**Лабораторна робота №13. Алгоритми переміщення та пошуку**

Тема: STL. Алгоритми переміщення та пошуку.

Мета: На практиці порівняти STL алгоритми, що не модифікують послідовність.

**ВИМОГИ**

**1.1 Інформація про розробника:**

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

Поширити попередню лабораторну роботу, додаючи наступні можливості діалогового меню:

- вивід всіх елементів масиву за допомогою STL функції for\_each;

- визначення кількості елементів за заданим критерієм;

- пошук елементу за заданим критерієм.

**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.

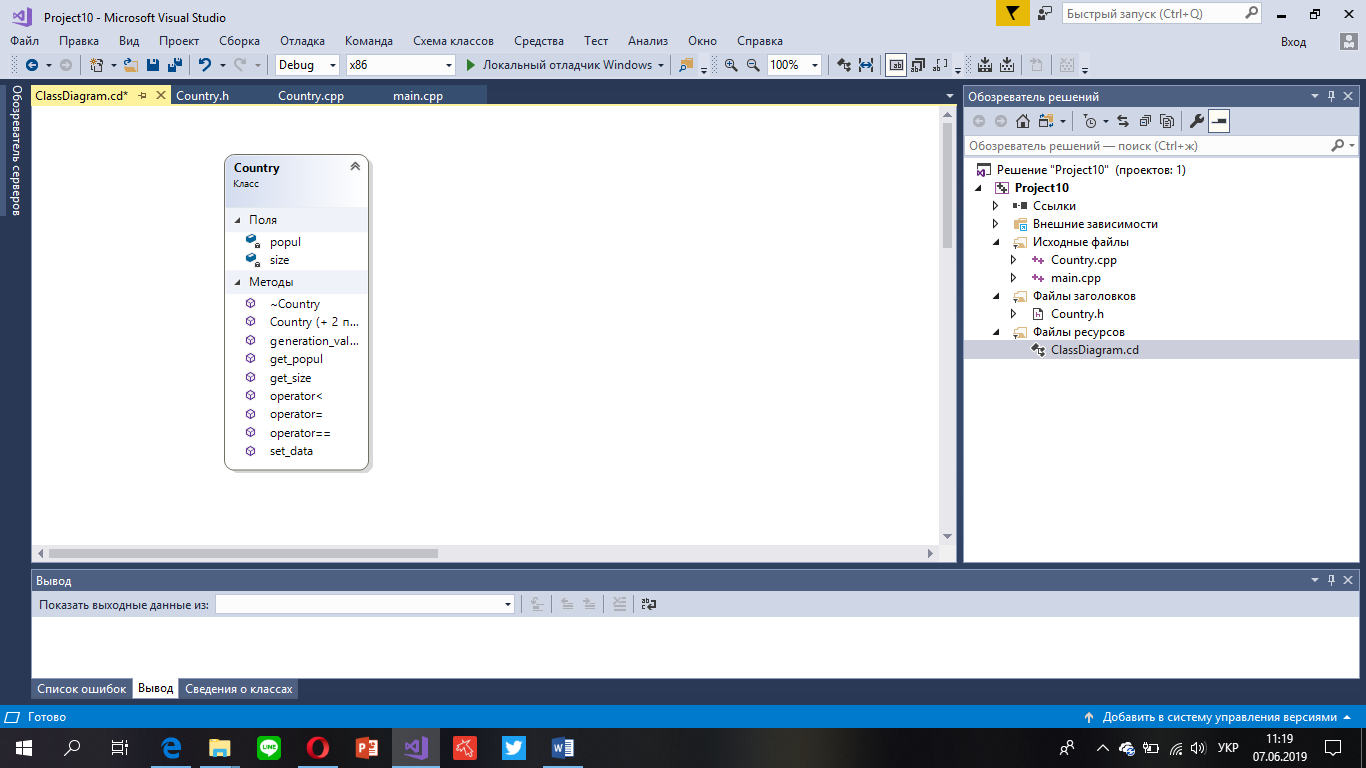
1. **ОПИС ПРОГРАМИ**

**2.1 Функціональне призначення**

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

