**Лабораторна робота №2. Перевантаження методів**

Мета: Отримати базові знання про класи, конструктори та деструктори. Дослідити механізм створення та видалення об’єктів.

**ВИМОГИ**

**1.1 Інформація про розробника:**

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

Поширити попередню лабораторну роботу наступним чином:

 в базовому класі необхідно додати:

 мінімум одне поле типу char\*;

 конструктор за замовчуванням, копіювання та конструктор з аргументами;

 деструктор;

 в клас-список потрібно додати метод обходу масиву для виконання індивідуального завдання.

Приклад сигнатури такого методу:

CPhone& findCheapestPhone(float diagonal);

В наведеному прикладі реалізоване завдання пошуку самого дешевого телефону з заданою діагоналлю (повертається один телефон). **Додаткові умови виконання завдання:**

 реалізація конструкторів повинна бути продемонстрована за допомогою списків ініціалізацій;  конструктори та деструктори повинні мати логіруючі повідомлення.

17

Студент повинен продемонструвати виклик деструктора та кожного типу конструктора, а також пояснити, коли вони викликаються;

 продемонструвати відсутність витоків пам’яті;

 продемонструвати роботу розроблених методів класу-списку за допомогою модульних тестів.

**Індивідуальні завдання:**

В табл. 2.1 оберіть завдання для обходу колекції по варіанту у відповідності до номера у журналі групи.

