розділ " << product.sectionNumber
<< ". " << product.sectionName << ", "
                          << "Група " << product.groupCode << " " <<
product.productName << std::endl;
                found = true;
            }
        }

        if (!found) {
            std::cout << "Product not found." << std::endl;
        }
    }

    void addProduct(const Product& newProduct) {
        products.push_back(newProduct);
        std::cout << "Product added to the directory." << std::endl;
    }

    void removeProduct(const std::string& productCode) {
        for (auto it = products.begin(); it != products.end(); ++it) {
            if (it->groupCode == productCode) {
                products.erase(it);
                std::cout << "Product removed from the directory." << std::endl;
                return;
```

```
            }
        }

        std::cout << "Product not found." << std::endl;
    }
};

int main() {
    ProductDirectory directory;
    directory.loadDirectoryFromFile("directory.txt");

    int choice;
    std::string productCode, productName;
    Product newProduct;

    do {
        std::cout << "Menu:\n";
        std::cout << "1. Search product by code\n";
        std::cout << "2. Save directory to file\n";
        std::cout << "3. Add new product to directory\n";
        std::cout << "4. Remove product from directory\n";
        std::cout << "5. Exit\n";
        std::cout << "Enter your choice: ";
        std::cin >> choice;

        switch (choice) {
            case 1:
                std::cout << "Enter product code: ";
                std::cin >> productCode;
                directory.searchProductByCode(productCode);
```

```
            break;
        case 2:
            directory.saveDirectoryToFile("directory.txt");
            break;
        case 3:
            std::cout << "Enter section number, section name, group code, and product name: ";
            std::cin >> newProduct.sectionNumber >> newProduct.sectionName >> newProduct.groupCode >> newProduct.productName;
            directory.addProduct(newProduct);
            break;
        case 4:
            std::cout << "Enter product code: ";
            std::cin >> productCode;
            directory.removeProduct(productCode);
            break;
        case 5:
            directory.saveDirectoryToFile("directory.txt");
            std::cout << "Exiting the program." << std::endl;
            break;
        default:
            std::cout << "Invalid choice. Please try again." << std::endl;
            break;
    }

    std::cout << std::endl;
  } while (choice != 5);

  return 0;
}
```

Цей код представляє клас `ProductDirectory`, який виконує функції довідника продуктів. Він має наступний набір функцій:

1. `loadDirectoryFromFile(const std::string& filename)`: Ця функція завантажує довідник з файлу з вказаним ім'ям. Вона відкриває файл, перебирає рядки файлу і зчитує дані про розділи та групи продуктів, додаючи їх до вектора `products`. Функція використовує рядки `"Розділ"` і `"Група"` для визначення розділів та груп продуктів у файлі.

2. `saveDirectoryToFile(const std::string& filename)`: Ця функція зберігає довідник у файл з вказаним ім'ям. Вона відкриває файл для запису, проходиться по всіх елементах вектора `products` і записує інформацію про розділи та групи продуктів у файл у відповідному форматі

3. `searchProductByCode(const std::string& productCode)`: Ця функція шукає продукт за кодом в довіднику. Вона перебирає всі продукти у векторі `products` і порівнює їх коди з вказаним `productCode`. Якщо збіг знайдений, виводиться інформація про знайдений продукт.

4. `addProduct(const Product& newProduct)`: Ця функція додає новий продукт до довідника. Вона приймає об'єкт типу `Product` і додає його до вектора `products`.

5. `removeProduct(const std::string& productCode)`: Ця функція видаляє продукт з довідника за його кодом. Вона перебирає всі продукти у векторі `products` і порівнює їх коди з вказаним `productCode`. Якщо збіг знайдений, продукт видаляється з вектора.

Головна функція `main` використовує об'єкт `ProductDirectory` для взаємодії з довідником продуктів. Вона надає меню користувачу, де він може вибрати різні опції, такі як пошук продукту за кодом, збереження довідника у файл, додавання нового продукту до довідника та його видалення. Програма повторює виконання операцій до тих пір, поки користувач не вибере опцію "Вихід".

ВІДПОВІДІ НА КОНТРОЛЬНІ ЗАПИТАННЯ:

1. Згідно з міжнародним стандартом ISO/IEC 12207, комплексування (інтегрування) ПЗ відноситься до процесу, який об'єднує різні компоненти програмного забезпечення (модулі, підсистеми, системи) в одну цілісну систему. Цей процес включає кроки, що передбачають поєднання окремих компонентів, встановлення взаємозв'язків між ними, тестування та верифікацію інтегрованої системи для забезпечення її працездатності та відповідності вимогам.

2. Вказівник та посилання є двома різними концепціями в мові програмування C++. Вказівник - це змінна, яка зберігає адресу пам'яті іншого об'єкта. Вказівники можуть бути переприсвоєні та змінювати адресу, яку

вони зберігають. Посилання - це псевдонім або альтернативне ім'я існуючого об'єкта. Посилання завжди посилаються на той самий об'єкт, на який були посилання визначені.

3. Допустимі операції над вказівниками в мові програмування C++ включають:

- Оператори присвоєння (`=`) та порівняння (`==`, `!=`, `<`, `>`, `<=`, `>=`).

- Арифметичні операції, такі як додавання (`+`), віднімання (`-`), інкремент (`++`), декремент (`--`), вказівникове зміщення (`+`, `-`).

- Операції роботи з пам'яттю, такі як використання `new` та `delete` для виділення та звільнення пам'яті.

- Оператори доступу до членів (`->`, `.`) для роботи з полями та методами об'єктів, на які вказує вказівник.

4. Операція опосередкованої адресації

(`*`) у мові програмування C++ використовується для отримання значення, на яке вказує вказівник. Синтаксис запису цієї операції полягає у розміщенні зірочки перед вказівником. Наприклад, якщо `ptr` є вказівником на об'єкт, то `*ptr` поверне значення цього об'єкта.

5. Функції-члени об'єктів `fstream` в мові програмування C++, які забезпечують роботу з потоками файлів, включають:

- `open`: відкриває файл з вказаним ім'ям і режимом.

- `is_open`: перевіряє, чи відкритий файловий потік.

- `close`: закриває файловий потік.

- `write`: записує дані у файловий потік.

- `read`: читає дані з файлового потоку.

- `seekg` і `seekp`: переміщають позицію зчитування (`get`) та запису (`put`) вказівника у файловому потоці.

- `tellg` і `tellp`: повертають поточну позицію зчитування (`get`) та запису (`put`) у файловому потоці.

6. Члени (поля) елемента динамічної структури, які підлягають зберіганню у файл, залежать від призначення цієї структури та потреб програми. Зазвичай у файл зберігаються ті поля, які містять важливі дані, що потрібні для відновлення структури. Наприклад, якщо динамічна структура представляє об'єкт користувача, то поля, які містять ім'я, прізвище, адресу тощо, можуть бути збережені у файл. У той же час, тимчасові або внутрішні поля, які не мають значення поза межами програми, можуть бути виключені з збереження.

7. Бінарний файловий потік відрізняється від текстового з погляду читання/запису інформації. В бінарних файлових потоках дані записуються і читаються у бінарному форматі, без будь-якої обробки текстового представлення. Це означає, що дані зберігаються безпосередньо у вигляді послідовності байтів, що забезпечує ефективність та точність збереження. З іншого боку, текстовий файловий потік використовує текстове представлення для запису та читання даних. В цьому випадку дані перетворюються у рядки символів, що дозволяє легше читати та редагувати файли у текстовому форматі, але може призводити до деякої втрати точності та займати більше місця у порівнянні з бінарним форматом.