Звіт

Автор: Капелька Я.І. КІТ-119а Дата: 5 травня 2020

Лабораторна робота №1. Класи

Тема. Класи та специфікатори доступу. Інкапсуляція. Константи. **Мета**: отримати базові знання про класи. Дослідити механізм інкапсуляції.

1. Завдання до роботи **Індивідуальне завдання**:

Для предметної галузі розробити два класи:

- 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'єкта);
- клас, що має у собі динамічний масив об'єктів базового класу та має в собі методи додавання, видалення елемента, отримання елемента по індексу (або ідентифікатору), вивід усіх елементів на екран.

Прикладна галузь: Світ

Базовий клас: Країна

2. Опис класів, змінних, методів та функцій

2.1 Опис класів

Базовий клас: CCountry

Клас, що має в собі масив базового класу та методи для роботи з ним: CMetod

2.2 Опис змінних

int number_of_cities – поле класу CCountry(кількість міст в країні).

int population – поле класу CCountry (популяція кріаїни).

int area – поле класу CCountry (площа країни).

int unical_index – поле класу CCountry (унікальний індекс).

int next_i – поле класу CMetod (номер наступної країни у директорії).

int new_i – поле класу CMetod (індекс наступної країни у директорії).

CCountry * countries – поле класу CMetod (масив елементів класу CCountry).

CCountry * copy – поле класу CMetod (показчик на клас CCountry, використовується для правильної роботи деяких методів).

2.3 Опис методів

int getNumber_of_cities () const – отримання значення поля number_of_cities змінної класу CCountry (метод класу CCountry).

int getPopulation () const – отримання значення поля population змінної класу CCountry (метод класу CCountry).

int getArea () const – отримання значення поля area змінної класу CCountry (метод класу CCountry).

int getUnical_index () const – отримання значення поля uncial_index змінної класу CCountry (метод класу CCountry).

void setNumber_of_cities (const int & Number_of_cities) – зміна значення поля number_of_cities змінної класу CCountry (метод класу CCountry).

void setPopulation (const int & Population) – зміна значення поля population змінної класу CCountry (метод класу CCountry).

void setArea (const int & Area) – зміна значення поля area змінної класу CCountry (метод класу CCountry).

void setUnical_index (const int & Unical_index) – зміна значення поля unical_index змінної класу CCountry (метод класу CCountry).

void add_el (const file & CCountry) – додавання об’єкту класу CCountry до масиву в класі dir (метод класу dir).

void remove_el (const int &index) – видалення об’єкту класу CCountry з масиву в класі CMetod (метод класу CMetod).

void del_all() – видалення усіх об’єктів класу CCountry з масиву в класі CMetod (метод класу CMetod).

file find_to_index (const int& index) const – отримання об’єкту класу по індексу (повертає індекс об’єкта) CCountry з масиву в класі CMetod (метод класу CMetod).

void get_to_Screen (const int &index) const – виведення об’єкту класу CCountry з масиву в класі CMetod на екран (метод класу CMetod) .

void print_all() const – виведення усіх об’єктів класу CCountry з масиву в класі CCountry на екран (метод класу CCountry).

2.4 Опис функцій

void menu() – функція меню.

3. Текст програми

Лабораторная №1.cpp

```
#pragma once
// Лабораторная работа 1.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается
выполнение программы.
//

#include <iostream>
#include "menu.h"
#define _CRTDBG_MAP_ALLOC

int main()
{
    menu();
    if (_CrtDumpMemoryLeaks())
    {
        std::cout << "Утечка памяти обнаружена." << "\n";
    }
    else
    {
        std::cout << "Утечка памяти не обнаружена." << "\n";
    }
    }
```

CCountry.h

```
#pragma once
#include <iostream>
class CCountry
{
private:
    int number_of_cities;
    int population;
    int area;
    int unical_index;
public:
    int getNumber_of_cities() const;
    int getPopulation() const;
    int getArea() const;
    int getUnical_index() const;
    void setNumber_of_cities(const int &Number_of_cities);
    void setPopulation(const int &Population);
    void setArea(const int &Area);
    void setUnical_index(const int& Unical_index);
                                                    };
```

CCountry.cpp

```
#include "CCountry.h"
int CCountry::getNumber_of_cities() const { return number_of_cities; }
int CCountry::getPopulation() const { return  population; }
int CCountry::getArea() const { return area; }
int CCountry::getUnical_index() const { return unical_index;}
void CCountry::setNumber_of_cities(const int& Number_of_cities) {number_of_cities = Number_of_cities; }
void CCountry::setPopulation(const int& Population) { population = Population; }
void CCountry::setArea(const int& Area) { area = Area; }
    void CCountry::setUnical_index(const int& Unical_index) { unical_index = Unical_index; }
```