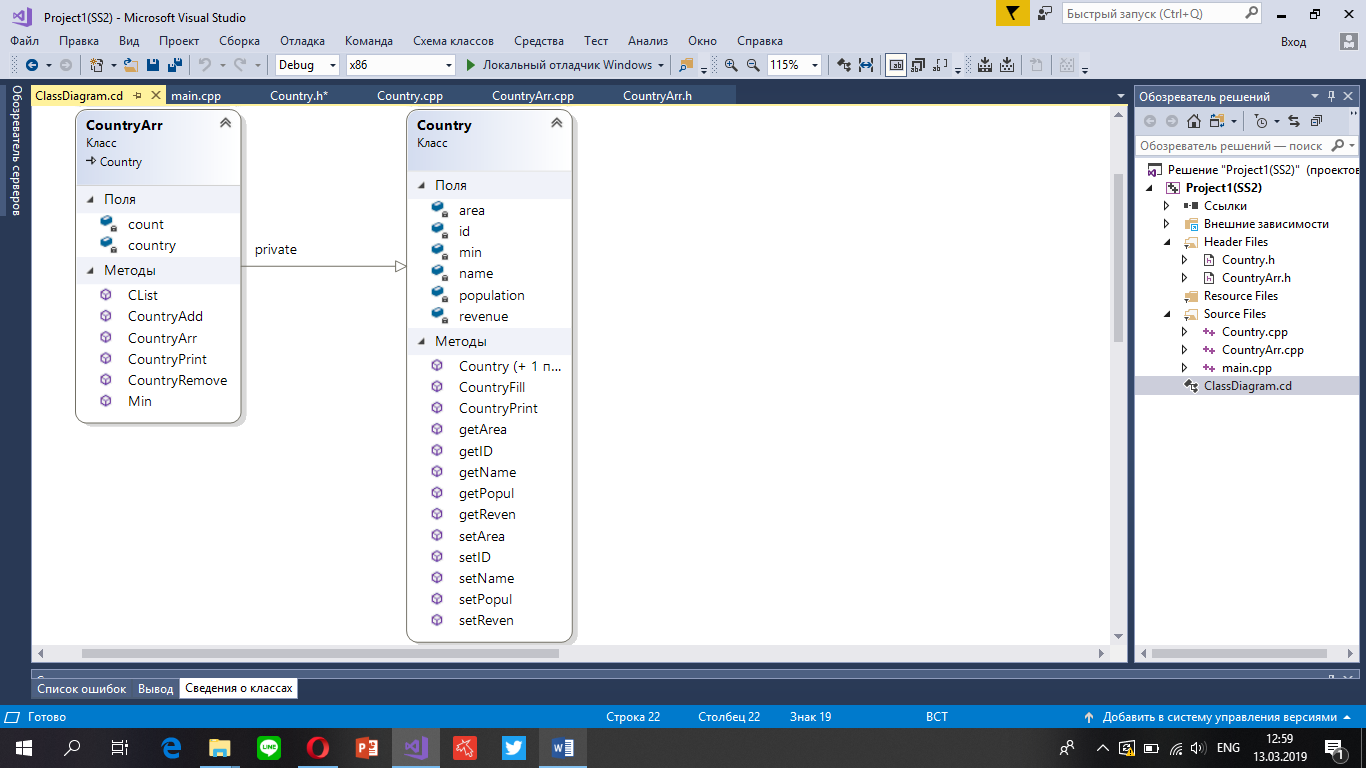
1. **ОПИС ПРОГРАМИ**

**2.1 Функціональне призначення**

Програма призначена щоб отримувати інформацію щодо різних країн світу.

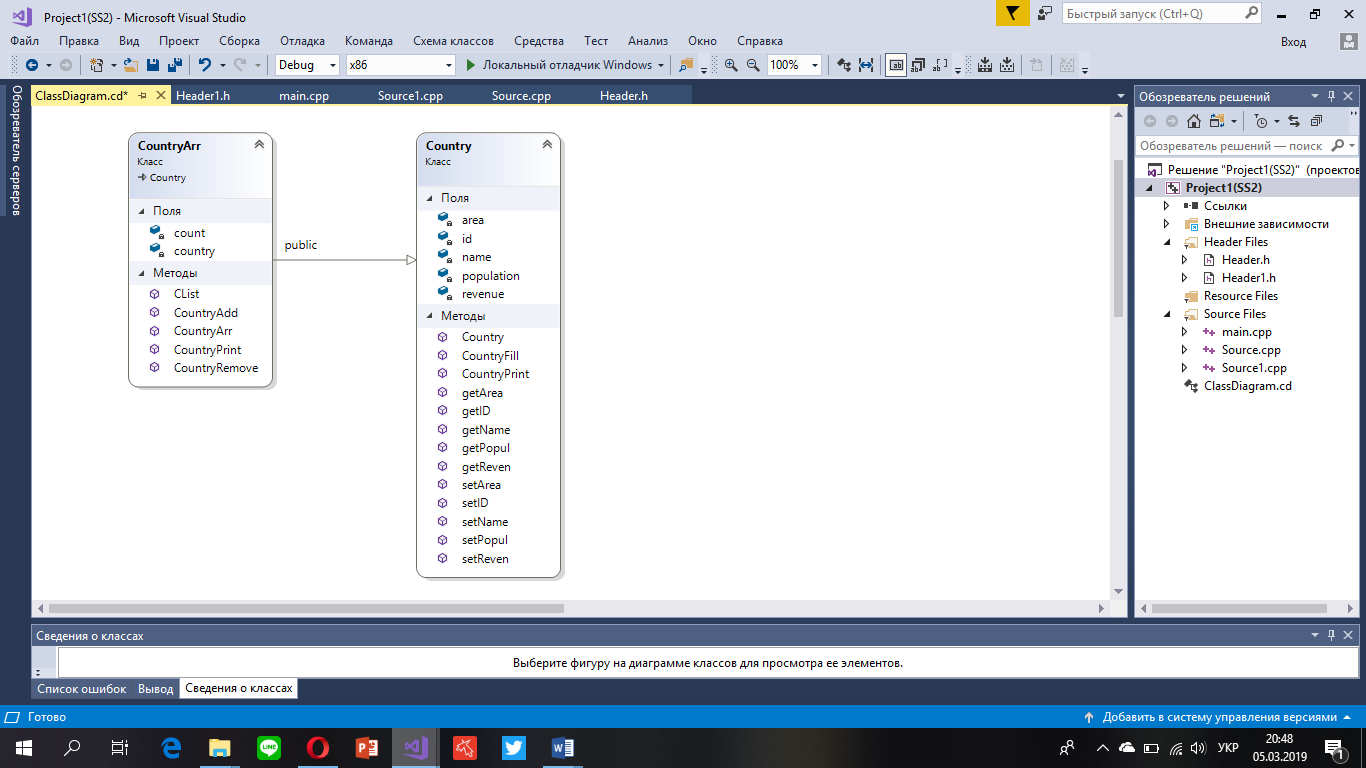
* 1. **Опис логічної структури**

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:

Main:

/\*\*

\* @mainpage

\* <b> Лабораторна робота № 2. <br/> Класи </b>

\* <br/><b><i>Мета роботи:</i>

\* </b>Отримати базові знання про класи. Дослідити механізм інкапсуляції. <br/>

\* <b><i>Загальне завдання:</i></b>

\* Клас, що відображає сутність «базового класу». При цьому, в

\* даному класі повинно бути мінімум три числових поля (бажано, щоб одне з

\* цих полів було унікальним ідентифікатором об’єкту);

\* Клас, що має в собі динамічний масив об’єктів базового класу та

\* має в собі методи додавання, видалення елементу, отримання елементу по

\* індексу (або ідентифікатору), вивід усіх елементів на екран. Рекомендовані

\* сигнатури методів:

\* -Додавання;

\* -Видалення;

\* -Отримання по індексу;

\* -Вивід усіх елементів;

\* <b><i>Індивідуальне завдання:</i></b>

\* В табл. 1.2 обрати прикладну галузь по

\* варіанту у відповідності до номера у журналі групи.

\* <br/>

\* <b><i> Додаткові умови виконання завдання: </i></b>

\* − звіт має бути виконаний згідно з вимогами до оформлення робіт;

\* − проект має складатися мінімум з трьох файлів;

\* − продемонструвати відсутність витоків пам’яті;

\* <br/>

\*

\* @author Klishchov B.

\* @date 07-02-2019

\* @version 1.2

\*/

#include"Header.h"

#include"Header1.h"

int main()

{

Country C1(555, (char\*)"Poland", 21, 160, 190);

CountryArr C2;

C2.CountryAdd(C1);

C1.CountryFill(111, (char\*)"USA", 19, 152, 127);

C2.CountryAdd(C1);

C1.CountryFill(222, (char\*)"Japan", 18, 140, 146);

C2.CountryAdd(C1);

C1.CountryFill(123, (char\*)"Greese", 15, 195, 193);

C2.CountryAdd(C1);

C2.CountryPrint();

printf("\n");

C2.CountryRemove(0);

Country minimum = C2.Min();

C2.CountryPrint();

system("pause");

return 0;}

Country:

#include"Country.h"

Country::Country() {}

Country::Country(int Id, char Name[20], int popul, int area1, int reven){

id = Id;

strcpy\_s(name, Name);

population = popul;

area = area1;

revenue = reven;

strcpy\_s(name, Name);

}

void Country::setID(int ID) {

id = ID;

}

int Country::getID() {

return id;

}

void Country::setPopul(int popul) {

population = popul;

}

int Country::getPopul() {

return population;

}

void Country::setArea(int area1) {

area = area1;

}

int Country::getArea() {

return area;

}

void Country::setReven(int reven) {

revenue = reven;

}

int Country::getReven() {

return revenue;

}

void Country::setName(char Name[20]) {

strcpy\_s(name, Name);

}

char \*Country::getName() {

return (char\*)name;

}

void Country::CountryPrint() const

{

printf("%i\t%s\t%i\t%i\t%i\n", id, name, population, area, revenue);

}

void Country::CountryFill(int Id, char Name[20], int popul, int area1, int reven) {

id = Id;

strcpy\_s(name, Name);

population = popul;

area = area1;

revenue = reven;

strcpy\_s(name, Name);

}

Sourse1:

#include"CountryArr.h"

CountryArr::CountryArr() {

count = 0;

}

void CountryArr::CountryAdd(Country countr) {

if (count == 0) {

country = new Country;

country[count] = countr;

count++;

}

else {

count++;

int j = 0;

Country \*tmp = (Country\*)operator new (sizeof(Country)\*count);

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

new(&tmp[i])Country();

}

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

tmp[i] = country[j];

j++;

}

tmp[count] = countr;

for (int i = count + 1; i < count; i++) {

tmp[j] = country[i];

}

delete[] country;

country = tmp;

}

}

void CountryArr::CountryPrint() const {

for (int i = 0; i < count; i++)

CList(i).CountryPrint();

}

void CountryArr::CountryRemove(int id) {

count--;

