**Лабораторна робота №12. STL**

Тема: STL. Ітератори. Послідовні контейнери. Цикл range-for. Асоціативні контейнери.

Мета: Отримати базові знання про STL контейнери. Освоїти основні механізми роботи з STL контейнерами.

**ВИМОГИ**

**1.1 Інформація про розробника:**

* Кліщов Б. Р.
* КІТ 102.8а
  1. **Загальне завдання**

Маючи класи з прикладної області РГЗ (тільки базовий клас та клас/класи спадкоємці), створити діалогове меню, що дозволяє продемонструвати роботу STL контейнерів (додавання / видалення / отримання даних, показ всіх елементів) та показати їх принципову різницю:

- vector;

- set;

- list;

- map .

При цьому врахувати, що контейнери містять елементи одного типу, наприклад, базового. Прохід по всьому контейнеру повинен виконуватися за допомогою циклу мови С++11 – range-for

**1.3 Додаткові умови виконання завдання:**

- продемонструвати відсутність витоків пам’яті;

- продемонструвати роботу розроблених методів за допомогою модульних тестів;

- не використовувати конструкцію «using namespace std;» , замість цього слід роботи «using» кожного необхідного класу: using std::string, using std::cout.

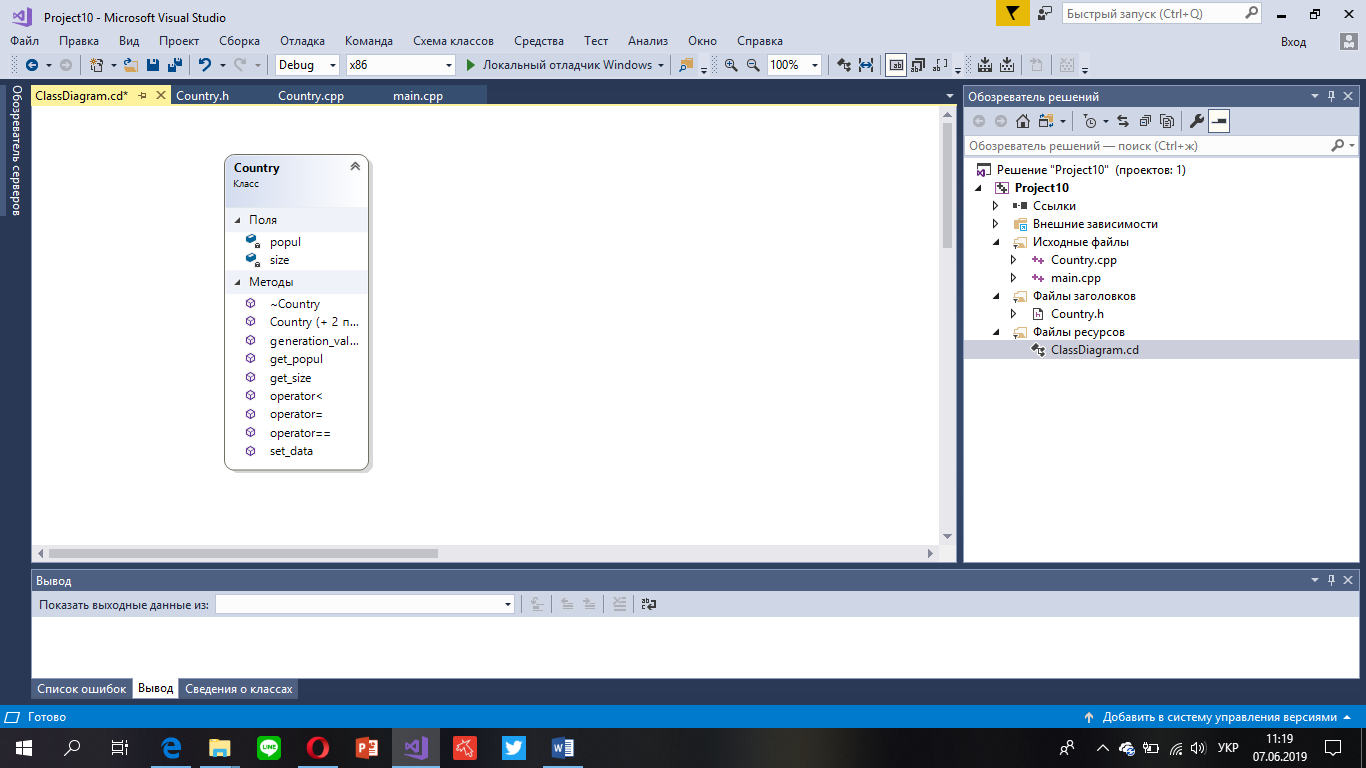
1. **ОПИС ПРОГРАМИ**

**2.1 Функціональне призначення**

Програма призначена щоб отримувати та зберігати інформацію щодо різних країн світу.