CMetod.h

```h
#include "CCountry.h"
class CMetod
{
private:
    CCountry* countries;
    CCountry* copy;
    int next_i = 0;
    int new_i = 1;
public:
    void add_el(const CCountry& CCountry);
    void remove_el(const int& index);
    void del_all();
    void get_to_Screen(const int& index) const;
    CCountry find_to_index(const int& index) const;
    void print_all() const;
    };
```

CMetod.cpp

```cpp
#include "CMetod.h"
void CMetod::add_el(const CCountry& Country)
{
	if (next_i == 0)
	{
		countries = (CCountry*)malloc(sizeof(CCountry));
		countries[next_i] = Country;
		next_i++;
	}
	else
	{
		copy = (CCountry*)malloc(sizeof(CCountry) * (next_i + 1));
		for (int i = 0; i < next_i; i++)
		{
			copy[i] = countries[i];
		}
		free(countries);
		countries = (CCountry*)malloc(sizeof(CCountry) * (next_i + 1));
		for (int i = 0; i < next_i; i++)
		{
			countries[i] = copy[i];
		}
		free(copy);
		countries[next_i] = Country;
		next_i++;
	}
}
void CMetod::remove_el(const int& index)
{
	if (next_i == 1)
	{
		free(countries);
		next_i--;
	}
	else
	{
		copy = (CCountry*)malloc(sizeof(CCountry) * (next_i - 1));
		for (int i = 0; i < index; i++)
		{
			copy[i] = countries[i];
		}
		for (int i = index + 1; i < next_i; i++)
		{
			copy[i - 1] = countries[i];
		}
		free(countries);
		countries = (CCountry*)malloc(sizeof(CCountry) * (next_i - 1));
		for (int i = 0; i < next_i; i++)
		{
```

```
                countries[i] = copy[i];
            }
            free(copy);
            next_i--;
        }
}
void CMetod::del_all()
{
        free(countries);
        next_i = 0;
}
void CMetod::get_to_Screen(const int& index) const
{
        std::cout << "Number_of_cities " << "Population " << "Area      " << "Unical_index " << "\n";
        std::cout << countries[index].getNumber_of_cities() << "               " <<
countries[index].getPopulation() << "    " << countries[index].getArea() << "  " <<
countries[index].getUnical_index() <<"\n";
}
CCountry CMetod::find_to_index(const int& index) const
{
        for (int i = 0; i < next_i; i++)
        {
            if (countries[i].getUnical_index() == index)
            {
                return countries[i];
            }
        }
}
void CMetod::print_all() const
{
        for (int i = 0; i < next_i; i++)
        {
            get_to_Screen(i);
        }
    }
```

menu.h

```
#pragma once
#include "CMetod.h"


void menu();
```

menu.cpp

```
#include "menu.h"
void menu()
{
    setlocale(LC_ALL, "Russian");
    int n = 0, temp_i;
    CMetod dir;
    CCountry firstcountry;
    firstcountry.setNumber_of_cities(143);
    firstcountry.setPopulation(32686854);
    firstcountry.setArea(50338346);
    firstcountry.setUnical_index(1);
    dir.add_el(firstcountry);
    firstcountry.setNumber_of_cities(156);
    firstcountry.setPopulation(65745656);
    firstcountry.setArea(56765888);
    firstcountry.setUnical_index(2);
    dir.add_el(firstcountry);
    firstcountry.setNumber_of_cities(587);
    firstcountry.setPopulation(87534345);
    firstcountry.setArea(13254756);
    firstcountry.setUnical_index(3);
    dir.add_el(firstcountry);
    firstcountry.setNumber_of_cities(146);
```

```
    firstcountry.setPopulation(38352463);
    firstcountry.setArea(15683556);
    firstcountry.setUnical_index(4);
    dir.add_el(firstcountry);
    while (n != 6)
    {
        std::cout << "_-_-_-Выберите желаемую опцию:_-_-_-_-_-_-_-_-_-" << "\n";
        std::cout << "_-1 - добавить элемент в список.-_-_-_-_-_-_-_-_" << "\n";
        std::cout << "_-2 - получить элемент в списке по индексу.-_-" << "\n";
        std::cout << "_-3 - удалить элемент из списка.-_-_-_-_-_-_-_-_" << "\n";
        std::cout << "_-4 - показать все элементы списка.-_-_-_-_-_-" << "\n";
        std::cout << "_-5 - завершить работу программы.-_-_-_-_-_-_-" << "\n";
        std::cin >> n;
        if (n == 1)
        {
            firstcountry.setNumber_of_cities(123);
            firstcountry.setPopulation(64336789);
            firstcountry.setArea(15678078);
            firstcountry.setUnical_index(5);
            dir.add_el(firstcountry);
            (firstcountry);
            std::cout << "Страна добавлена." << "\n";
        }
        else if (n == 2)
        {
            std::cout << "Введите индекс нового элемента: ";
            std::cin >> temp_i;
            dir.find_to_index(temp_i);
        }
        else if (n == 3)
        {
            std::cout << "Введите номер удаляемого элемента (нумерация начинаеться с 1): ";
            std::cin >> temp_i;
            dir.remove_el(temp_i - 1);
            std::cout << "Страна удалена";
        }
        else if (n == 4)
        {
            dir.print_all();
        }
        else
            break;
    }
    dir.del_all();
    }
```