Country \*deltmp = (Country\*)operator new(sizeof(Country)\*count);

int j = 0;

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

if (i == id) {

j++;

}

deltmp[i] = country[j];

j++;

}

delete[] country;

country = deltmp;

}

Country & CountryArr::CList(int id) const {

return country[id];

}

Country & CountryArr::Min() {

int min = country[0].getPopul;

int index = 0;

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

if (min < country[i].getPopul) {

min = country[i].getPopul;

index = i;

}

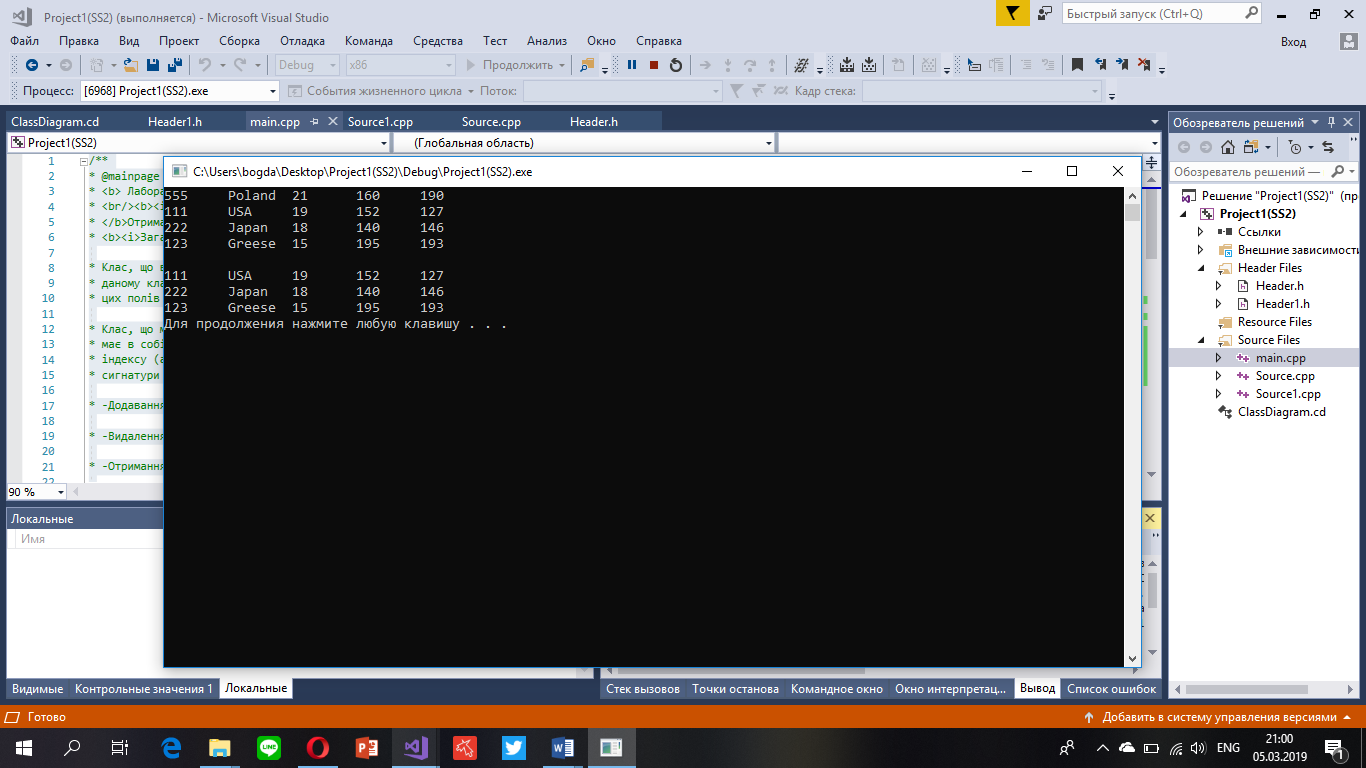
}

return country[index];

}

1. }**ВАРІАНТИ ВИКОРИСТАННЯ**

На рисунку № 3 зоображено результат роботи програми.



Висновок: Я придбав практичні навички щодо розроблення програм із використанням структурованих типів даних.