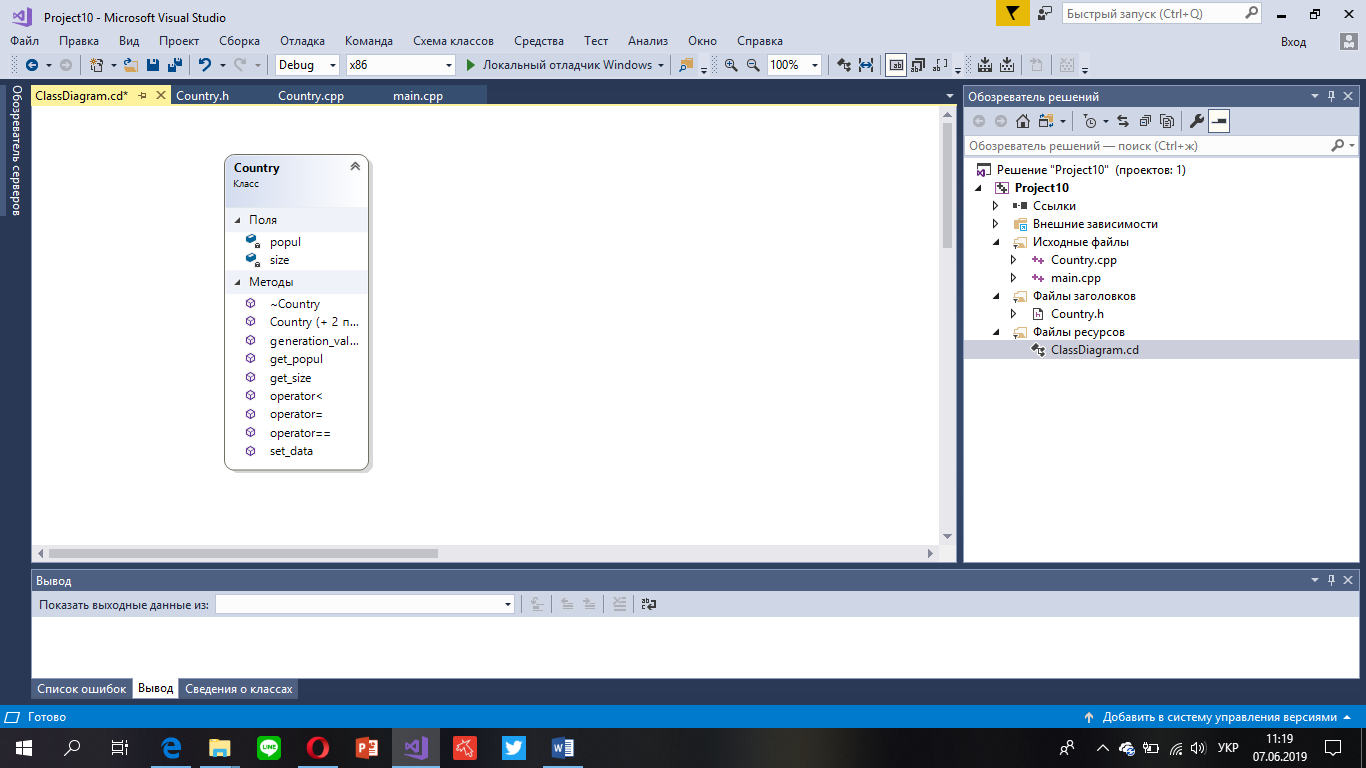
* 1. **Опис логічної структури**

На рисунку № 1 зображена діаграма класу



Малюнок №1. Діаграма класу

На рисунку № 2 зоображена структура програми:



Малюнок №2: Cтруктура програми

**2.3 Важливі фрагменти програми**

Код програми :