tests.cpp

```
#pragma once
#include <iostream>
#include "menu.h"
#define _CRTDBG_MAP_ALLOC

int main()
{
       setlocale(LC_ALL, "Russian");
       CCountry firstcountry;
       firstcountry.setNumber_of_cities(100);
       firstcountry.setPopulation(20000000);
       firstcountry.setArea(32000000);
       firstcountry.setUnical_index(19);
       if ((firstcountry.getNumber_of_cities() == 100) && (firstcountry.getPopulation() == 20000000) &&
(firstcountry.getArea() == 32000000) && (firstcountry.getUnical_index() == 19))
       {
              std::cout << "Первый тест на работу геттеров и сеттеров базового класса пройден успешно."
<< "\n";
       }
       else
```

```
	{
		std::cout << "Первый тест на работу геттеров и сеттеров базового класса провален." << "\n";
	}
	CMetod test_metod;
	test_metod.add_el(firstcountry);
	test_metod.print_all();
	std::cout << "Если перед этим сообщением на экран вывелась информация о то файле методы add_el, print_all и get_to_Screen работают корректно." << "\n";
	test_metod.del_all();
	test_metod.print_all();
	std::cout << "Если перед этим сообщение на экран не выводились новые числа то методы del_all и remove_el работают корректно." << "\n";
	if (_CrtDumpMemoryLeaks())
	{
		std::cout << "Утечка памяти обнаружена." << "\n";
	}
	else
	{
		std::cout << "Утечка памяти не обнаружена." << "\n";
	}
	int t;
	std::cin >> t;
}
```

4. Результати роботи програми

Результати роботи програми:

```
_-_-_-Выберите желаемую опцию:_-_-_-_-_-_-_-_-_-_
_-1 - добавить элемент в список.-_-_-_-_-_-_-_-_-_
_-2 - получить элемент в списке по индексу.-_-
_-3 - удалить элемент из списка.-_-_-_-_-_-_-_-_-_
_-4 - показать все элементы списка.-_-_-_-_-_-_-
_-5 - завершить работу программы.-_-_-_-_-_-_-_-
4
Number_of_cities Population Area      Unical_index
143              32686854   50338346  1
Number_of_cities Population Area      Unical_index
156              65745656   56765888  2
Number_of_cities Population Area      Unical_index
587              87534345   13254756  3
Number_of_cities Population Area      Unical_index
146              38352463   15683556  4
_-_-_-Выберите желаемую опцию:_-_-_-_-_-_-_-_-_-_
_-1 - добавить элемент в список.-_-_-_-_-_-_-_-_-_
_-2 - получить элемент в списке по индексу.-_-
_-3 - удалить элемент из списка.-_-_-_-_-_-_-_-_-_
_-4 - показать все элементы списка.-_-_-_-_-_-_-
_-5 - завершить работу программы.-_-_-_-_-_-_-_-
5
Утечка памяти не обнаружена.
```

Результати тестів:

```
Первый тест на работу геттеров и сеттеров базового класса пройден успешно.
Number_of_cities Population Area      Unical_index
100              20000000   32000000  19
Если перед этим сообщением на экран вывелась информация о то файле методы add_el, print_all и get_to_Screen работают корректно.
Если перед этим сообщение на экран не выводились новые числа то методы del_all и remove_el работают корректно.
Утечка памяти не обнаружена.
```

5. Висновки

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з класами та їх специфікаторами доступу, інкапсуляцією, константами. Програма протестована, витоків пам’яті немає, виконується без помилок.