Міністерство освіти і науки України
Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення
Дисципліна: Об’єктно-орієнтоване програмування

Лабораторна робота №7
Тема: «КОНТЕЙНЕРНІ КЛАСИ. СТАНДАРТНА БІБЛІОТЕКА ШАБЛОНІВ (STL) В С++»

Виконав: ст. гр. КН-24
Куріщенко П. В.
Перевірив: асистент
Козірова Н. Л.

Кропивницький 2025

Мета: ознайомитись Контейнерні класи та навчитись їх програмно реалізовувати мовою C++.

Варіант – 14 - 10 = 4

Завдання:

Розробіть **систему управління складом магазину автозапчастин**, використовуючи послідовний контейнер std::vector для зберігання інформації про запчастини.

Кожна запчастина повинна містити:

- Назву,
- Виробника,
- Ціну,
- Кількість на складі.

Реалізуйте:

- Додавання кількох запчастин до списку.
- Виведення списку запчастин.
- Видалення однієї або кількох запчастин за назвою.
- Оновлення ціни та кількості для певної запчастини.
- Виведення оновленого списку запчастин.

Додатково:

Реалізуйте метод сортування запчастин за назвою або ціною. Поясніть, чому обрали саме цей тип контейнера.

Результат:

```
* All parts *
Name: Brake Pads, manufacturer: Brembo, price: 1250, amount: 40
Name: Oil Filter, manufacturer: Bosch, price: 350, amount: 100
Name: Spark Plug, manufacturer: NGK, price: 270, amount: 80
Name: Battery, manufacturer: Varta, price: 4200, amount: 15
Name: Winter Tires, manufacturer: Michelin, price: 6900, amount: 25
Name: Summer Tires, manufacturer: Continental, price: 6400, amount: 30

Enter part name to remove: Summer Tires
Enter part name to update: Winter Tires
Enter new part price: 8000
Enter new part amount: 50
Sort by name/price?: price

Updated:
* All parts *
Name: Spark Plug, manufacturer: NGK, price: 270, amount: 80
Name: Oil Filter, manufacturer: Bosch, price: 350, amount: 100
Name: Brake Pads, manufacturer: Brembo, price: 1250, amount: 40
Name: Battery, manufacturer: Varta, price: 4200, amount: 15
Name: Winter Tires, manufacturer: Michelin, price: 8000, amount: 50
```

Лістинг Part.h:

```cpp
#ifndef PART_H
#define PART_H

#include <string>

using namespace std;

struct Part {
    string name;
    string manufacturer;
    double price = 0.0;
    unsigned short amount = 0;
};

#endif // PART_H
```

Лістинг inventory.h:

```cpp
#ifndef INVENTORY_H
#define INVENTORY_H

#include <vector>
#include <string>
#include <iostream>
#include "Part.h"

using namespace std;

class Inventory
{
    vector<Part> parts;
```

```
public:
    Inventory();
    Inventory(const initializer_list<Part>&);

    void addPart(const Part&);
    void showAllParts() const;
    void removeByName(const string&);
    void updateByName(const string&, double, unsigned short);
    void sortBy(const string&);
};

#endif // INVENTORY_H
```

Лістинг inventory.cpp:

```
#include "inventory.h"

Inventory::Inventory() {}

Inventory::Inventory(const initializer_list<Part>& list) {
    for (const auto& part : list) parts.push_back(part);
}

void Inventory::addPart(const Part& part) { parts.push_back(part); }

void Inventory::showAllParts() const{
    if (parts.empty()) {
        cout << "Inventory is empty(" << endl;
        return;
    }

    cout << "* All parts *" << endl;
    for (const auto& part : parts)
        cout << "Name: " << part.name
             << ", manufacturer: " << part.manufacturer
             << ", price: " << part.price
             << ", amount: " << part.amount << endl;
    cout << endl;
}

void Inventory::removeByName(const string& name) {
    for (auto iterator = parts.begin(); iterator != parts.end(); ) {
        if (iterator->name == name) iterator = parts.erase(iterator);
        else ++iterator;
    }
}

void Inventory::updateByName(const string& name, double newPrice, unsigned
short newAmount) {
    bool found = false;

    for (auto& part : parts)
        if (part.name == name) {
            part.price = newPrice;
            part.amount = newAmount;
            found = true;
            break;
        }
```

```
    if (!found) cout << "Part \"" << name << "\" not found!" << endl;
}

bool compareByName(const Part& left, const Part& right) { return left.name < right.name; }

bool compareByPrice(const Part& left, const Part& right) { return left.price < right.price; }

void Inventory::sortBy(const string& input) {
    if (input == "name") sort(parts.begin(), parts.end(), compareByName);
    else if (input == "price") sort(parts.begin(), parts.end(), compareByPrice);
    else cout << "Should enter name/price" << endl;
}
```

Лістинг main.cpp:

```
#include "inventory.h"

int main()
{
    string input;
    double price = 0.0;
    unsigned short amount = 0;
    Inventory store = {
        {"Brake Pads", "Brembo", 1250.0, 40},
        {"Oil Filter", "Bosch", 350.0, 100},
        {"Spark Plug", "NGK", 270.0, 80},
        {"Battery", "Varta", 4200.0, 15},
        {"Winter Tires", "Michelin", 6900.0, 25},
        {"Summer Tires", "Continental", 6400.0, 30}
    };

    store.showAllParts();

    cout << "Enter part name to remove: ";
    getline(cin, input);
    store.removeByName(input);

    cout << "Enter part name to update: ";
    getline(cin, input);
    cout << "Enter new part price: ";
    cin >> price;
    cout << "Enter new part amount: ";
    cin >> amount;
    cin.ignore();
    store.updateByName(input, price, amount);

    cout << "Sort by name/price?: ";
    cin >> input;
    store.sortBy(input);

    cout << "\nUpdated: " << endl;
    store.showAllParts();

    return 0;
}
```