|  |
| --- |
| #include <iostream> |
|  | #include <vector> |
|  | #include <set> |
|  | #include <map> |
|  | #include <list> |
|  | #include <iterator> |
|  | #include <algorithm> |
|  | #include "Country.h" |
|  |  |
|  | using namespace std; |
|  |  |
|  | void Vector() { |
|  | vector<Country> myVector; |
|  | int choose, id; |
|  | Country country; |
|  | string name; |
|  | cout << "Vector" << endl; |
|  | do { |
|  | cout << "0 - Exit" << endl; |
|  | cout << "1 - Print vector" << endl; |
|  | cout << "2 - Find element by index" << endl; |
|  | cout << "3 - Add element" << endl; |
|  | cout << "4 - Delete element" << endl; |
|  | cout << "Choose: "; |
|  | cin >> choose; |
|  |  |
|  | switch (choose) |
|  | { |
|  | case 1: |
|  | for (int i = 0; i < myVector.size(); i++) { |
|  | cout << myVector[i] << endl; |
|  | } |
|  | break; |
|  | case 2: |
|  | cout << "input index: "; |
|  | cin >> id; |
|  | cout << myVector.at(id); |
|  | break; |
|  | case 3: { |
|  | cout << "Please, enter the name of country: "; |
|  | cin >> name; |
|  | cout << endl; |
|  | country.setData(name); |
|  | myVector.push\_back(country); |
|  | break; |
|  | } |
|  | case 4: |
|  | myVector.clear(); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } while (choose != 0); |
|  | } |
|  |  |
|  | void Set() { |
|  | int choose, id; |
|  | set<Country> mySet; |
|  | Country country; |
|  | string name; |
|  | cout << "Set" << endl; |
|  |  |
|  | do { |
|  | cout << "0 - Exit" << endl; |
|  | cout << "1 - Print" << endl; |
|  | cout << "2 - Add element" << endl; |
|  | cout << "3 - Delete element" << endl; |
|  | cout << "Choose: "; |
|  | cin >> choose; |
|  |  |
|  | switch (choose) |
|  | { |
|  | case 1: |
|  | for (auto &it : mySet) {//öèêë for\_each |
|  | cout << it << endl; |
|  | } |
|  | break; |
|  | case 2: { |
|  | cout << "Please, enter the name of country: "; |
|  | cin >> name; |
|  | cout << endl; |
|  | country.setData(name); |
|  | mySet.insert(country); |
|  | break; |
|  | } |
|  | case 3: |
|  | mySet.erase(country); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } while (choose != 0); |
|  | } |
|  |  |
|  | void List() { |
|  | int choose, id; |
|  | list<Country> myList; |
|  | Country country; |
|  | string name; |
|  | cout << "List" << endl; |
|  | do { |
|  | cout << "0 - Exit" << endl; |
|  | cout << "1 - Print" << endl; |
|  | cout << "2 - Find element by index" << endl; |
|  | cout << "3 - Add element" << endl; |
|  | cout << "4 - Delete element" << endl; |
|  | cout << "Choose: "; |
|  | cin >> choose; |
|  |  |
|  | auto it = myList.begin(); |
|  |  |
|  | switch (choose) |
|  | { |
|  | case 1: |
|  | for (auto i = myList.cbegin(); i != myList.cend(); i++) { |
|  | cout << \*i; |
|  | } |
|  | break; |
|  | case 2: |
|  | cout << "input index: "; |
|  | cin >> id; |
|  |  |
|  | advance(it, id); |
|  |  |
|  | cout << id; |
|  | break; |
|  | case 3: { |
|  | it = myList.begin(); |
|  | cout << "Please, enter the name of country: "; |
|  | cin >> name; |
|  | cout << endl; |
|  | country.setData(name); |
|  | myList.insert(it, country); |
|  |  |
|  | break; |
|  | } |
|  | case 4: |
|  | cout << "input delete point: "; |
|  | cin >> id; |
|  | it = myList.begin(); |
|  |  |
|  | advance(it, id); |
|  | myList.erase(it); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } while (choose != 0); |
|  | } |
|  |  |
|  | void Map() { |
|  | int choose, id; |
|  | map<int, Country> myMap; |
|  | Country country; |
|  | string name; |
|  | cout << "Map" << endl; |
|  | do { |
|  | cout << "0 - Exit" << endl; |
|  | cout << "1 - Print" << endl; |
|  | cout << "2 - Find element by index" << endl; |
|  | cout << "3 - Add element" << endl; |
|  | cout << "4 - Delete element" << endl; |
|  | cout << "Choose: "; |
|  | cin >> choose; |
|  |  |
|  | auto it = myMap.begin(); |
|  | int k; |
|  |  |
|  | switch (choose) |
|  | { |
|  | case 1: |
|  | for (const auto& el : myMap) { |
|  | std::cout << el.second << endl; |
|  | } |
|  | break; |
|  | case 2: |
|  | cout << "Input index: "; |
|  | cin >> id; |
|  | cout << myMap.at(id); |
|  | break; |
|  | case 3: { |
|  | cout << "Please, enter key: "; |
|  | cin >> k; |
|  | cout << endl; |
|  | cout << "Please, enter the name of country: "; |
|  | cin >> name; |
|  | cout << endl; |
|  | country.setData(name); |
|  | myMap.emplace(k, country); |
|  | break; |
|  | } |
|  | case 4: |
|  | cout << "input delete point: "; |
|  | cin >> id; |
|  | myMap.erase(id); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } while (choose != 0); |
|  | } |
|  |  |
|  | int main() { |
|  | int choose; |
|  |  |
|  | do { |
|  | cout << "0 - Exit" << endl; |
|  | cout << "1 - Work with VECTOR" << endl; |
|  | cout << "2 - Work with SET" << endl; |
|  | cout << "3 - Work with LIST" << endl; |
|  | cout << "4 - Work with MAP" << endl; |
|  | cout << "Choose: "; |
|  | cin >> choose; |
|  |  |
|  | switch (choose) |
|  | { |
|  | case 1: |
|  | Vector(); |
|  | break; |
|  | case 2: |
|  | Set(); |
|  | break; |
|  | case 3: { |
|  | List(); |
|  | break; |
|  | } |
|  | case 4: |
|  | Map(); |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | } while (choose != 0); |
|  |  |
|  |  |
|  | return 0; |
|  | } |

