СТЕКОВІ ОБ’ЄКТИ. КОНСТРУКТОР, ДЕСТРУКТОР, ВІДОБРАЖЕННЯ, ПЕРЕДАЧА

Лабораторна робота №1

Мета:

- навчитися створювати об’єкти
- отримати розуміння створення об’єкта на стеку, а також передачу об’єкта по значенню.

1 ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

Варіант 10. Створити клас даних <Ємність> та клас відображення даних <View>. Об’єкт відображення конструюється на стеку функції main() об’єктом даних, що заздалегідь створений на стеку. Передавати <Ємність> як значення. <Ємність> має всі публічні поля та методи. <View> лише виконує відображення даних у форматі <Назва поля>=<Значення>.

2 РОЗРОБКА ПРОГРАМИ

Для реалізації програми було створено:

- клас <Ємність> з полями одиниці вимірювання, об’єм;
- клас <View> з методом відображення даних;

2.1 Засоби

У розробленій програмі використані наступні засоби ООП:

- розділення програми на ієрархію класів (інкапсуляція);

2.2 Ієрархія та структура класів

На рис.2.2 наведена ієрархія розроблених класів

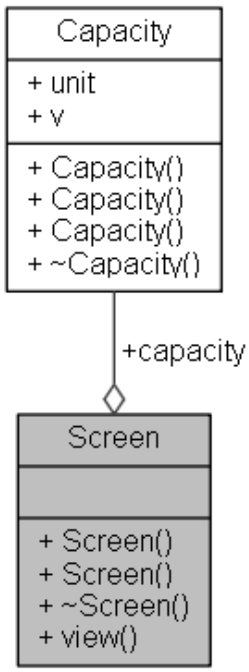

Рисунок 2.2 – Ієрархія класів

2.3 Опис програми

На рис.2.3 наведена структура розробленого проекту

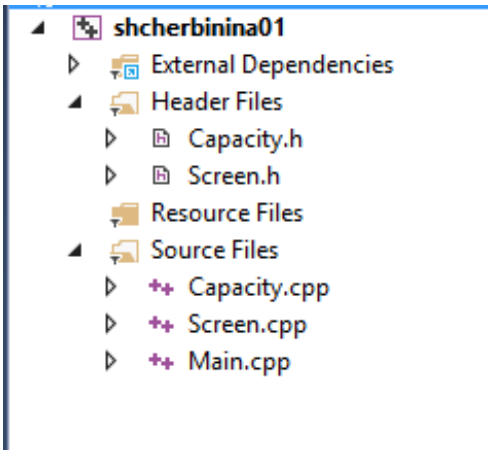

Рисунок 2.3 – Структура проекту

Призначення спроектованих класів наведено на рис.2.4

Capacity	Клас, що містить реалізацію ємності
Screen	Клас відображення інформації про об'єкт класу Capacity

Рисунок 2.4 – Призначення класів

2.4 Важливі фрагменти програми

2.4.1 Файл Capacity.h

```
/*
 * @file Capacity.h
 * Містить опис об'єкту класа Capacity
 * @date 2017.09.10
 * @author shcherbinina
 */

#ifndef CAPACITY_H_
#define CAPACITY_H_

#include <iostream>
using namespace std;
enum units { ml, l };

/**
* Клас, що містить реалізацію ємності.
*/
class Capacity {
	/**
	* Перевантаження оператора виводу для коректного відображення об'єкта класу
	* @param out - поток виводу
	* @param capacity - об'єкт, що виводиться
	* @return форматований поток із даними
	*/
	friend ostream& operator <<(ostream& out, Capacity capacity);
public:
	/**
		* Конструктор без параметра
		*/
	Capacity();
	/**
		* Конструктор з параметрами
		* @param _unit - Задає одиницю вимірювання
		* @param _v - Задає об'єм
		*/
	Capacity(units _unit, float _v);
	/**
		* Конструктор копіювання
		* @param _Capacity - Вихідний об'єкт із даними для копіювання
		*/
	Capacity(const Capacity& _Capacity);
	/**
		* Деструктор
		*/
	virtual ~Capacity();
public:
	///одиниця вимірювання
	units unit;
	///об'єм
	float v;
};

#endif /* CAPACITY_H_ */
```

2.4.2 Файл Capacity.cpp

```
/*
* @file Capacity.cpp
* Містить реалізацію класа Capacity
```

```
* @date 2017.09.10
* @author shcherbinina
*/

#include "Capacity.h"
#include <iostream>

using namespace std;

///Конструктор без параметра
Capacity::Capacity() : unit(l), v(1){
	cout << "Capacity constructor\n";
}
///Деструктор
Capacity::~Capacity() {
	cout << "Capacity destructor\n";
}
///Конструктор копіювання
Capacity::Capacity(const Capacity& capacity) : unit(capacity.unit), v(capacity.v){
	cout << "Capacity copy constructor\n";
}
///Конструктор із параметрами
Capacity::Capacity(units _unit, float _v) : unit(_unit), v(_v){
	cout << "Capacity constructor with parametrs\n";
}
///Перевантаження оператора виводу для коректного відображення об'єкта класу
ostream& operator <<(ostream& out, Capacity capacity){
	switch (capacity.unit){
	case ml: return (out << "Об'єм = " << capacity.v << " мл" << endl); break;
	case l: return (out << "Об'єм = " << capacity.v << " л" << endl); break;
	}
	return out;
}
```