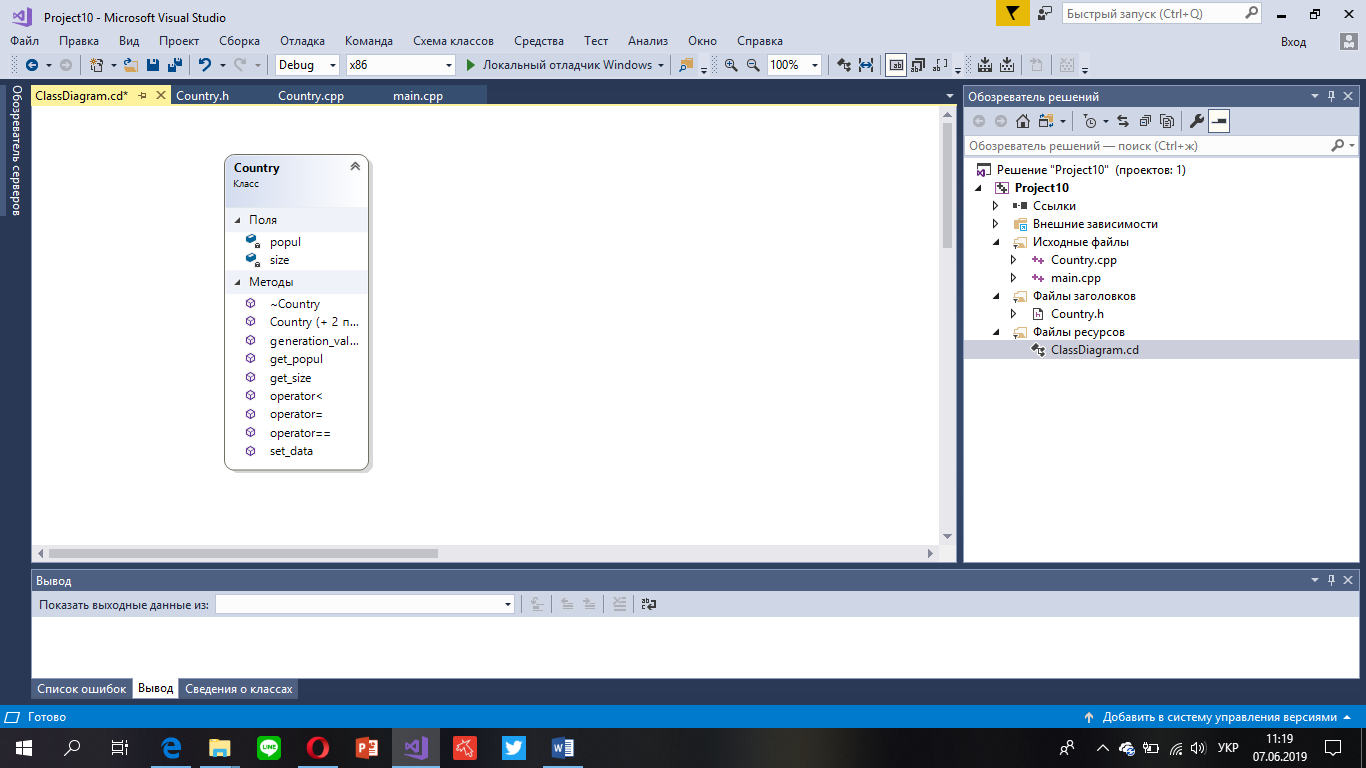
* 1. **Опис логічної структури**

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 :

int main() {

vector<Country> myVector;

int choose, id;

Country country;

std::set<Country> mySet;

std::list<Country> myList;

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: {

country.setData("Japan");

myVector.push\_back(country);

break;

}

case 4:

myVector.clear();

break;

default:

break;

}

} while (choose != 0);

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 << "\t";

}

break;

case 2: {

country.setData("Japan");

mySet.insert(country);

break;

}

case 3:

mySet.erase(country);

break;

default:

break;

}

} while (choose != 0);

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;

switch (choose)