1. **ВАРІАНТИ ВИКОРИСТАННЯ**

**3.1 Результат роботи функцій**

На рисунку № 3 зоображено результат робрти програми

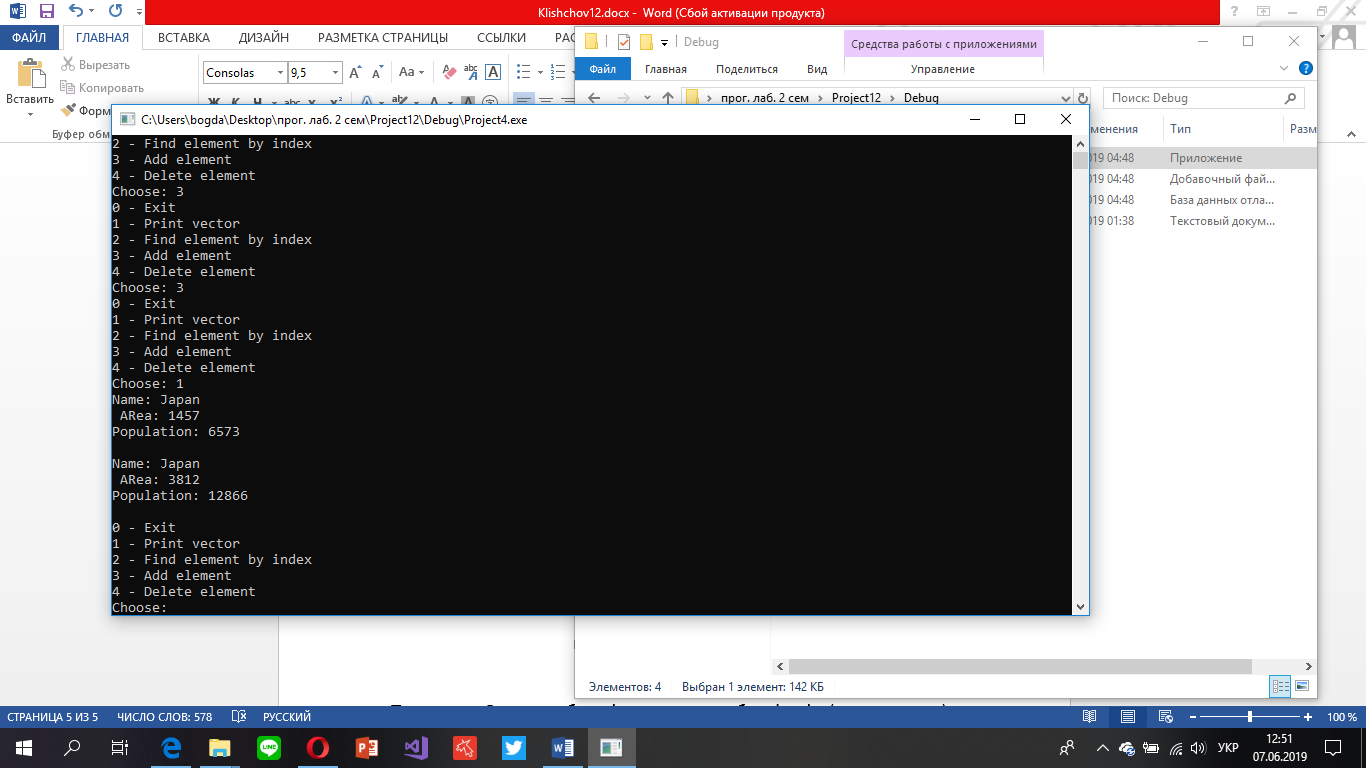


Рисунок № 3. Результат роботи програми

**Висновок:** Отримав базові знання про STL контейнери. Освоїв основні механізми роботи з STL контейнерами.