2.4.3 Файл Screen.h

```
/*
 * Screen.h
 * Created on: 10 сент. 2017 г.
 * Author: shcherbinina
 * Description: Screen declaration
 */

#ifndef CSCREEN_H_
#define CSCREEN_H_

#include "Capacity.h"

/**
* Клас відображення інформації про об'єкт класу Capacity
*/
class Screen {
public:
	/**
	* Конструктор без параметра
	*/
	Screen();
	/**
	* Конструктор для всіх полей
	* @param capacity
	*/
	Screen(Capacity capacity);
	/**
	* Деструктор
	*/
```

```
	virtual ~Screen();
	/**
	 * Функція відображення даних об'єкта класу Capacity
	 */
	void view();
public:
	///Об'єкт класу, що має бути відображений
	Capacity capacity;
};

#endif /* CSCREEN_H_ */
```

2.4.4 Файл Screen.cpp

```
/*
 * CScreen.cpp
 * Created on: 10 сент. 2017 г.
 * Author: shcherbinina
 * Description: CScreen implementation
 */


#include <iostream>
#include "Screen.h"

using namespace std;

///Конструктор без параметра
Screen::Screen() {
	cout << "Screen constructor\n";
}
///Конструктор для всіх полей
Screen::Screen(Capacity capacity) : capacity(capacity) {
	cout << "Screen constructor with parametrs\n";
}
///Деструктор
Screen::~Screen() {
	cout << "Screen destructor\n";
}

///Функція відображення даних об'єкта класу Capacity
void Screen::view() {
	switch (capacity.unit){
	case ml: cout << "Об'єм = " << capacity.v << " мл" << endl; break;
	case l: cout << "Об'єм = " << capacity.v << " л" << endl; break;
	}
}
```

2.4.5 Файл Main.cpp

```
/*
 * @file Main.cpp
 * Точка входу в програму
 * @date 2017.09.10
 * @author shcherbinina
 */
#include "Screen.h"
#include <iostream>


/**
* Точка входу в програму
```

```
*/
int main() {
	setlocale(LC_ALL, "Russian");

	Capacity capacity(ml, 200);
	Screen view(capacity);
	view.view();

	system("pause");
	return 0;
}
```

3 РЕЗУЛЬТАТИ РОБОТИ

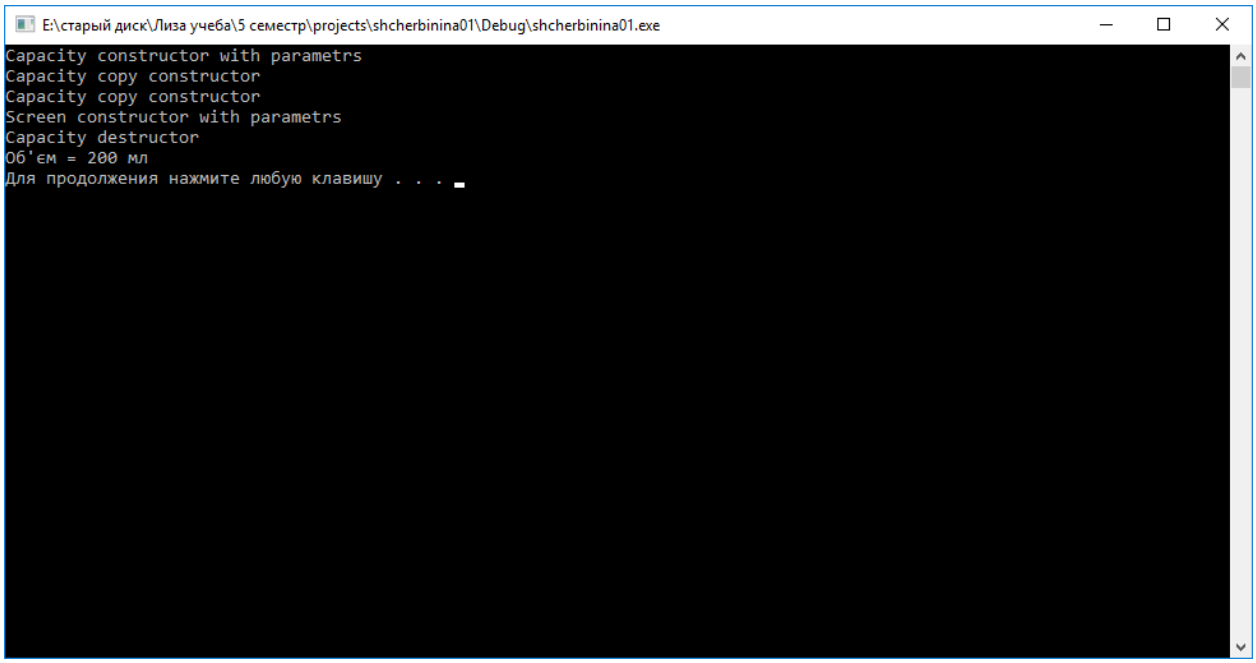

Рисунок 3.1 – Приклад роботи програми

ВИСНОВКИ

В розробленій програмі я навчилася створювати об’єкти, отримала розуміння створення об’єктів на стеку, використання списків ініціалізації, а також передачі об’єкта по значенню.