{

case 1:

for (auto i = myList.cbegin(); i != myList.cend(); i++) {

cout << \*i;

}

break;

case 2:

cout << "input index: ";

cin >> id;

auto it = myList.begin();

advance(it, id);

cout << id;

break;

case 3: {

auto it = myList.begin();

cout << "Input index: ";

cin >> id;

advance(it, id);

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);

return 0;

}

1. **ВАРІАНТИ ВИКОРИСТАННЯ**

**3.1 Результат роботи функцій**

На рисунку № 3 зоображено результат робрти програми

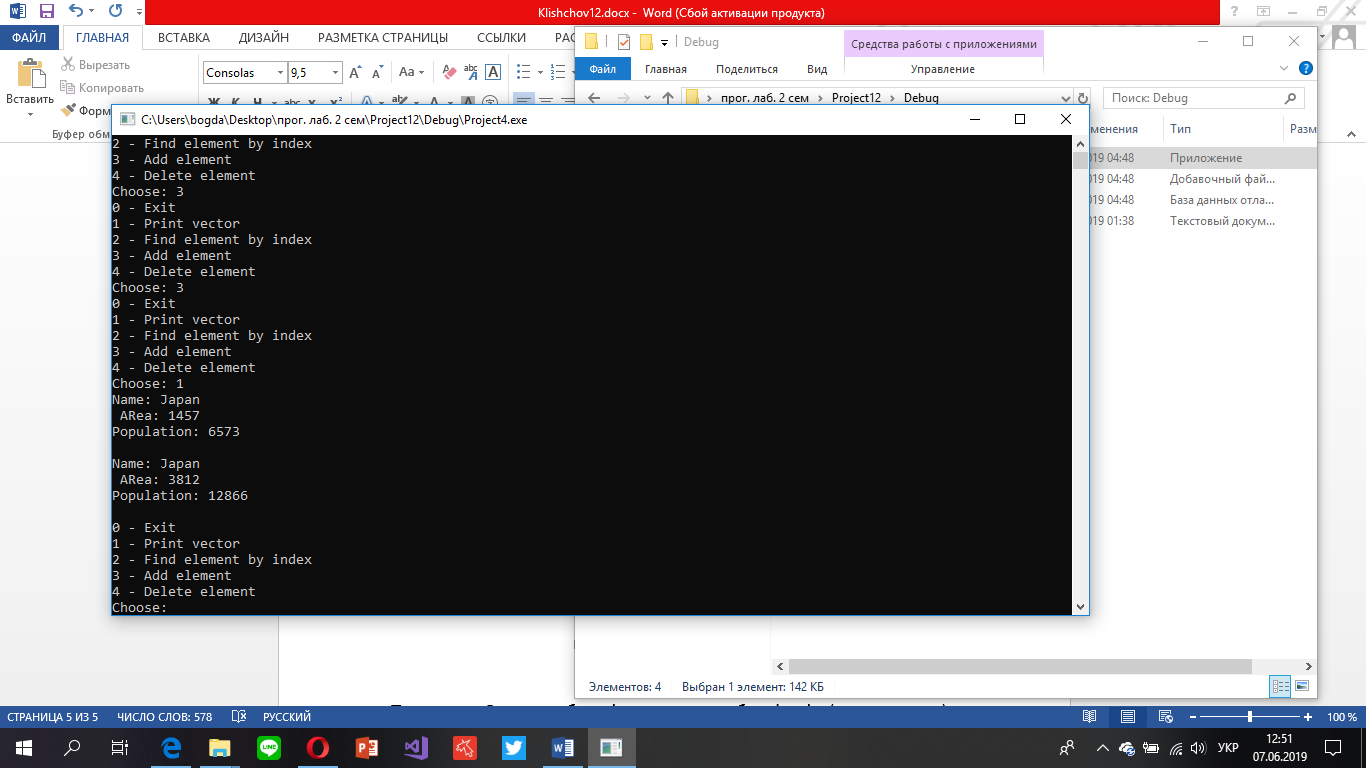


Рисунок № 3. Результат роботи програми

**Висновок:** На практиці порівняти STL алгоритми, що не модифікують послідовність.